Teoría de la Información y Codificación

Cándido López ♢ Manuel Veiga

Universidad de Vigo

Teoría de la Información y Codificación

Cándido López García
Manuel Fernández Veiga

UNIVERSIDAD DE VIGO

A Jacobo López

Índice general

Prefacio

Acaso por ser la capacidad de comunicación la facultad más propiamente humana, ha existido desde la antigüedad un fuerte interés en desvelar la naturaleza de los procesos de transmisión de la información, a menudo siguiendo más una aproximación filosófica que científica. Sin embargo, no es hasta mediados del siglo XX cuando el progreso técnico en los medios y sistemas de telecomunicación modernos demanda la formulación firme y precisa del concepto de información y, en general, la concepción de una teoría objetiva de la comunicación. Este libro versa precisamente sobre los fundamentos matemáticos de los procesos de transmisión digital de la información. En él pretendemos exponer los contenidos fundamentales de dos cuerpos teóricos complementarios (la Teoría de la Información y la Teoría de la Codificación), cuyo origen se sitúa en un monumental artículo de C. E. Shannon (1916–2001) publicado en 1948 con el título *A Mathematical Theory of Communication*. En este artículo, Shannon establece los principales conceptos y resultados de la Teoría de la Información, y sienta las bases para el posterior desarrollo de la Teoría de la Codificación. Ambas teorías contienen los conceptos, modelos y resultados que han hecho posible la presente era de la comunicación digital.

La obra se ha organizado en tres partes:

- En la primera se desarrollan los principios de la Teoría de la Información. La finalidad es plantear una medida objetiva de la información y estudiar sus propiedades matemáticas para analizar los límites teóricos de cualquier proceso de transmisión fiel de la información sobre canales discretos, tanto ideales como ruidosos. En esta parte se esbozarán además las principales técnicas de compresión de datos.

- La segunda parte presenta, en un primer bloque, las principales técnicas algebraicas de codificación de canal para la detección y corrección de errores de transmisión (códigos lineales y códigos cíclicos). En el planteamiento se persigue un equilibrio entre la exposición de los prin-

cipios teóricos y la indispensable aplicación práctica de tales técnicas a la construcción de dispositivos reales eficaces de codificación y decodificación.

En un segundo bloque se discuten las características básicas de los protocolos de retransmisión, tanto desde la perspectiva de la corrección lógica como de la obtención cuantitativa de su rendimiento.

- La tercera parte está dedicada a técnicas de codificación avanzadas, de gran interés práctico por su aplicación en numerosos sistemas reales. Su objetivo es introducir las propiedades fundamentales de los códigos BCH, los códigos Reed–Solomon y los códigos convolucionales, con un interés especial en explicar métodos de decodificación eficiente.

Se ha adoptado en la elaboración del texto un enfoque académico, que se concreta en el desarrollo sistemático y completo de la materia, la demostración rigurosa de los resultados, el énfasis en los conceptos básicos y las relaciones entre ellos, las referencias a obras de consulta más especializadas, y la utilización de abundantes ejemplos ilustrativos, tablas y diagramas a lo largo de cada capítulo. Esta obra se complementa además con un libro de problemas para verificar la comprensión de la teoría.

El libro se ha pensado, fundamentalmente, como libro de texto para cursos básicos de Transmisión de Datos en los estudios de Ingeniería de Telecomunicación e Ingeniería Informática. Así, las dos primeras partes contienen el material básico necesario para la comprensión cabal del problema de la comunicación digital fiable y eficiente. Por este motivo, excepción hecha de algunas secciones o apéndices que incluyen material suplementario, se ha empleado en las dos primeras partes un lenguaje matemático simple y directo.

Pero, además, pensando también en quienes necesiten profundizar más en el conocimiento de la materia (profesionales, estudiantes de cursos especializados o, incluso, estudiantes de doctorado), se han incluido en el texto algunas secciones y capítulos avanzados, que en cursos elementales habrá que soslayar. Los capítulos más avanzados se han relegado a la parte tercera; las secciones de los restantes capítulos que, por su dificultad, no forman parte esencial de una primera lectura se han identificado con asterisco o se han incluido como apéndices.

De esta manera, el libro ofrece dos niveles de lectura: uno introductorio y otro más avanzado. Según creemos, esto le confiere un valor pedagógico añadido como obra de consulta o referencia puntual, y le aporta suficiente flexibilidad para adaptarse a diferentes programas.

Este libro es fruto de la experiencia docente acumulada durante más de una década de impartición de la asignatura de Fundamentos de Telemática

en la E.T.S.I. de Telecomunicación de la Universidad de Vigo. Y, al final del camino, justo es proclamar que el libro no habría sido escrito sin la contribución de muchas personas que inspiraron nuestro trabajo. En varios de los capítulos el lector avezado reconocerá un tributo especial a los admirables textos, ya clásicos, de R. Gallager y de F. J. MacWilliams y N. J. A. Sloane, que han tenido una influencia decisiva en nuestra comprensión básica de la disciplina. En nuestro entorno más cercano, agradecemos la inestimable y desinteresada colaboración de Raúl F. Rodríguez Rubio y Alberto Gil Solla, compañeros de labores docentes durante algunos años, quienes aplicaron sus conocimientos y una buena parte de su tiempo a revisar los borradores de la obra. Al resto de compañeros del Grupo de Redes e Ingeniería del Software —Ana F. Vilas, Andrés Suárez, Belén Barragáns, Estela Sousa, Jorge García, José C. L. Ardao, José J. Pazos, Manolo Ramos y Rebeca Díaz— les damos las gracias por tejer un ámbito de trabajo tan grato y estimulante para todos.

Cándido López García
Manuel Fernández Veiga
Vigo, junio de 2002

CAPÍTULO 1

Introducción

Es el propósito de este libro abordar el análisis matemático de algunas cuestiones fundamentales de la ingeniería de los sistemas de comunicación, entendiéndose aquí por comunicación toda transmisión de señales mediante un código común al emisor y al receptor.[1] Así, si se deja de lado el proceso de configuración de los estados de emisor y receptor que las señales representan (cuestión ésta que, en el caso de la comunicación humana, es más propia del campo de la Psicología), admitiéndose como punto de partida que emisor y receptor comparten un mismo espacio de estados que se conoce de antemano, la cuestión fundamental de la comunicación es la elección del código, es decir, la elección de las señales adecuadas para la representación de los distintos estados de emisor y receptor.

El medio físico que se utiliza para la transmisión de las señales entre emisor y receptor se denomina canal de comunicación o simplemente *canal*. Como es lógico, la elección del código idóneo estará determinada por la naturaleza del canal. En consecuencia, resulta también fundamental disponer de canales adecuados.

Precisamente, en su afán por ampliar su capacidad de comunicación, el hombre ha ideado diversos procedimientos para la transmisión de señales a grandes distancias de forma prácticamente instantánea. Para designar tales procedimientos, que hoy día constituyen una rama fundamental de la Técnica, se ha acuñado el término telecomunicación.[2] El objetivo de la

[1] Tercera acepción del *Diccionario de la Lengua Española* de la Real Academia Española.

[2] La U.I.T. (Unión Internacional de Telecomunicación) ha definido la telecomunicación como «toda emisión, transmisión y recepción de signos, señales, escritos e imágenes, sonidos e informaciones de cualquier naturaleza, por hilo, radioelectricidad, medios ópticos u otros sistemas electromagnéticos».

Ingeniería de Telecomunicación es, en definitiva, la creación de nuevos y mejores canales de comunicación y el uso eficiente de los mismos.

En función del tipo de señal que pueden transmitir (analógica o digital), se distinguen dos tipos de canales, que se denominan, respectivamente, canales analógicos y canales digitales. En la práctica, es fácil encontrar casos de canales tanto de un tipo como de otro. Sin embargo, la posibilidad de digitalizar[3] cualquier señal analógica, dadas las ventajas que comporta la utilización de señales digitales,[4] ha desembocado en que los sistemas de telecomunicación actuales sean predominantemente digitales y estén basados tanto en el empleo de redes de ordenadores como en la digitalización de la información. Así, en este libro se considerará únicamente el caso de la utilización de canales digitales, y más concretamente de canales punto a punto,[5] y la cuestión de interés será la óptima utilización de los mismos.

1.1. Un modelo de sistema de comunicación

En definitiva, el objetivo de este libro es abordar un análisis matemático general de los sistemas de transmisión de señales o mensajes digitales en los que se utilizan canales punto a punto. Empecemos, pues, por formular una abstracción conveniente de los dos elementos básicos de tales sistemas: el mensaje digital y el canal punto a punto.

Mensaje digital: Se denominará mensaje digital a cualquier secuencia de elementos de un cierto *alfabeto*, entendiéndose a su vez por alfabeto

[3] Se entiende por digitalización la conversión de una señal analógica en digital. El proceso de digitalización tiene dos fases diferenciadas: la fase de muestreo (el teorema de muestreo de Nyquist establece que se puede recuperar cualquier señal analógica a partir de una secuencia de valores puntuales de la misma convenientemente equiespaciados) y la fase de cuantificación (cada uno de los valores de la secuencia anterior se representa, con tanta fidelidad como se requiera, mediante una cierta secuencia de bits).

[4] Actualmente, el empleo de señales digitales para la comunicación goza de varias importantes ventajas: primero, los ordenadores pueden manipular fácilmente cualquier tipo de información digital; además, cualquier información digital se puede copiar o transmitir tantas veces como se quiera sin sufrir ningún tipo de alteración; por último, se pueden emplear los mismos procedimientos, es decir, una red única, para el envío de cualquier tipo de información digital.

[5] Se denominan canales punto a punto los canales con un único emisor y un único receptor. Si bien estos canales son los que se utilizan en mayor medida para la construcción de redes de comunicaciones, algunos sistemas de comunicación se basan en otro tipo de canales, con múltiples emisores y/o receptores, que se denominan típicamente canales multipunto o canales de acceso múltiple. La utilización de canales de acceso múltiple no es más que un caso particular de una disciplina más general que se ocupa de la construcción de redes de comunicaciones. Desde el punto de vista de la Teoría de la Información, el problema general de las comunicaciones en red, del que se ocupa la Teoría de la Información de Red, no está todavía bien resuelto. En todo caso, tal problema cae fuera del ámbito de este libro.

cualquier conjunto numerable y finito de símbolos. El número de elementos que forman la secuencia se denominará longitud del mensaje.

Se considerará únicamente el caso de que las características individuales de los símbolos sean intrascendentes para la transmisión, siendo así lo único importante de los alfabetos su tamaño. En el caso particular (muy habitual, por otra parte) de que el alfabeto tenga sólo dos elementos, se denomina alfabeto binario. Los símbolos de un alfabeto binario se representan típicamente por los dígitos 0 y 1, que reciben el nombre de bits.[6]

Los mensajes digitales se suponen generados por un dispositivo que denominaremos *fuente discreta*. Así pues, se supondrá la existencia de una fuente discreta capaz de generar de forma ininterrumpida símbolos de un cierto alfabeto, característico de la fuente, que se denomina habitualmente alfabeto de la fuente. Naturalmente, las fuentes discretas de interés generan los mensajes de forma aleatoria, ya que de lo contrario sería innecesaria su transmisión.

Canal punto a punto: Se denominará canal punto a punto cualquier medio físico apto para la transmisión de mensajes de un punto (fuente) a otro (destino).

Si por el canal sólo pueden transmitirse mensajes digitales, se dice que se trata de un canal digital, también conocido como *canal discreto*, que será la denominación que se empleará preferentemente en el resto del libro.

El canal punto a punto discreto estará caracterizado por un cierto alfabeto (el conjunto de los símbolos que puede transmitir), así como por un cierto régimen uniforme de transmisión (la inversa del tiempo necesario para la transmisión de cada símbolo) y por un cierto retardo de propagación[7] (el tiempo necesario para que un símbolo atraviese el canal). Idealmente, el mensaje que sale del canal debería ser siempre idéntico al que ha entrado;[8] si así fuese, se diría que el canal empleado es un canal ideal. Desafortunadamente, no ocurre siempre así; sino que, a veces y de forma aleatoria, el mensaje que sale del canal es distinto del que ha entrado. A la causa de tal efecto se la denomina genéricamente ruido; y el canal que se ve afectado por algún tipo de ruido se denomina canal ruidoso. Aun más, por razones de generalidad del análisis, vamos a suponer que no sólo el mensaje que sale del canal puede ser distinto del que ha entrado, sino que también puede contener

[6] El termino bit procede de la contracción de las palabras inglesas *binary digit* (dígito binario), y se usa para denominar a cualquiera de los símbolos de un alfabeto binario.

[7] Este retardo no tiene ninguna relevancia para la mayor parte de las cuestiones consideradas en este libro, por lo que en tales casos se ignorará su existencia.

[8] O el mensaje entrante debería poder deducirse unívocamente del mensaje saliente.

símbolos de un alfabeto distinto al utilizado para conformar el mensaje que se transmite.[9] Por tanto, y además del régimen de transmisión y el retardo, un canal discreto estará caracterizado por:

- un alfabeto de entrada, con un número de elementos igual al de símbolos distintos que puedan formar parte de los mensajes que se introducen en el canal;
- un alfabeto de salida (no necesariamente igual al alfabeto de entrada), con un número de elementos igual al de símbolos distintos que se puedan identificar en los mensajes que salen del canal;
- y una representación estocástica del ruido que afecta a la transmisión de los mensajes.

1.1.1. Requisitos del sistema

Para que resulte satisfactorio, un sistema de transmisión de mensajes digitales debe cumplir determinados requisitos, fundamentalmente de fiabilidad y coste.[10] Tales requisitos dependen, en general, de la naturaleza de los mensajes que se van a transmitir. En algunos casos se pueden producir distorsiones en los mensajes transmitidos que son, no obstante, imperceptibles para el destinatario de los mismos, o cuando menos tolerables. En tales casos, puede aprovecharse la tolerancia al error del destinatario para la consecución de menores costes de transmisión de los mensajes. Aun así, en este texto se considerarán únicamente aquellos sistemas de transmisión de mensajes digitales cuyo objetivo es la reproducción fiel en el punto de destino del mensaje generado por la fuente. Es decir, en este texto nos interesaremos por determinar en qué condiciones y cómo es posible la consecución de una transmisión fiable. Y además, aun sin establecer una restricción de coste a priori, nos interesa determinar la manera más económica de hacerlo. Establezcamos, pues, tanto un criterio de fidelidad en la copia de mensajes como una medida del coste de uso de un canal dado.

Criterio de fidelidad: Se dirá que es posible la reproducción fiel de los mensajes cuando la probabilidad de que la copia del mensaje que se obtiene en el punto de destino sea distinta del mensaje originalmente

[9] Aunque puede parecer sorprendente a primera vista, existen situaciones prácticas en las que conviene considerar en la salida símbolos adicionales a los de entrada (véase 11.6).

[10] A veces, es también importante garantizar el secreto o la autenticidad de un mensaje, y se han desarrollado una serie de técnicas matemáticas para abordar la solución de estos problemas. No obstante, tanto porque tales problemas son de diferente índole de los que aquí se van a considerar, como por la amplitud de los mismos, se han excluido del presente texto.

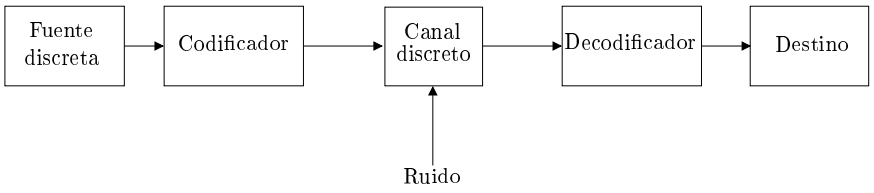


Figura 1.1.

enviado por la fuente pueda hacerse tan pequeña como se quiera. Obsérvese que, en rigor, no se exige que la copia del mensaje sea idéntica al original siempre; si bien, a efectos prácticos, puede suponerse así.

Coste de transmisión: Se supondrá que el coste de la transmisión de un mensaje es la suma de los costes de las transmisiones sucesivas de los símbolos que lo componen; y se supondrá que el coste de transmisión de todos los símbolos es siempre el mismo, para un canal dado. Entonces, el coste de transmisión de los mensajes es proporcional al número de símbolos que lo componen, es decir, a su longitud.

1.1.2. Codificación de los mensajes

Como un mensaje digital no es sino la representación simbólica de un cierto estado de una fuente, y como en nuestro estudio se supondrá que tanto la fuente como el canal están dados de forma independiente, debe admitirse que la representación de los estados de la fuente es independiente de la naturaleza del canal. Puede ocurrir, por tanto, que tales representaciones no sean las más convenientes para el propósito que se persigue (transmisión fiel y eficiente de los mensajes de la fuente). En tales casos, cabe considerar la posibilidad de encontrar una forma alternativa para su representación (codificación) que sirva mejor a nuestros propósitos. El proceso de codificación de los mensajes de una fuente consistirá en la aplicación de alguna regla de transformación para la reescritura de los mensajes de la fuente utilizando como alfabeto de codificación el alfabeto de entrada del canal. Tal regla deberá ser, obviamente, invertible; es decir, en el destino ha de ser siempre posible recuperar la forma original del mensaje (decodificación).

En definitiva, el sistema de transmisión de mensajes que vamos a considerar puede esquematizarse de la siguiente manera (figura 1.1):

- La **fuente discreta** genera, aleatoriamente, mensajes digitales arbitrariamente largos, es decir, secuencias arbitrariamente largas de símbolos de un cierto alfabeto (alfabeto de la fuente).

- El **codificador** transforma los mensajes generados por la fuente; es decir, convierte una secuencia de símbolos dada en otra distinta (en general, con símbolos de un alfabeto también distinto del alfabeto de la fuente, que se denomina alfabeto de codificación, y que habrá de ser necesariamente igual que el alfabeto de entrada del canal). Esta transformación, cuyo objetivo es la transmisión fiel y eficiente de los mensajes de la fuente dada, utilizando el canal dado, deberá hacerse teniendo en cuenta tanto las características estadísticas de los mensajes de la fuente (es decir, de la fuente) como las del ruido que se produce en el canal.
- El **canal discreto ruidoso** realiza, aleatoriamente, una transformación indeseable sobre los mensajes que lo atraviesan, de forma que el mensaje que recibe el decodificador no es, en general, el mismo que ha generado el codificador.
- El **decodificador** tiene que adivinar qué mensaje de la fuente corresponde al mensaje recibido, conocidas las reglas de transformación que aplica el codificador.
- Finalmente, el **destino** simplemente recoge el mensaje que le entrega el decodificador.

La posibilidad de construir un codificador y el correspondiente decodificador con unas determinadas prestaciones, dados un canal y una fuente, así como las cuestiones de diseño de los mismos, serán precisamente el objeto de nuestro interés. Ahora bien, en la práctica, resulta conveniente aislar el efecto que tiene la fuente en el sistema del efecto que tiene el canal, descomponiendo el codificador (y, correspondientemente, también el decodificador) de la figura 1.1 en dos partes:

- un **codificador de fuente**, cuya construcción ha de adaptarse a las características de la fuente de forma que consiga representar los mensajes de la fuente con el menor número posible de símbolos de un cierto alfabeto de codificación (típicamente los símbolos que se pueden transmitir por el canal dado); y
- un **codificador de canal**, cuya construcción ha de adaptarse a las características del canal de forma que permita la consecución de una transmisión fiable sobre un canal ruidoso con un coste mínimo.

La descomposición del proceso de codificación en dos subprocesos independientes, codificación de fuente y codificación de canal (figura 1.2), tal y como se acaba de exponer, facilita el análisis del sistema, por una parte; y, por otra parte, proporciona una gran flexibilidad en la implementación,

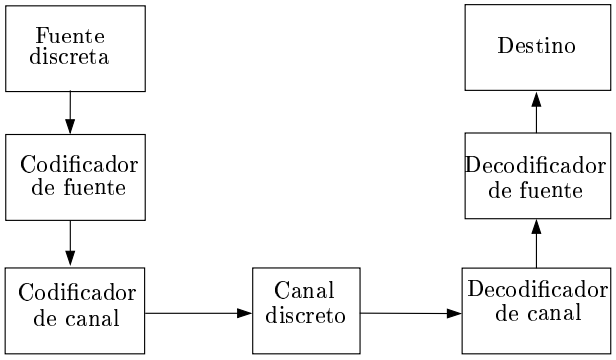


Figura 1.2.

porque hace virtualmente independientes los diseños del codificador de canal y del codificador de fuente. Ahora bien, decidir si esta forma de proceder impone alguna limitación a las prestaciones del sistema de transmisión de información no es una cuestión trivial. Sin embargo, uno de los resultados importantes de la teoría que se va a exponer es que tal división puede hacerse sin menoscabo de las prestaciones del sistema.

El problema de la comunicación, tal y como está planteado, fue formulado y resuelto originalmente por el matemático e ingeniero estadounidense C. E. Shannon (1916–2001)[11] en un artículo celebérrimo publicado en la revista *Bell Systems Technical Journal* en 1948.[12] En dicho artículo, Shannon formula los conceptos de entropía de la fuente y capacidad de canal; conceptos que cobran significado físico en dos teoremas fundamentales, hoy conocidos, respectivamente, como primer y segundo teoremas de Shannon o como teoremas de Shannon de codificación de fuente y de canal. El primero establece la longitud mínima de codificación sin pérdidas de los mensajes de una fuente dada; y el segundo, la redundancia mínima necesaria para la comunicación fiable de los mensajes. Este trabajo ha sido el germen de un nuevo campo de las Matemáticas que hoy se conoce como Teoría de la Información.[13]

[11] Recientemente, R. G. Gallager, profesor del MIT y antiguo colaborador de Shannon, ha publicado un artículo [27] en el que glosa la trascendencia de su obra. Unos años antes, en 1993, se publicaba una recopilación de sus escritos [65].

[12] El artículo apareció publicado en dos partes: la primera [59] en julio y la segunda [60] en octubre.

[13] Si bien Shannon creó la Teoría de la Información para resolver una serie de cuestiones fundamentales de la ingeniería de los sistemas de comunicación, que son asimismo las que a nosotros nos ocupan en este libro, con el paso del tiempo esta teoría se ha enriquecido considerablemente, encontrando también aplicaciones en campos diversos como la Estadística o la Economía.

1.2. Los modelos matemáticos de las fuentes discretas y los canales discretos

Se exponen en este apartado los modelos matemáticos que se utilizarán en nuestro estudio para la representación de las fuentes discretas y los canales discretos. Mas debe advertirse que no es cuestión que vaya a tratarse en este libro el problema general del modelado de las fuentes y de los canales reales (es decir, el problema de cómo decidir cuál es el modelo que mejor caracteriza una fuente real o un canal real, lo que, además, es fuertemente dependiente del problema que se considere); sino que se supondrá que los modelos matemáticos de los mismos están dados. De hecho, cuando en este libro se habla de una fuente dada o de un canal dado, se habla en realidad de un modelo dado de fuente o de un modelo dado de canal. Aun más, inicialmente consideraremos únicamente los modelos más sencillos, tanto de fuente como de canal, con el fin de reducir al máximo la complejidad matemática de nuestro análisis.

1.2.1. Los modelos de las fuentes discretas

Definición 1.1 (Fuente discreta). *Una fuente discreta es un dispositivo que emite con regularidad y de forma aleatoria símbolos pertenecientes a un cierto conjunto discreto y finito llamado alfabeto de la fuente.*

Una fuente discreta podrá representarse matemáticamente, por tanto, mediante un proceso estocástico de tiempo discreto, es decir, mediante una secuencia de variables aleatorias $\{X_i\}_{i=1,2,\ldots}$, donde X_i es la variable aleatoria discreta que representa el i-ésimo símbolo del mensaje de la fuente.

En la práctica, no resulta habitualmente sencillo desarrollar un modelo totalmente preciso de una fuente, ni tampoco desarrollar esquemas de codificación convenientes que aprovechen todas las características de una fuente real. Típicamente, los esquemas de codificación más empleados soslayan la relación entre los sucesivos símbolos de los mensajes (memoria), limitándose a capturar la frecuencia de aparición de los distintos símbolos para conseguir una mayor eficiencia de codificación a través de la asignación de representaciones más cortas a los símbolos más frecuentes.[14]

Por tal motivo, en una primera etapa, nosotros vamos a interesarnos únicamente por un tipo particular de fuentes discretas, las más simples, que son las denominadas fuentes discretas sin memoria.

Definición 1.2 (Fuente discreta sin memoria). *Una fuente discreta sin memoria es una fuente discreta en la que la emisión de cada símbolo es*

[14] Tal es el caso, por ejemplo, del conocido sistema telegráfico de Morse.

independiente de los símbolos que se hayan emitido con anterioridad.

En este caso, las variables aleatorias X_i de la secuencia son independientes y tienen idéntica distribución, por lo que una fuente discreta sin memoria queda perfectamente representada mediante una variable aleatoria discreta que representa la emisión de un símbolo de la fuente en cualquier instante de tiempo.

Esta restricción, sin embargo, no es tan severa como pudiera parecer, porque se puede obtener de esta forma una aproximación tan buena como se quiera a una fuente con memoria por el simple artificio de considerar que la fuente no produce símbolos elementales, sino pares, ternas, etc., con las probabilidades adecuadas.[15] No obstante lo anterior, existen modelos matemáticos más adecuados para la caracterización de las fuentes con memoria, como las cadenas de Markov, que finalmente también serán considerados.

Ejemplo 1.1 (Fuente binaria simétrica). Se denomina fuente binaria simétrica a una fuente de alfabeto binario que, en todo instante, genera ambos símbolos equiprobablemente. Así, una fuente binaria simétrica se representaría por una variable aleatoria X de rango $\mathcal{X} = \{0, 1\}$ y función de probabilidad $p(x)$:

$$p(x) = \mathrm{P}(X = x) = 1/2 \quad \forall x \in \mathcal{X}. \qquad \blacksquare$$

1.2.2. Los modelos de los canales discretos

Definición 1.3 (Canal discreto). *Se denomina canal discreto al canal por el que sólo se pueden enviar secuencias de símbolos de un cierto alfabeto. Nosotros supondremos, además, que tales símbolos se envían con un régimen uniforme, que es característico del canal.*

Si el símbolo recibido es siempre igual al enviado, o el símbolo enviado se deduce unívocamente del símbolo recibido, se tiene un canal ideal; en caso contrario, se dice que el canal es ruidoso.

En general, el comportamiento del canal en un cierto instante puede depender de la historia pasada; por lo que deberá considerarse el canal, en general, como un autómata (que se supone finito). Así, un canal discreto ruidoso podrá representarse por medio de una serie de distribuciones de probabilidad $\{\mathrm{P}_{\alpha,i}(\beta, j)\}$, una para cada posible estado del canal, donde $\mathrm{P}_{\alpha,i}(\beta, j)$ representa la probabilidad de que, estando el canal en el estado

[15] El mismo Shannon [59] pone como ejemplo una imitación del idioma inglés a través de una fuente que produce aleatoriamente pares, ternas, etc., de letras.

α, al transmitir el símbolo i, se reciba el símbolo j y el canal pase al estado β.

Pero, en primera aproximación, y para simplificar el estudio, se considerará que el comportamiento del canal es el mismo en todo instante.[16] Un canal con tal característica se denomina canal discreto sin memoria.

Definición 1.4 (Canal discreto ruidoso sin memoria). *Un canal discreto ruidoso sin memoria es un canal discreto ruidoso en el que el ruido afecta de forma independiente a la transmisión de todos los símbolos, de manera que se verifica*

$$\mathrm{P}\left((y^1, y^2, \ldots, y^n) \,|\, (x^1, x^2, \ldots, x^n)\right) = \prod_{i=1}^{n} \mathrm{P}(y^i \,|\, x^i)$$

siendo $(y^1, y^2, \ldots, y^n)$ la secuencia de salida del canal correspondiente a la secuencia de entrada $(x^1, x^2, \ldots, x^n)$, para cualquier valor de n.

En este caso el canal tendrá un único estado, por lo que quedará perfectamente representado mediante la especificación, para cada uno de los símbolos del alfabeto de entrada $\mathcal{X} = \{x_1, x_2, \ldots, x_i, \ldots, x_r\}$, de las probabilidades de que su transmisión dé lugar a los distintos símbolos del alfabeto de salida $\mathcal{Y} = \{y_1, y_2, \ldots, y_j, \ldots, y_s\}$. Tal especificación suele adoptar la forma de una matriz de probabilidades como la siguiente:

$$Q_{r\times s} = \begin{pmatrix} p(y_1 \,|\, x_1) & \cdots & p(y_j \,|\, x_1) & \cdots & p(y_s \,|\, x_1) \\ \vdots & & \vdots & & \vdots \\ p(y_1 \,|\, x_i) & \cdots & p(y_j \,|\, x_i) & \cdots & p(y_s \,|\, x_i) \\ \vdots & & \vdots & & \vdots \\ p(y_1 \,|\, x_r) & \cdots & p(y_j \,|\, x_r) & \cdots & p(y_s \,|\, x_r) \end{pmatrix}$$

donde las probabilidades $p(y_j \,|\, x_i)$, que representan la probabilidad de obtener el símbolo y_j a la salida del canal cuando se ha transmitido el símbolo x_i, se denominan probabilidades de transición del canal. Obsérvese que en tales matrices los elementos de cada fila suman siempre uno (matrices estocásticas).

Ejemplo 1.2 (Canal binario simétrico). En un canal binario simétrico, tanto el alfabeto de entrada como el de salida son binarios, y la probabilidad de error en la transmisión de cada símbolo es siempre la misma, tanto si

[16] Si bien en la práctica no es fácil encontrar canales sin memoria, puede conseguirse que, a los efectos de la transmisión de un mensaje, cualquier canal se comporte como un canal sin memoria con el simple artificio de entremezclar aleatoriamente los símbolos del mensaje antes de su transmisión.

se trata de un cero como si se trata de un uno. Esto es, su matriz de probabilidades de transición es de la forma

$$Q_{2\times 2} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}. \qquad \blacksquare$$

1.3. La codificación de fuente

El problema de la codificación de fuente es el de decidir la forma más conveniente de representar los mensajes de la fuente utilizando los símbolos que pueden transmitirse a través del canal (es decir, los símbolos de un cierto alfabeto de codificación) cuando se ignoran los problemas derivados de los errores en la transmisión.

Antes de formalizar el estudio del problema en capítulos posteriores, en este apartado vamos a tratar de explicar con la mayor claridad posible las características del mismo utilizando algunos ejemplos sencillos de transmisión de mensajes a través de un canal binario ideal, es decir, un canal cuyos alfabetos de entrada y salida coinciden y son binarios, y en el que la transmisión de cualquier símbolo siempre es correcta. Sea $\mathcal{B} = \{0, 1\}$ el alfabeto binario.

Considérese una fuente discreta sin memoria representada por la variable aleatoria discreta X de rango $\mathcal{X} = \{0, 1, 2, 3\}$ y función de probabilidad $p(x) = \mathrm{P}(X = x)$:

$$\begin{aligned} p(0) &= 0{,}7 \\ p(x) &= 0{,}1 \quad \forall\, x \in \{1, 2, 3\}. \end{aligned}$$

Un mensaje de la fuente de longitud k es cualquier secuencia de k símbolos sucesivos de $\mathcal{X}$, o lo que es lo mismo, cualquier símbolo de $\mathcal{X}^k$, donde $\mathcal{X}^k$ representa el producto cartesiano

$$\mathcal{X}^k = \overbrace{\mathcal{X} \times \mathcal{X} \times \ldots \times \mathcal{X}}^{k}.$$

Como por el canal sólo pueden transmitirse mensajes binarios (mensajes compuestos por símbolos binarios), para poder transmitir el mensaje de la fuente a través del canal, será preciso realizar una transformación que convierta el mensaje de la fuente en una secuencia de símbolos binarios. Pero, para obtener una copia fiel del mensaje en el destino, no basta que la transmisión sea fiable (canal ideal); sino que, además, la transformación en cuestión ha de ser invertible para cualquier valor de k, es decir, en recepción

debe poderse recuperar siempre el mensaje original a partir de su versión binaria: una regla de codificación tal se denomina *unívocamente decodificable*.

1.3.1. Codificación de longitud constante

Resulta obvio que una codificación válida puede conseguirse de manera muy simple asignando a cada uno de los símbolos del mensaje de la fuente (símbolos de $\mathcal{X}$) un par de símbolos del alfabeto $\mathcal{B}$ (alfabeto de codificación), o lo que es lo mismo, un símbolo del producto cartesiano $\mathcal{B}^2$. Por ejemplo:

$\mathcal{X}$	$\xrightarrow{\mathcal{C}_1}$	$\mathcal{C}_1(\mathcal{X}) = \mathcal{B}^2$
0	$\longrightarrow$	00
1	$\longrightarrow$	01
2	$\longrightarrow$	10
3	$\longrightarrow$	11

donde, para mayor comodidad en la notación, los pares de símbolos del producto cartesiano $\mathcal{B}^2$ se han representado sin los paréntesis y la coma de separación entre símbolos habituales, pero respetando, eso sí, el orden de los símbolos. Este mismo convenio se empleará a menudo en el resto del libro para la representación de cualquier n-tupla.

Procediendo de esta forma, para enviar un mensaje de la fuente de k símbolos, habremos de transmitir $2k$ símbolos a través del canal. Esta operación requerirá $2k$ unidades de tiempo (tomando como referencia temporal el tiempo necesario para la transmisión de un símbolo por el canal); o lo que es lo mismo, la velocidad del canal habrá de ser el doble de la de la fuente.

1.3.2. Codificación de longitud variable

Cabe preguntarse ahora si no habría alguna otra manera de realizar la codificación que nos permitiese ahorrar tiempo en la transmisión. Si se sigue manteniendo todavía la idea de realizar la transformación del mensaje de la fuente símbolo a símbolo,[17] la única manera de ahorrar tiempo es acortar la representación binaria de algún símbolo de $\mathcal{X}$. Pero, como con un solo bit no se pueden construir cuatro secuencias distintas, está claro que la única manera de poder asignar un solo símbolo binario a algún elemento de $\mathcal{X}$ es permitiendo secuencias de símbolos binarios de distintas longitudes, como se propone, a modo de ejemplo, en el siguiente esquema de codificación:

[17] Se verá a continuación que ésta no es la única manera en la que se puede proceder para realizar la codificación de los mensajes de la fuente, ya que también se pueden codificar los símbolos de las fuentes por grupos.

$\mathcal{X}$	$\xrightarrow{\mathcal{C}_2}$	$\mathcal{C}_2(\mathcal{X})$
0	$\longrightarrow$	0
1	$\longrightarrow$	101
2	$\longrightarrow$	110
3	$\longrightarrow$	111

En general, no es sencillo decidir si una transformación de este tipo es o no invertible; aunque, como se verá en el capítulo 3, existe una clase de códigos de interés en los que se decide de forma trivial la univocidad de la decodificación. Éste es precisamente uno de tales códigos. En cualquier caso, resulta fácil verificar que la transformación de cualquier mensaje realizada con el esquema que se propone en este ejemplo sí lo es. Pero ahora las secuencias de bits que codifican los símbolos de la fuente no tienen una longitud constante. Como consecuencia de ello, el número de dígitos binarios necesarios para representar un mensaje de la fuente de longitud k, representémoslo por l_k, no será una función determinista de k; sino que, para un valor dado de k, habrá varios valores posibles de l_k, pudiendo determinarse fácilmente la probabilidad de cada uno de ellos: en definitiva, l_k es ahora una variable aleatoria discreta.

Siendo así, l_k podrá ser mayor, menor o igual que $2k$, según cuál sea la secuencia de k símbolos generada por la fuente; es decir, en algunos casos la representación del mensaje será más eficiente que la anterior, pero en otros casos lo será menos. ¿Cómo se puede, entonces, determinar la conveniencia o no de este esquema? Si se tratase de enviar un único mensaje de longitud finita, tal cosa no sería posible; pero si se piensa en una fuente que genera símbolos de forma continuada, es decir, si se piensa que los mensajes pueden tener una longitud arbitrariamente grande (k tan grande como se quiera), entonces l_k, en virtud de la ley de los grandes números, deja de ser una variable aleatoria para convertirse en una constante de valor[18] $l_k = L \cdot k$ donde L es el número medio de dígitos binarios necesarios para representar los símbolos de $\mathcal{X}$. Obsérvese que L es, por tanto, el valor medio de la variable aleatoria l_1.

En tales condiciones, este último esquema de codificación será más eficiente que el anterior si $L < 2$. Veamos si es así. En este ejemplo, l_1 es una variable aleatoria discreta que puede tomar dos valores, es decir, el rango de la variable aleatoria l_1 es $\mathcal{L}_1 = \{1, 3\}$: tomará el valor 1 cuando el símbolo generado por la fuente sea el "0", lo que ocurre con probabilidad 0,7; y tomará el valor 3 en cualquier otro caso, es decir, con probabilidad 0,3. Por

[18] Para expresarlo con mayor rigor matemático, la ley (débil) de los grandes números establece que, dado cualquier $\varepsilon > 0$, se cumple que $\lim_{k\to\infty} \mathrm{P}\left(\left|\frac{l_k}{k} - L\right| < \varepsilon\right) = 1$.

tanto, en este caso,

$$L(\mathcal{C}_2) = 1 \times 0{,}7 + 3 \times 0{,}3 = 1{,}6$$

y, en consecuencia, se obtiene un ahorro de tiempo en el envío del mensaje del 20 %.

Se puede utilizar, entonces, como métrica de eficiencia de los códigos el valor del parámetro L; de forma que el código será tanto mejor cuanto menor sea el valor de L. Este parámetro del código se denominará en adelante *longitud del código*.

Obsérvese ahora que la obtención de un valor de L lo más pequeño posible requiere una asignación conveniente de las secuencias de bits a los símbolos de la fuente. Efectivamente, si en lugar de asignar la secuencia más corta al símbolo de la fuente más probable (el símbolo "0" en nuestro ejemplo), se hubiese asignado a cualquier otro, por ejemplo:

$\mathcal{X}$	$\xrightarrow{\mathcal{C}_3}$	$\mathcal{C}_3(\mathcal{X})$
0	$\longrightarrow$	100
1	$\longrightarrow$	101
2	$\longrightarrow$	110
3	$\longrightarrow$	0

se obtendría un código igualmente válido para una comunicación fiable, pero más ineficiente, ya que en este caso la longitud del código valdría

$$L(\mathcal{C}_3) = 3 \times 0{,}9 + 1 \times 0{,}1 = 2{,}8\,.$$

En resumidas cuentas, se acaba de mostrar que una codificación conveniente de los mensajes de una fuente permite ahorrar tiempo en su transmisión. Por tanto, dada una fuente, parece lógico preguntarse cuál es la codificación que proporciona un mayor ahorro de tiempo.

Para el caso particular de fuente que estamos considerando, y codificando los símbolos de la fuente de uno en uno, siempre de la misma forma, mediante secuencias de bits que pueden tener distintas longitudes, resulta relativamente sencillo convencerse de que no hay otro código unívocamente decodificable con una longitud L menor que el siguiente:

$\mathcal{X}$	$\xrightarrow{\mathcal{C}_4}$	$\mathcal{C}_4(\mathcal{X})$
0	$\longrightarrow$	0
1	$\longrightarrow$	10
2	$\longrightarrow$	110
3	$\longrightarrow$	111

$$L(\mathcal{C}_4) = 1 \times 0{,}7 + 2 \times 0{,}1 + 3 \times 0{,}2 = 1{,}5\,.$$

Los códigos como éste, cuya longitud L es la menor de las posibles, se denominan *códigos compactos*. Y, llegados a este punto, surgen algunas cuestiones fundamentales:

1. ¿No será posible encontrar un método que permita la obtención sistemática de un código compacto para una fuente cualquiera?

   Se verá más adelante, en el capítulo 3, que la respuesta es afirmativa; y se explicará además un método para la construcción de códigos compactos de una fuente cualquiera: el *algoritmo de Huffman*.

2. ¿Los códigos compactos obtenidos mediante la codificación de los símbolos de la fuente de uno en uno nos proporcionan siempre la representación más corta, en media, de los mensajes de la fuente?

   Vamos a ver inmediatamente a continuación que la respuesta a esta pregunta es negativa: se pueden obtener códigos más convenientes codificando los símbolos de la fuente en grupos.

1.3.3. Codificación de bloques de símbolos

En lugar de codificar los símbolos de la fuente uno a uno, podrían codificarse por grupos o en bloques; y, en tal caso, lo más sencillo es hacer la codificación utilizando bloques de longitud fija: el único requisito adicional del codificador sería una cierta cantidad de memoria.

Así, si se codifican, por ejemplo, los símbolos de la fuente por pares, la situación sería equivalente a la de tener una fuente distinta cuyo alfabeto vendría dado por el producto cartesiano $\mathcal{X}^2$ y cuya distribución de probabilidad sería

$$p(x,y) = p(x)p(y) \quad \forall\, x,\ y\ \in\ \mathcal{X}.$$

Esta nueva fuente así obtenida se dice que es una *extensión* de orden dos de la fuente considerada originalmente.

Si se considera ahora la fuente dada por la extensión de orden dos de la fuente cuaternaria que habíamos tomado como ejemplo, y a esta nueva fuente se le aplica el algoritmo de Huffman para la construcción de un código compacto, se obtiene el código de la tabla 1.1, cuya longitud (número medio de bits por símbolo de $\mathcal{X}^2$), que representaremos por L_2, es

$$\begin{aligned} L_2 &= 1 \times 0{,}49 + 6 \times 4 \times 0{,}07 + 7 \times 6 \times 0{,}01 + 2 \times 7 \times 0{,}01 \\ &= 2{,}73\,. \end{aligned}$$

$\mathcal{X}^2$	P(·)	$\mathcal{C}(\mathcal{X}^2)$
00	0,49	0
01	0,07	1000
02	0,07	1001
03	0,07	1010
10	0,07	1011
20	0,07	1100
30	0,07	1101
11	0,01	111000
12	0,01	111001
13	0,01	111010
21	0,01	111011
22	0,01	111100
23	0,01	111101
31	0,01	111110
32	0,01	1111110
33	0,01	1111111

Tabla 1.1.

Si se tiene en cuenta que ahora se asignan secuencias de bits a pares de símbolos de $\mathcal{X}$, no a elementos individuales, el número medio de bits por cada símbolo de $\mathcal{X}$ (cantidad que representábamos por L) será la mitad:

$$L = \frac{L_2}{2} = 1{,}365 < 1{,}5\,.$$

Pero también se podrían considerar longitudes de bloques mayores que dos; es decir, también se podrían considerar como alfabetos de la fuente $\mathcal{X}^3$, $\mathcal{X}^4$, etc., y se obtendrían cada vez menores valores de L. Y esto nos lleva a plantearnos nuevas cuestiones:

1. ¿Puede hacerse L tan pequeño como se quiera sin más que aumentar lo suficiente el orden de la extensión de la fuente, o, por el contrario, existe alguna cota inferior no nula para L? Y si tal cota existe, ¿cuál es su valor?

   Pues bien, se verá que no se puede hacer L, en general, arbitrariamente pequeña; sino que existe una cota inferior no nula para L, y que tal cota depende de las características de la fuente considerada. En realidad, dicha cota ha resultado ser una función de las probabilidades de los símbolos de la fuente que se denomina *entropía*, que es la magnitud fundamental de la Teoría de la Información. Y en el caso particular de la fuente que estamos considerando como ejemplo, la entropía de la fuente, y por tanto el valor de la cota inferior para L, es igual a 1,356 bits.

2. Finalmente, conocido que el valor de L no puede bajar de determinado umbral, parece lógico que nos preguntemos si tal umbral se puede alcanzar siempre; y en caso afirmativo, ¿cómo se alcanza dicho umbral?

   La respuesta es que, en algunos casos, es sencillo obtener un código con el valor óptimo de L; y que, en general, siempre se podrá obtener un código con un valor de L tan próximo como se quiera al óptimo sin más que considerar una extensión de la fuente lo suficientemente grande.

   Este resultado constituye uno de los dos teoremas fundamentales de codificación obtenidos por C. E. Shannon en [59], y es uno de los principales resultados de la Teoría de la Información; se lo conoce habitualmente como primer teorema de Shannon o como teorema de Shannon de codificación de fuente.

1.4. La codificación de canal

El problema de la codificación de canal es el de decidir si existe alguna forma de representar los mensajes de una fuente que permita una transmisión libre de errores aun en el caso de utilizar un canal ruidoso.

Antes de formalizar el estudio del problema en capítulos posteriores, en este apartado vamos a tratar de explicar con la mayor claridad posible las características del mismo utilizando algunos ejemplos sencillos de transmisión de mensajes binarios a través de un canal binario ruidoso.

Consideremos un canal binario simétrico sin memoria, es decir, un canal cuyos alfabetos de entrada y salida son binarios, y en el que la transmisión de cualquier símbolo tiene siempre la misma probabilidad de error, p ($p < 1/2$).[19]

Un canal tal está caracterizado por la siguiente matriz de probabilidades de transición:

$$Q_{2\times 2} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

o también mediante un diagrama como el de la figura 1.3, en el que se representan las transiciones mediante flechas, cada una con su probabilidad asociada.

[19] Si $p > 1/2$, parecería lógico considerar la transición de probabilidad p como la correcta, etiquetando el símbolo obtenido a la salida igual que el de la entrada, que es lo mismo que decir que el canal funciona más veces bien que mal. Si p fuese $1/2$, el canal sería totalmente inútil, ya que la probabilidad de obtener a la salida del canal un símbolo cualquiera sería $1/2$ con independencia del símbolo que se hubiese transmitido, de forma que el mensaje que saliese del canal sería siempre indistinguible de una secuencia aleatoria de bits en la que cada bit se obtiene con probabilidad $1/2$, como sería el caso, por ejemplo, de lanzar una moneda al aire para decidir el símbolo de la secuencia aleatoria.

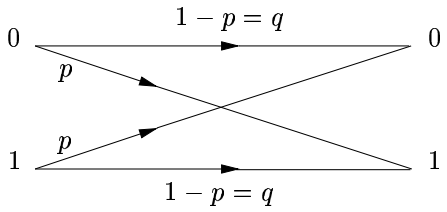


FIGURA 1.3. Canal binario simétrico sin memoria.

Consideremos también una fuente binaria sin memoria que genere ambos símbolos equiprobablemente; es decir, sea la fuente una variable aleatoria X de rango $\mathcal{X} = \{0, 1\}$ y medida de probabilidad $\mathrm{P}(x) = 1/2 \ \ \forall x \in \{0, 1\}$. Como se verá más adelante, en el capítulo 3, una fuente de este tipo es una fuente incompresible, es decir, una fuente que no requiere codificador de fuente.

Si se transmite sobre el canal dado un mensaje de la fuente de longitud n (es decir, una secuencia de n bits), se sabe que, siempre que n sea suficientemente grande, en virtud de la ley de los grandes números, se tendrá en destino otro mensaje de n bits que diferirá del generado por la fuente en aproximadamente pn bits; aunque no se sabe en qué bits, puesto que los errores se distribuyen aleatoriamente en el mensaje. Si se supiese qué bits son los erróneos, podría recuperarse el mensaje original, y los errores de transmisión no tendrían trascendencia.

Como no es así, nos enfrentamos a un problema: la utilización de un canal introduce errores en el mensaje transmitido, de suerte que no podemos estar seguros de que el mensaje recibido en destino sea igual al emitido por la fuente. Cabe entonces preguntarse si puede conseguirse la eliminación de los errores en la transmisión.[20] Por otra parte, parece claro que tal cosa sólo será posible si se restringen los mensajes que se envían por el canal (mensajes válidos) a algún subconjunto de todos los mensajes posibles, ya que resulta imposible detectar algún error si no se puede afirmar con seguridad que un cierto mensaje no puede haber sido enviado.

Una forma sencilla de hacer tal cosa consiste en fragmentar el mensaje de la fuente en secuencias de símbolos de longitud fija, digamos k símbolos, y asignar a cada una de tales secuencias una secuencia distinta de longitud n fija ($n > k$). Cuando se emplea un código tal para la transmisión de los mensajes de una fuente, sólo se transmiten por el canal 2^k mensajes (secuencias de n bits) distintos. Y, si como consecuencia de un error, se

[20] El criterio de fiabilidad que hemos elegido no exige la eliminación de todos los errores. Así, cuando en este texto se habla de transmisión sin errores, debe entenderse, en rigor, que se habla de una transmisión con una proporción de errores que se puede hacer tan pequeña como se quiera.

obtiene a la salida del canal una secuencia de n bits que no corresponde al código, entonces es infalible la detección del error.

Es obvio, por tanto, que la consecución de un sistema de transmisión fiable tiene un coste, puesto que es preciso transmitir más símbolos que los que genera la fuente. La proporción de símbolos adicionales ($\frac{n-k}{n} = 1 - \frac{k}{n}$) representa el coste (redundancia) del código. Por tanto, la ratio k/n, a la que denominaremos tasa de codificación, es una medida de la eficiencia de la codificación.

Veamos a continuación, mediante un sencillo ejemplo, cómo el empleo de redundancia permite una transmisión fiable, sin más que disponer de la cantidad de redundancia necesaria.

1.4.1. Códigos de repetición

Imagine que entre la fuente y el canal introducimos un dispositivo (codificador de canal) que reproduce por triplicado los símbolos que recibe, es decir, que realiza la siguiente transformación:

$\mathcal{X}$	$\longrightarrow$	$\mathcal{X}^3$
0	$\longrightarrow$	000
1	$\longrightarrow$	111

Por su parte, el decodificador de canal agrupará los símbolos que reciba del canal de tres en tres y les hará corresponder un único símbolo, que entregará al destino.

Al haber errores en la transmisión de los símbolos, el decodificador de canal podrá recibir cualquier terna. Sea $(x_1, x_2, x_3) \in \mathcal{X}^3$ una terna cualquiera de bits obtenida en recepción, ¿cómo debe realizarse la decodificación? Pueden haberse transmitido sólo dos cosas: o tres ceros o tres unos. Resulta claro que siempre podemos equivocarnos en la interpretación, sigamos el procedimiento que sigamos, porque incluso habiendo recibido tres ceros (análogamente tres unos) no puede haber certeza de que no se hayan transmitido tres unos (tres ceros). Así que parece lo más razonable tratar de minimizar la probabilidad de fallo al decidir, es decir, elegir aquel suceso que tenga mayor probabilidad.

Calculemos entonces la probabilidad de que se hayan transmitido tres ceros[21] a posteriori de saber que se ha recibido una terna cualquiera $x_1x_2x_3$

[21] Análogamente se hará para el caso de tres unos.

de bits, es decir, calculemos $\mathrm{P}(\text{transm.} = 000 \,|\, \text{rec.} = x_1x_2x_3)$:

$$\mathrm{P}\left(\text{transm.} = 000 \,|\, \text{rec.} = x_1x_2x_3\right) = \frac{\mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{transm.} = 000\right)\mathrm{P}(\text{transm.} = 000)}{\mathrm{P}(\text{rec.} = x_1x_2x_3)}\,.$$

Como

$$\begin{aligned}\mathrm{P}(\text{rec.} = x_1x_2x_3) = {}& \\ & \mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{transm.} = 000\right)\mathrm{P}(\text{transm.} = 000) + \\ & \mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{transm.} = 111\right)\mathrm{P}(\text{transm.} = 111)\end{aligned}$$

y puesto que $\mathrm{P}(\text{transm.} = 000) = \mathrm{P}(\text{transm.} = 111) = 1/2$, entonces

$$\mathrm{P}\left(\text{transm.} = 000 \,|\, \text{rec.} = x_1x_2x_3\right) = \frac{\mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{transm.} = 000\right)}{\mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{transm.} = 000\right) + \mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{transm.} = 111\right)}\,.$$

Ahora bien, como por la independencia de las sucesivas transmisiones sobre el canal se tiene que

$$\mathrm{P}\left(\text{rec.} = x_1x_2x_3 \,|\, \text{trasm.} = iii\right) = \prod_{j=1}^{3} \mathrm{P}\left(\text{rec.} = x_j \,|\, \text{transm.} = i\right)$$

y dado que se cumple:

$$\begin{aligned}\mathrm{P}(x_j \,|\, i) &= q \quad \text{si } x_j = i \\ \mathrm{P}(x_j \,|\, i) &= p \quad \text{si } x_j \neq i\end{aligned}$$

se obtienen los resultados que se recogen a continuación, en la tabla adjunta, para las probabilidades de que se hayan transmitido o bien tres ceros o bien tres unos, para cada una de las ternas que pueden llegar al decodificador de canal.

$\mathrm{P}\left(iii \,\|\, x_1x_2x_3\right)$		iii	
		000	111
$x_1x_2x_3$	000	$\frac{q^3}{q^3+p^3}$	$\frac{p^3}{q^3+p^3}$
	001	q	p
	010	q	p
	100	q	p
	011	p	q
	101	p	q
	110	p	q
	111	$\frac{p^3}{q^3+p^3}$	$\frac{q^3}{q^3+p^3}$

Así, por ejemplo, si se reciben tres ceros, no podemos estar seguros de que se hayan transmitido también tres ceros; podría haber ocurrido también que se hubieran transmitido tres unos, pero esto último es normalmente mucho menos probable (para $p = 0{,}1$, por ejemplo, la probabilidad de que se hubiesen transmitido tres unos sería $1{,}37 \times 10^{-3}$, mientras que se habrían transmitido tres ceros con probabilidad 0,9986; lo que significa que, suponiendo que se han transmitido tres ceros cada vez que se reciben tres ceros, nos equivocaremos —para un número suficientemente grande de decisiones— sólo un 0,137 % de las veces).

Aplicando esta regla de decisión al caso concreto que estamos considerando, resulta, a la vista de las probabilidades de la tabla, que debe suponerse que se ha transmitido un cero cuando haya una mayoría de ceros en la terna recibida; y debe suponerse que se ha transmitido un uno cuando haya una mayoría de unos. Es decir, debe suponerse que se ha enviado el símbolo que aparece más veces, por lo que esta regla de decisión se denomina de decodificación por mayoría.

Procediendo así, sólo habrá error en la transmisión de un símbolo de la fuente cuando se produzcan al menos dos errores en la transmisión de cada terna de símbolos por el canal. En consecuencia, la probabilidad de error en la transmisión de un símbolo vendrá dada por $\mathrm{P}_e^{(3)}$:

$$\begin{aligned}\mathrm{P}_e^{(3)} &= \mathrm{P}(2 \text{ errores}) + \mathrm{P}(3 \text{ errores}) \\ &= 3p^2q + p^3.\end{aligned}$$

Como $q = 1 - p$, entonces se tiene que

$$\begin{aligned}\mathrm{P}_e^{(3)} &= 3p^2 - 2p^3 \\ &= p^2(3 - 2p)\end{aligned}$$

y es inmediato comprobar que $\mathrm{P}_e^{(3)} < p$:

$$\mathrm{P}_e^{(3)} - p = -2p(p-1)(p-1/2) < 0 \quad \forall p \in (0, 1/2).$$

Así, el conjunto CODIFICADOR–CANAL–DECODIFICADOR es equivalente a un nuevo canal binario simétrico con probabilidad de error $\mathrm{P}_e^{(3)}$.

Puede comprobarse fácilmente que, al aumentar la redundancia, aumenta también, como cabía esperar, la reducción de la probabilidad de error en la transmisión de símbolos de la fuente. Así, con cinco repeticiones, resultaría un canal binario simétrico equivalente con una probabilidad de error $\mathrm{P}_e^{(5)}$:

$$\begin{aligned}\mathrm{P}_e^{(5)} &= \mathrm{P}(\text{número de errores} \geq 3) \\ &= \binom{5}{3} p^3 q^2 + \binom{5}{4} p^4 q + p^5.\end{aligned}$$

Como $q = 1 - p$, entonces se tiene que

$$\mathrm{P}_e^{(5)} = p^3(10 - 15p + 6p^2)$$

y es inmediato comprobar que $\mathrm{P}_e^{(5)} < \mathrm{P}_e^{(3)}$:

$$\mathrm{P}_e^{(5)} - \mathrm{P}_e^{(3)} = 6p^2(p-1)^2(p-1/2) < 0, \quad \forall p \in (0, 1/2).$$

En general, con $2n + 1$ repeticiones se tiene un canal binario simétrico equivalente con una probabilidad de error $\mathrm{P}_e^{(2n+1)}$:

$$\begin{aligned} \mathrm{P}_e^{(2n+1)} &= \mathrm{P}(\text{número de errores} \geq n+1) \\ &= \sum_{i=0}^{n} \binom{2n+1}{n+1+i} p^{n+1+i}(1-p)^{n-i}. \end{aligned}$$

Y se puede comprobar que $\lim_{n\to\infty} \mathrm{P}_e^{(2n+1)} = 0$. Basta para ello considerar que, como consecuencia de la ley (débil) de los grandes números, la proporción de bits erróneos de un mensaje converge en probabilidad a p ($p < 1/2$) según $n \to \infty$.

1.4.2. Existencia de códigos óptimos

Ha quedado patente, entonces, que siempre es posible conseguir una transmisión fiable; si bien, posiblemente, a costa del uso de una cantidad de redundancia no acotada, lo que significaría que la transmisión de cada símbolo de la fuente requeriría una cantidad ilimitada de tiempo. Pero tal solución es insatisfactoria, por lo que debe replantearse el problema en otros términos:

a) ¿Será posible conseguir una transmisión fiable con una redundancia acotada?

b) Si así fuese, ¿cuál sería la redundancia necesaria en cada caso?

c) Y por último, ¿cómo habría de generarse dicha redundancia?

Pues bien, la Teoría de la Información responde a las dos primeras preguntas:

a) Sí que es posible conseguir una transmisión fiable con una cantidad acotada de redundancia.

b) La cantidad de redundancia necesaria depende de las características del canal, y viene dada por un parámetro característico del mismo denominado *capacidad* del canal.

Pero todavía no se conoce una respuesta plenamente satisfactoria para la última cuestión. Existen, no obstante, múltiples esquemas de codificación de canal que, si no óptimos, sí se aproximan al límite de redundancia mínima establecido por Shannon. Se verá que, para conseguirlo sin que el proceso de decodificación resulte excesivamente complicado, es preciso emplear reglas de transformación que elijan representaciones de los mensajes de la fuente de forma que éstas posean una estructura matemática bien fundada.

Tales reglas de transformación pertenecen a uno de dos esquemas prototipo: en el más sencillo, ya conocido, se hace corresponder a cada secuencia distinta de símbolos de la fuente de una longitud fija una misma secuencia de símbolos de cierta longitud mayor, también fija. El nombre genérico que recibe este proceso es el de codificación de bloques. En el segundo esquema, cada grupo de símbolos de la fuente se completa igualmente con un cierto número (fijo) de símbolos de redundancia; pero éstos, ahora, no son siempre los mismos, sino que se hacen depender de los grupos de símbolos codificados con anterioridad, de forma que exista cierta relación conocida entre los grupos consecutivos de símbolos codificados. Por tanto, los códigos de esta segunda familia se caracterizan por precisar dispositivos codificadores y decodificadores con memoria.

Pero las clases de códigos de bloques (sin memoria) y de códigos con memoria son demasiado amplias como para permitir un análisis matemático cabal. Ante tal hecho, nos limitaremos a analizar, en ambos casos, transformaciones lineales entre los grupos de símbolos de entrada y de salida. Esta condición no supone ningún obstáculo para la búsqueda de códigos con buenas propiedades de control de errores, y, en cambio, da pie al desarrollo de un vasto campo de conceptos teóricos, procedentes del álgebra, la geometría y la teoría de números. Los códigos de bloques lineales se denominan, simplemente, *códigos lineales*, y a ellos dedicaremos la segunda parte del libro y varios capítulos de la tercera. Los códigos con memoria lineales se denominan *códigos convolucionales*, y se verá que, aun a pesar de exigir un proceso de decodificación con cierta complejidad, son equiparables o superiores en eficiencia a los códigos lineales.

1.5. Estructura del libro

Aparte de este capítulo introductorio, el libro contiene otros 11 capítulos, divididos en tres partes: la primera dedicada a la Teoría de la Información,

la segunda a los principios básicos de la corrección de errores, y la tercera al estudio más detallado de los principales códigos de control de errores.

La primera parte está dedicada a la exposición de los conceptos y resultados básicos de la Teoría de la Información, para el caso de fuentes y canales discretos. En el capítulo 2 se define el concepto de entropía y se analizan sus propiedades fundamentales. En el capítulo 3 se establece uno de los resultados fundamentales de la Teoría de la Información: el teorema de Shannon de codificación de fuente, que nos ofrece una interpretación física del concepto de entropía. Además, aunque de forma secundaria, se considera el problema de la construcción de códigos compactos. En el capítulo 4 se presentan una serie de funciones derivadas de la entropía, como son la entropía condicional, la información mutua y la capacidad del canal; y se analizan seguidamente sus propiedades más relevantes. Por último, en el capítulo 5 se establece el resultado probablemente más trascendental de la Teoría de la Información: el teorema de Shannon de codificación de canales ruidosos, que nos ofrece una interpretación física del concepto de capacidad.

La segunda parte se inicia con un capítulo, el 6, dedicado a exponer la teoría básica general de los códigos lineales. Fundamentalmente, la teoría consiste en una caracterización matricial de los códigos lineales y en el manejo de algunas propiedades métricas en un espacio de Hamming, ya sea para la síntesis de códigos concretos o para la deducción de un algoritmo de decodificación genérico. El capítulo 7 se centra en la teoría de los códigos cíclicos, el subconjunto de los lineales caracterizado por la particularidad de ser invariantes ante todas las permutaciones cíclicas. Esta propiedad confiere a la familia de los códigos cíclicos una estructura algebraica mucho más rica que la de espacio vectorial, lo que facilita la detección de grupos de errores y al tiempo permite ingeniar métodos de decodificación muy eficientes. La utilización de códigos de canal con potentes propiedades de corrección de errores no es, sin embargo, la única alternativa cuando de lo que se trata es de la transmisión fiable. En el capítulo 8 se describen una serie de estrategias basadas en la utilización conjunta de códigos detectores de errores y de retransmisiones automáticas de bloques de información. Allí se presta atención a dos aspectos de tales estrategias: por una parte, a garantizar la coherencia de las reglas lógicas con que operan los extremos emisor y receptor; y por otra, a calcular el régimen efectivo de transmisión de cada estrategia particular.

La parte tercera profundiza en la teoría de códigos algebraicos, presentando las tres familias de códigos de mayor importancia práctica hoy en día. Para cada una de estas familias, se dan sus propiedades y se describen los fundamentos de los métodos de decodificación algebraica. Puesto que se trata por lo común de códigos no binarios, el capítulo 9 establece previamente las propiedades fundamentales de los cuerpos finitos, sus representaciones y

su estructura. En el capítulo 10 se estudian los códigos BCH, una familia de códigos cíclicos polinómicos con excelentes propiedades de corrección de ráfagas de errores. La familia BCH con alfabeto no binario contiene a la clase de códigos Reed–Solomon, objeto del capítulo 11, cuyo interés reside en que constituyen un conjunto de códigos lineales de máxima distancia para el que se dispone de un algoritmo de decodificación computacionalmente eficiente. El capítulo 12, último, desplaza el centro de atención hacia los códigos convolucionales, que son códigos lineales con memoria. El capítulo se ocupa de dar, además de la caracterización clásica de éstos en el contexto de la teoría de sistemas discretos lineales, una descripción algebraica útil para la síntesis de los dispositivos de codificación y para la identificación de los codificadores catastróficos. Además, se explican en detalle los principios del algoritmo de Viterbi para la decodificación óptima de los códigos convolucionales.

PARTE I

Teoría de la Información

CAPÍTULO 2

La entropía

En este capítulo se propone una medida de la cantidad de información que contienen los mensajes de una fuente discreta. Tal medida se basa en la interpretación de la generación de un mensaje como el resultado de un experimento aleatorio, pudiéndose representar la cantidad de información del mismo como una función (logarítmica) de la inversa de la probabilidad de que tal resultado ocurra. Así, la información media de los mensajes dependerá de la distribución de probabilidades de los mismos, y será por tanto característica de la fuente que los genera. Esta medida de información recibe el nombre de *entropía* y constituye el pilar básico de toda la Teoría de la Información.

Como se verá en el capítulo siguiente, la entropía representa el número mínimo de símbolos necesarios para codificar sin pérdida de información los estados (mensajes) de la fuente, siempre que el número de éstos sea suficientemente grande (desde un punto de vista estadístico): a menor cantidad de información, mensajes más cortos. Este es uno de los resultados centrales de la Teoría de la Información y se lo conoce habitualmente como teorema de Shannon de codificación de fuente.

Este capítulo se dedicará por entero a la definición del concepto de entropía de una variable aleatoria discreta, así como al análisis de sus propiedades fundamentales.

2.1. Definición de entropía

Sea X una variable aleatoria discreta de rango finito $\mathcal{X}=\{x_1, x_2, \ldots, x_n\}$ que no incluya valores de probabilidad nula, y sea $p(x) = \mathrm{P}(X = x)$ su función de probabilidad.

Definición 2.1 (Entropía). *Se define la entropía de la variable aleatoria X como*

$$H(X) = \sum_{i=1}^{n} p(x_i) \log \frac{1}{p(x_i)}.$$

Para mayor claridad del concepto, así como de la notación asociada, conviene tener presentes las siguientes observaciones.

Observación 1. Dado que $\log \frac{1}{p(x_i)} = -\log p(x_i)$, la entropía de la variable aleatoria X se puede escribir también como

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i).$$

Observación 2. Por razones de generalidad, y dado que

$$\lim_{p(x_i)\to 0} p(x_i) \log p(x_i) = 0$$

se puede extender la definición anterior para incluir en el rango de la variable aleatoria X valores de probabilidad nula sin que varíe el valor de la entropía, adoptando el convenio de que

$$\forall p(x_i) = 0, \qquad p(x_i) \log p(x_i) = 0.$$

Observación 3. El valor numérico de la entropía depende de la base del logaritmo; pero en la definición anterior no se ha especificado, porque un cambio de base no supone más que un cambio de escala. Así pues, la adopción de una determinada base, b, no supone más que la elección de una cierta unidad de medida (que denominaremos unidad de base b); y la entropía expresada en unidades de base b se representará por $H_b(X)$. Para ciertos valores de la base b, las unidades suelen recibir denominaciones especiales:

base 2 → las unidades se denominan bits
base e (natural) → las unidades se denominan nats[1]
base 10 → las unidades se denominan hartleys[2]

[1] El origen del término parece ser la contracción de la expresión inglesa *natural units* (unidades naturales).

[2] En honor de R. V. L. Hartley, que fue el primero en establecer el principio de que la *información* es el resultado de una selección entre un número finito de posibilidades [31]. Hartley introdujo, además, la letra H para denotar la cantidad de información asociada con n selecciones

$$H = n \log s$$

donde s es el número de símbolos disponibles en cada selección.

La equivalencia entre las distintas unidades de medida se obtiene en virtud de la conocida fórmula del cambio de base del logaritmo:

$$\log_a x = \frac{\log_b x}{\log_b a}$$

de la que resulta que

$$H_a(X) = \frac{H_b(X)}{\log_b a}$$

o bien que

$$H_a(X) = H_b(X) \log_a b.$$

Así, por ejemplo, se tiene que

$$\begin{aligned} H_2(X) &= \sum_{i=1}^{n} p(x_i) \log_2 \frac{1}{p(x_i)} \\ &= \frac{1}{\ln 2} \sum_{i=1}^{n} p(x_i) \ln \frac{1}{p(x_i)} \\ &= \frac{1}{\ln 2} H_n(X). \end{aligned}$$

Observación 4. Obsérvese que la entropía no depende de los valores que pueda tomar la variable aleatoria X, sino sólo de sus probabilidades. Así pues, la entropía es en realidad función de una cierta distribución[3] de probabilidades $\mathbf{p} = (p_1, p_2, \ldots, p_n)$, por lo que también se suele representar como $H(\mathbf{p})$. Y en el caso de una distribución de probabilidad binaria, $\mathbf{p} = (p, 1-p)$, la entropía es función únicamente de la variable p, y se representa simplemente por $H(p)$.

Observación 5. Puede considerarse $p_X(x)$ como una función de la variable aleatoria X que a cada valor $x \in X$ le hace corresponder su probabilidad $p_X(x)$. Para explicitar la dependencia de la variable aleatoria X, representamos tal función como $p(X)$. Por su definición, esta nueva variable aleatoria tiene la peculiaridad de que los distintos elementos de su rango y sus probabilidades respectivas coinciden numéricamente; pero es, por lo demás, una variable aleatoria absolutamente igual que las demás.

De manera análoga, se puede definir otra función de la variable aleatoria X:

$$I(X) = \log \frac{1}{p(X)}$$

[3] Aun más, en realidad del conjunto de las probabilidades de una distribución (función de masa de probabilidad): una distribución de probabilidad implica un cierto secuenciamiento de las probabilidades, que tampoco influye en el valor de la entropía.

cuyo valor medio será:

$$\mathrm{E}\left[I(X)\right] = \sum_x p(x) \log \frac{1}{p(x)} = H(X).$$

Según este resultado, la entropía viene a ser el valor medio de una variable aleatoria, $I(X)$, que es una variable aleatoria discreta cuyos valores, $I(x_i) = \log \frac{1}{p(x_i)}$, son inversamente proporcionales a las probabilidades de los mismos, $p(x_i)$. Por tanto, si se interpreta $I(x_i)$ como la información que se obtiene cuando ocurre x_i (los sucesos menos probables aportan más información), entonces $H(X)$ es la información que proporcionan, en media, los distintos resultados posibles de la variable aleatoria X. O lo que es lo mismo, es la información que proporciona, en media, la fuente X por cada símbolo transmitido. Más adelante, en este mismo capítulo, se abundará en esta interpretación de la entropía como medida de información.

Observación 6. Tal y como se ha definido la entropía, ésta no tiene en cuenta el régimen de transmisión de la fuente. Ahora bien, si se multiplica la entropía de la fuente, $H(X)$, por su régimen de transmisión, v_f, se tiene una función que representa la información que transmite la fuente, en media, por unidad de tiempo:

$$H^t(X) = v_f H(X).$$

Ejemplo 2.1. Si X es una variable aleatoria con distribución de probabilidad $\mathbf{p} = (1/2, 1/4, 1/8, 1/8)$, su entropía en bits valdrá:

$$\begin{aligned} H_2(X) &= -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{4}\log_2\frac{1}{4} - \frac{1}{8}\log_2\frac{1}{8} - \frac{1}{8}\log_2\frac{1}{8} \\ &= \frac{7}{4} \text{ bits.} \end{aligned}$$

■

Ejemplo 2.2. La entropía de la variable aleatoria binaria

$$X = \begin{cases} 1 & \text{con probabilidad } p, \\ 0 & \text{con probabilidad } 1-p \end{cases}$$

vale:

$$H(X) = -p\log p - (1-p)\log(1-p) = H(p).$$

En particular, $H_2(X) = 1$ bit cuando $p = 1/2$; es decir, un bit es la cantidad de entropía obtenida al elegir uno de entre dos símbolos equiprobables. La gráfica de la figura 2.1 representa la función $H_2(p)$ e ilustra alguna de sus propiedades: $H(p)$ es una función con mínimo en $p = 0$ o $p = 1$, cuando la variable X es determinista, y máximo cuando $p = 1/2$; $H(p)$ es una función estrictamente convexa. ■

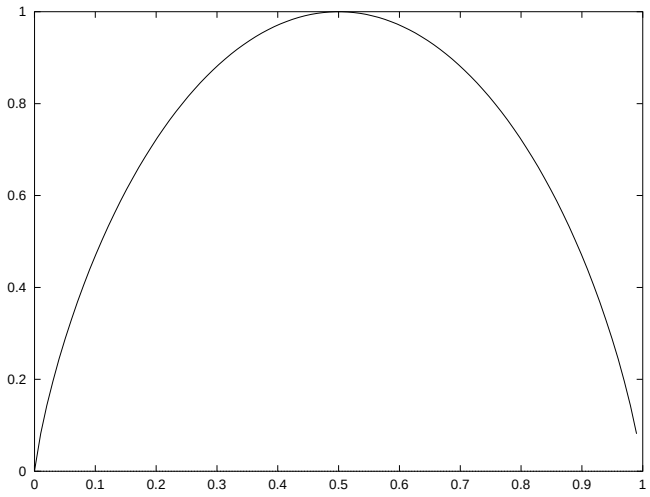


Figura 2.1. Gráfica de $H_k(p)/\log_k 2$.

2.2. Funciones convexas

Algunas de las más notables propiedades de la entropía son consecuencia directa de la convexidad de la función logaritmo; y se verá que aun la propia entropía es también una función convexa. Por tal motivo, detengámonos brevemente para repasar la noción de convexidad.

Sea $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ un vector n-dimensional de componentes reales definido en una región $\mathcal{R}$ del espacio vectorial.

Definición 2.2 (Región convexa). *Se dice que una región $\mathcal{R}$ es convexa si $\forall \mathbf{p}, \mathbf{q} \in \mathcal{R}$ se cumple que*

$$\alpha\mathbf{p} + (1-\alpha)\mathbf{q} \in \mathcal{R}, \ \ \forall\, \alpha \in \mathbb{R}, \ \ 0 < \alpha < 1.$$

Geométricamente, al variar α entre 0 y 1, $\alpha\mathbf{p} + (1-\alpha)\mathbf{q}$ traza una línea recta de $\mathbf{p}$ a $\mathbf{q}$:

$$\alpha\mathbf{p} + (1-\alpha)\mathbf{q} = \mathbf{q} + \alpha(\mathbf{p} - \mathbf{q}), \ \ \forall\, \alpha \in \mathbb{R}, \ \ 0 < \alpha < 1.$$

Así, una región es convexa si, para cada par de puntos de la región, la línea recta que une dichos puntos pertenece por entero a la región (es decir, todos los puntos de la línea recta pertenecen a la región).

Ejemplo 2.3. Es, por tanto, una región convexa cualquier intervalo (a, b) de puntos de la recta real $\mathbb{R}$: resulta fácil verificar que para cualesquiera

dos puntos $x_1, x_2 \in (a,b)$ se cumple que

$$\alpha x_1 + (1-\alpha)x_2 \in (a,b), \quad \alpha \in \mathbb{R}, \quad 0 < \alpha < 1. \qquad \blacksquare$$

Ejemplo 2.4. Otro ejemplo de región convexa de interés para nosotros es la región de vectores de probabilidad, denominando vectores de probabilidad a todos aquellos vectores cuyas componentes sean todas no negativas y sumen uno. Para comprobar que es así, basta comprobar que todos los puntos de la recta que une dos vectores de probabilidad cualesquiera, $\mathbf{p}$ y $\mathbf{q}$:

$$\mathbf{v} = \alpha\mathbf{p} + (1-\alpha)\mathbf{q}, \quad \alpha \in \mathbb{R}, \quad 0 < \alpha < 1$$

son también vectores de probabilidad; es decir, que $\forall \alpha \in \mathbb{R}, \quad 0 < \alpha < 1$, se cumplen las dos condiciones siguientes:

$$v_i \geq 0, \qquad \sum_{i=1}^{n} v_i = 1$$

cuya verificación es trivial ya que simplemente se tiene que

$$\begin{aligned} v_i &= \alpha p_i + (1-\alpha)q_i \\ \sum_{i=1}^{n} v_i &= \alpha \sum_{i=1}^{n} p_i + (1-\alpha) \sum_{i=1}^{n} q_i. \end{aligned} \qquad \blacksquare$$

Definición 2.3 (Función convexa). *Se dice que una función real f definida en una región convexa $\mathcal{R}$ de un espacio vectorial es una función convexa si para cualesquiera dos puntos $\mathbf{p}, \mathbf{q} \in \mathcal{R}$ y $\forall\, \alpha \in \mathbb{R}, \quad 0 < \alpha < 1$, se cumple que*

$$f\big(\alpha\mathbf{p} + (1-\alpha)\mathbf{q}\big) \geq \alpha f(\mathbf{p}) + (1-\alpha) f(\mathbf{q}). \tag{2.1}$$

Geométricamente, la interpretación de la ecuación (2.1) es que el arco de cuerda que une los puntos $f(\mathbf{p})$ y $f(\mathbf{q})$ está siempre por debajo de la superficie $f(\alpha\mathbf{p} + (1-\alpha)\mathbf{q})$, lo que aparece ilustrado en la figura 2.2 para una función convexa de variable real definida en cualquier intervalo de la recta real (a,b) que contenga a los puntos x_1 y x_2.

Ahora bien, si se considera la curva simétrica respecto al eje de abscisas, es decir, si se considera la gráfica de la función $-f(x)$, se tiene justo lo contrario: que el arco de cuerda está por encima de la curva. Puede decirse que las funciones convexas tienen dos caras; y para distinguirlas es habitual emplear los términos convexidad $\cap$ y convexidad $\cup$, o también convexidad y concavidad.[4]

[4]En este punto, y para evitar confusiones con la terminología empleada en otros textos,

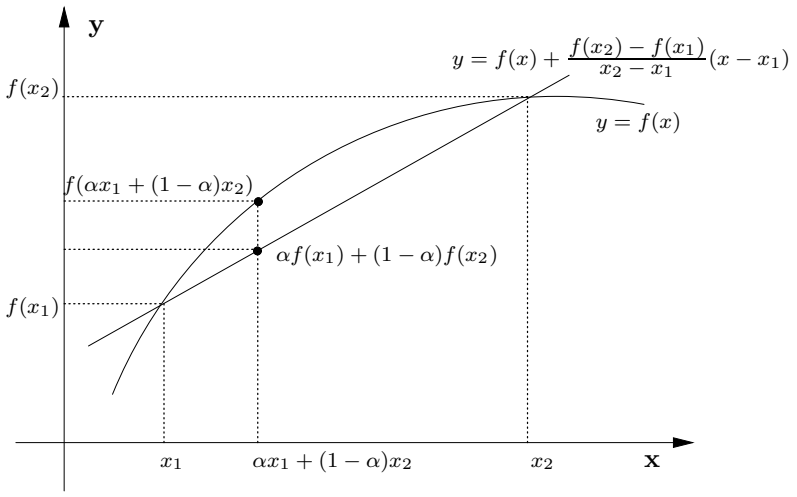


Figura 2.2. Convexidad de $f(x)$.

Definición 2.4 (Función estrictamente convexa). *Se dice que una función real f definida en una región convexa $\mathcal{R}$ de un espacio vectorial es estrictamente convexa si para cualesquiera dos puntos $\mathbf{p}, \mathbf{q} \in \mathcal{R}$ y $\forall\ \alpha \in \mathbb{R},\ 0 < \alpha < 1$, se cumple que*

$$f\big(\alpha\mathbf{p} + (1-\alpha)\mathbf{q}\big) > \alpha f(\mathbf{p}) + (1-\alpha) f(\mathbf{q}).$$

2.2.1. Algunas propiedades de las funciones convexas

Por su utilidad en el desarrollo del tema, se enuncian a continuación algunas propiedades bien conocidas de las funciones convexas.

Teorema 2.1. *Si $f_1(\mathbf{p}), \ldots, f_K(\mathbf{p})$ son funciones convexas, y $\alpha_1, \ldots, \alpha_K$ son números reales positivos, entonces*

$$\sum_{i=1}^{K} \alpha_i f_i(\mathbf{p})$$

es también convexa; siendo estrictamente convexa si cualquiera de las $f_i(\mathbf{p})$ es estrictamente convexa.

---

es preciso decir que cuando se emplean los términos concavidad y convexidad, es frecuente en la literatura matemática denominar función cóncava a la que aquí hemos llamado función convexa, y viceversa.

DEMOSTRACIÓN. La propiedad se deduce de forma inmediata aplicando la definición que se acaba de dar de función convexa a $\sum_{i=1}^{K} \alpha_i f_i(\mathbf{p})$. ▶

TEOREMA 2.2 (CRITERIO DE LA SEGUNDA DERIVADA). *Siendo f una función de variable real que admite derivadas de segundo orden, $f(x)$ es convexa en un intervalo abierto (a, b) si y sólo si*

$$\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2} \leq 0$$

en ese intervalo; siendo estrictamente convexa si la desigualdad es estricta.

DEMOSTRACIÓN. Supongamos primero que $\mathrm{d}^2 f(x)/\mathrm{d}x^2$ es no positiva en (a, b). Entonces $\mathrm{d}f(x)/\mathrm{d}x$ es no creciente en (a, b). Así, $\forall a < x < y < b$, $0 < \alpha < 1$, $z = (1-\alpha)x + \alpha y$, se tiene:

$$f(z) - f(x) = \int_x^z f'(t)\,\mathrm{d}t \geq f'(z)(z - x)$$
$$f(y) - f(z) = \int_z^y f'(t)\,\mathrm{d}t \leq f'(z)(y - z).$$

Puesto que $z - x = \alpha(y - x)$ e $y - z = (1-\alpha)(y - x)$, se tiene:

$$f(z) \geq f(x) + \alpha f'(z)(y - x)$$
$$f(z) \geq f(y) - (1-\alpha) f'(z)(y - x).$$

Multiplicando las dos desigualdades por $1 - \alpha$ y α, respectivamente, y sumándolas, se obtiene que

$$(1-\alpha) f(z) + \alpha f(z) \geq (1-\alpha) f(x) + \alpha f(y).$$

Pero el lado izquierdo de la desigualdad es precisamente $f(z) = f((1-\alpha)x + \alpha y)$, lo que prueba la convexidad de f en (a, b) de acuerdo con la definición.

Supongamos ahora que $\mathrm{d}^2 f(x)/\mathrm{d}x^2$ es positiva en un cierto subintervalo (a', b'). Argumentando de la misma forma que acabamos de hacer, en (a', b') tendríamos:

$$f(z) - f(x) < f'(z)(z - x)$$
$$f(y) - f(z) > f'(z)(y - z)$$

y de aquí que

$$f\big((1-\alpha)x + \alpha y\big) < (1-\alpha) f(x) + \alpha f(y).$$

Así, f no sería convexa en (a, b). ▶

Esta propiedad es bien conocida porque nos proporciona un criterio sencillo para la determinación de la convexidad de una función. Así, resulta fácil verificar que la función logaritmo es una función convexa.

Ejemplo 2.5 (Convexidad de la función logaritmo). Considérese la función $f(x) = \ln(x)$, que está definida $\forall x \in \mathbb{R},\ \mathrm{x} > 0$. En todo su dominio de definición, la función admite derivadas de primer y segundo orden:

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \frac{1}{x}; \qquad \frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2} = -\frac{1}{x^2}.$$

Como

$$\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2} < 0 \qquad \forall x \in \mathbb{R},\ \mathrm{x} > 0,$$

entonces $f(x)$ es una función estrictamente convexa en todo su dominio de definición. ■

A continuación se establece una desigualdad ampliamente utilizada en matemáticas, y de gran interés para nosotros porque en ella se basan muchos de los resultados básicos de la Teoría de la Información.

Teorema 2.3 (Desigualdad de Jensen). *Si f es una función convexa y X es una variable aleatoria discreta, entonces*

$$\mathrm{E}\left[f(X)\right] \leq f\left(\mathrm{E}[X]\right).$$

Demostración. Apliquemos el método de inducción. Para una variable aleatoria binaria X, la desigualdad de Jensen no es más que la definición de convexidad:

$$p(x_1)f(x_1) + p(x_2)f(x_2) \leq f\big(p(x_1)x_1 + p(x_2)x_2\big).$$

Supóngase ahora que la desigualdad es cierta para distribuciones de probabilidad con $k-1$ valores, es decir, que se cumple que

$$\sum_{i=1}^{k-1} p(x_i)f(x_i) \leq f\Big(\sum_{i=1}^{k-1} p(x_i)x_i\Big).$$

Probemos que también se cumple para una distribución de probabilidad con

k valores:

$$\begin{aligned}\sum_{i=1}^{k} p(x_i)f(x_i) &= p(x_k)f(x_k) + \sum_{i=1}^{k-1} p(x_i)f(x_i) \\ &= p(x_k)f(x_k) + \big(1-p(x_k)\big)\sum_{i=1}^{k-1} q(x_i)f(x_i) \\ &\leq p(x_k)f(x_k) + \big(1-p(x_k)\big)f\Big(\sum_{i=1}^{k-1} q(x_i)x_i\Big) \\ &\leq f\Big(p(x_k)x_k + \big(1-p(x_k)\big)\sum_{i=1}^{k-1} q(x_i)x_i\Big) \\ &= f\Big(\sum_{i=1}^{k} x_i p(x_i)\Big)\end{aligned}$$

donde la segunda igualdad se sigue del cambio $q(x_i) = \frac{p(x_i)}{1-p(x_k)}$, de forma que las $q(x_i)$ forman una distribución de probabilidad; la primera desigualdad se sigue de la hipótesis de inducción; la segunda desigualdad se sigue de la convexidad de la función f; y la última igualdad, de deshacer el cambio en las $q(x_i)$. ▶

Utilizando la desigualdad de Jensen, puede obtenerse otra notable desigualdad, también de amplio alcance en el campo de la Teoría de la Información, conocida habitualmente como desigualdad de Gibbs.

Teorema 2.4 (Desigualdad de Gibbs). *Sean dos distribuciones de probabilidad de n elementos:*

$$\begin{aligned}\mathbf{p} &= (p_1, p_2, \ldots, p_n) \\ \mathbf{q} &= (q_1, q_2, \ldots, q_n).\end{aligned}$$

Se cumple la siguiente desigualdad:

$$\sum_{i=1}^{n} p_i \log \frac{1}{p_i} \leq \sum_{i=1}^{n} p_i \log \frac{1}{q_i}.$$

Asimismo, se cumple la igualdad si y sólo si las dos distribuciones de probabilidad son idénticas.

Demostración. Demostremos en primer lugar que se satisface la desigualdad:

$$\begin{aligned}\sum_{i=1}^{n} p_i \log \frac{1}{p_i} - \sum_{i=1}^{n} p_i \log \frac{1}{q_i} = \sum_{i=1}^{n} p_i \log \frac{q_i}{p_i} \\ &\leq \log \sum_{i=1}^{n} p_i \frac{q_i}{p_i} \\ &= \log \sum_{i=1}^{n} q_i \\ &= \log 1 \\ &= 0\end{aligned}$$

siguiéndose la desigualdad de la demostración de la desigualdad de Jensen.

Consideremos ahora las condiciones para que se verifique la igualdad. La suficiencia de la condición es evidente. Demostremos, pues, que la condición es también necesaria.

Como la base del logaritmo es irrelevante para la demostración, tomaremos logaritmos neperianos por razones de conveniencia. Por ser la función logaritmo una función convexa, la gráfica de la función logaritmo no está nunca por encima de la tangente a la misma en cualquier punto; en particular, no está por encima de la tangente a la curva en el punto $x = 1$, que para el logaritmo neperiano será la recta $x - 1$. Por otra parte, al ser la función logaritmo estrictamente convexa, sólo coincidirá con la recta tangente en el punto de tangencia, es decir, el punto de abscisa 1; en este caso:

$$\begin{cases}\ln x = 0; & x = 1 \\ \ln x < x - 1; & x \neq 1 \ (x > 0)\end{cases}$$

En virtud de lo anterior, se cumple que

$$\ln \frac{q_i}{p_i} < \frac{q_i}{p_i} - 1 \Leftrightarrow p_i \neq q_i.$$

Por tanto, si y sólo si $\exists\ i$ tal que $p_i \neq q_i$, se cumple que

$$\sum_{i=1}^{n} p_i \ln \frac{q_i}{p_i} < \sum_{i=1}^{n} p_i \left(\frac{q_i}{p_i} - 1 \right) = \sum_{i=1}^{n} q_i - \sum_{i=1}^{n} p_i = 0$$

ya que $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i = 1$; con lo que queda demostrado que

$$\sum_{i=1}^{n} p_i \log \frac{q_i}{p_i} = 0 \Longrightarrow p_i = q_i \ \forall\ i. \qquad \blacktriangleright$$

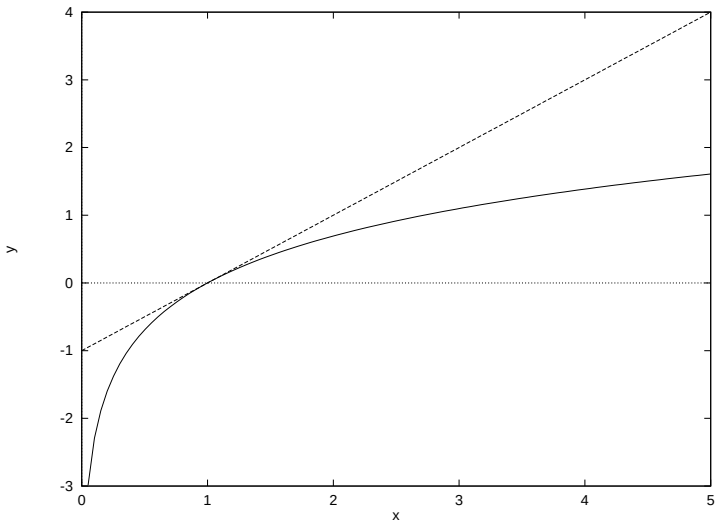


FIGURA 2.3. Gráficas de $\ln x$ y de $x - 1$.

2.3. Propiedades de la entropía

De acuerdo con la definición de entropía, se trata de una función definida en una región convexa (la región de vectores de probabilidad), continua en todo el dominio de definición de la función. Se verá ahora, además, que tal función es también convexa en todo el dominio de definición y que está acotada tanto superior como inferiormente.

La cota inferior es el cero, valor que se alcanza solamente cuando el vector de probabilidades tiene una única componente no nula, es decir, cuando corresponde a un fenómeno en el que no hay aleatoriedad (fenómeno determinista). Esto significa, de entrada, que la entropía permite distinguir entre aleatoriedad (entropía no nula) y determinismo (entropía nula).

Por otra parte, la cota superior de la entropía depende del número de componentes no nulas del vector de probabilidad, siendo una función estrictamente creciente de dicho número. Y tal cota superior se alcanza solamente para una distribución de probabilidad uniforme. Es decir, de entre todas las distribuciones de probabilidad con un número de elementos de probabilidad no nula dados, la uniforme es la que tiene un mayor valor de la entropía. Si se tiene en cuenta que una distribución uniforme es aquélla en la que la probabilidad de acertar lo que va a ocurrir, siempre que se adopte una hipótesis razonable, es mínima, puede decirse que una distribución uniforme representa la incertidumbre máxima, por lo que también el máximo de la entropía se corresponde con el máximo de la incertidumbre.

Habida cuenta, además, de que $H(\mathbf{p})$ es una función continua de $\mathbf{p}$, y que, como se verá más adelante, tal función es convexa, puede pensarse en utilizar la entropía como una medida de la incertidumbre de un fenómeno aleatorio. En este sentido, si se tiene en cuenta que la incertidumbre acerca de un experimento o fenómeno aleatorio desaparece una vez conocido el resultado del mismo, puede decirse que la realización del experimento o la observación del fenómeno nos proporciona información, y en la misma medida, además, en que antes había incertidumbre. Por este motivo, puede considerarse la entropía una medida de la cantidad de información que proporciona un experimento o fenómeno. Obviamente, esta medida no es única; pero sí que es la única que satisface una propiedad adicional, que denominaremos *propiedad de partición*, según la cual la adquisición del conocimiento por partes no requiere más información.[5]

2.3.1. Propiedad de partición

Teorema 2.5. *Sea X una variable aleatoria discreta de rango $\mathcal{X}$. Consideremos una partición de $\mathcal{X}$ dada por dos subconjuntos del mismo, $\mathcal{A}_1$ y $\mathcal{A}_2$; es decir, sean $\mathcal{A}_1$ y $\mathcal{A}_2$ dos subconjuntos disjuntos de $\mathcal{X}$ tales que $\mathcal{A}_1 \cup \mathcal{A}_2 = \mathcal{X}$. Entonces se cumple que*

$$H(X) = H(p) + pH(A_1) + (1-p)H(A_2)$$

siendo $p = \mathrm{P}(\mathcal{A}_1)$ y siendo $H(A_i)$ la incertidumbre remanente si se sabe que el valor que toma la variable aleatoria X pertenece a $\mathcal{A}_i$.

Demostración. Sean

$$q(x_i) = \mathrm{P}(X = x_i \mid x_i \in \mathcal{A}_1) = \frac{p(x_i)}{p}$$
$$r(x_i) = \mathrm{P}(X = x_i \mid x_i \in \mathcal{A}_2) = \frac{p(x_i)}{1-p}$$

las funciones de masas de probabilidad de los elementos de $\mathcal{A}_1$ y $\mathcal{A}_2$, respectivamente.

[5] En todo caso, no es claro que la entropía sea una medida de la información más adecuada por el hecho de satisfacer la propiedad de partición; de hecho, la entropía se usa porque una cantidad de interés, como la longitud mínima de los mensajes de una fuente para una comunicación fiable, se determina a través del concepto de entropía.

$$
\begin{aligned}
H(X) &= \sum_{i=1}^{n} p(x_i) \log \frac{1}{p(x_i)} = \sum_{x_i \in \mathcal{A}_1} p(x_i) \log \frac{1}{p(x_i)} + \sum_{x_i \in \mathcal{A}_2} p(x_i) \log \frac{1}{p(x_i)} \\
&= p \sum_{x_i \in \mathcal{A}_1} q(x_i) \log \frac{1}{pq(x_i)} + (1-p) \sum_{x_i \in \mathcal{A}_2} r(x_i) \log \frac{1}{(1-p)r(x_i)} \\
&= p \left(\sum_{x_i \in \mathcal{A}_1} q(x_i) \log \frac{1}{q(x_i)} + \sum_{x_i \in \mathcal{A}_1} q(x_i) \log \frac{1}{p} \right) \\
&+ (1-p) \left(\sum_{x_i \in \mathcal{A}_2} r(x_i) \log \frac{1}{r(x_i)} + \sum_{x_i \in \mathcal{A}_2} r(x_i) \log \frac{1}{1-p} \right).
\end{aligned}
$$

Teniendo en cuenta que

$$
\begin{aligned}
\sum_{x_i \in \mathcal{A}_1} q(x_i) \log \frac{1}{q(x_i)} &= H(A_1) \\
\sum_{x_i \in \mathcal{A}_2} r(x_i) \log \frac{1}{r(x_i)} &= H(A_2)
\end{aligned}
$$

y que

$$
\begin{aligned}
\sum_{x_i \in \mathcal{A}_1} q(x_i) \log \frac{1}{p} &= \log \frac{1}{p} \sum_{x_i \in \mathcal{A}_1} q(x_i) = \log \frac{1}{p} \\
\sum_{x_i \in \mathcal{A}_2} r(x_i) \log \frac{1}{1-p} &= \log \frac{1}{1-p}
\end{aligned}
$$

entonces resulta que

$$
\begin{aligned}
H(X) &= pH(A_1) + (1-p)H(A_2) + p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p} \\
&= pH(A_1) + (1-p)H(A_2) + H(p).
\end{aligned}
$$

▶

Esta propiedad nos dice que un experimento aleatorio que se pueda descomponer en varios otros más elementales tendrá una entropía igual a la suma de las entropías de todos los experimentos elementales.

Ejemplo 2.6. Supongamos que tenemos en una urna ocho bolas numeradas del 1 al 8. Supongamos además que las bolas impares están dentro de una caja etiquetada con un 1; y las pares, dentro de una caja etiquetada con un 2. Imaginemos en primer lugar un experimento E_1 consistente en extraer al azar una bola de la urna (sin reparar en qué caja estaba) y observar cuál es su número. Representemos por X la variable aleatoria que caracteriza

dicho experimento. El rango de X contiene ocho valores equiprobables, por lo que la información que nos suministra la realización del experimento E_1 es $H(X) = \log 8 = 3$ bits.

Imaginemos ahora que el experimento anterior se realiza en dos pasos consecutivos. Representemos tal experimento por $F_1 \times F_2$, siendo F_1 el primer paso del experimento, consistente en elegir al azar una de las cajas de bolas de la urna y observar cuál es el número de su etiqueta, y siendo F_2 el segundo paso del experimento, consistente en extraer finalmente una bola de la caja elegida en el primer paso del experimento y observar cuál es su número. El conocimiento que se obtiene al final de este nuevo experimento es exactamente el mismo que el que se obtiene con el experimento E_1.

Representemos por Y_1 la variable aleatoria que caracteriza el experimento F_1. El rango de Y_1 contiene dos valores equiprobables, por lo que la información que nos suministra la realización del experimento F_1 es $H(Y_1) = \log 2 = 1$ bit.

El resultado del experimento F_2 estará condicionado, obviamente, por el resultado de F_1. Representemos por Y_2^1 la variable aleatoria que caracteriza el experimento F_2 cuando en el primer paso se ha elegido la caja etiquetada con el 1. El rango de Y_2^1 contiene cuatro valores equiprobables, por lo que la información que nos suministra la realización del segundo paso del experimento es, en este caso, $H(Y_2^1) = \log 4 = 2$ bits. Representemos por Y_2^2 la variable aleatoria que caracteriza el experimento F_2 cuando en el primer paso se ha elegido la caja etiquetada con el 2. El rango de Y_2^2 contiene igualmente cuatro valores equiprobables, por lo que la información que nos suministra la realización del segundo paso del experimento es, también en este caso, $H(Y_2^2) = \log 4 = 2$ bits. Finalmente, si se denomina $H(Y_2)$ al promedio de las entropías del segundo experimento, se tiene que

$$H(Y_2) = H(Y_2^1)\,\mathrm{P}(Y_1 = 1) + H(Y_2^2)\,\mathrm{P}(Y_1 = 2) = 2 \text{ bits}$$

y por tanto, efectivamente, se cumple que

$$H(X) = H(Y_1) + H(Y_2). \qquad \blacksquare$$

2.3.2. Definición axiomática de la entropía

Se puede demostrar que la entropía es la única función de una distribución de probabilidad, $H(p_1, p_2, \ldots, p_n)$, que cumple las siguientes propiedades:

1. $H(p_1, p_2, \ldots, p_n)$ es una función continua de los p_i;
2. si todos los p_i son iguales, $H(p_1, p_2, \ldots, p_n)$ es una función monótona creciente de n;

3. $H(p_1, p_2, \ldots, p_n)$ cumple la propiedad de partición anterior;

motivo por el que las tres condiciones expuestas pueden ser utilizadas para una introducción axiomática del concepto de entropía.

TEOREMA 2.6. *La única función $H(p_1, p_2, \ldots, p_n)$ que satisface las tres condiciones anteriores es de la forma*

$$H = -K \sum_{i=1}^{n} p_i \log p_i.$$

DEMOSTRACIÓN. Como ya se sabe que la función anterior satisface las tres propiedades fijadas, lo que habrá que probar es que no puede satisfacer dichas propiedades ninguna otra función.

Como, por la condición 2, $H(p_1, p_2, \ldots, p_n)$ debe ser una función de n cuando todos los p_i son iguales, sea $A(n) = H(1/n, 1/n, \ldots, 1/n)$. Vamos a demostrar en primer lugar que, por las condiciones 2 y 3, debe ser $A(n) = K \log n$.

Supongamos que la elección aleatoria de un elemento de entre un conjunto de m^s elementos equiprobables, siendo m y s números enteros positivos cualesquiera, y cuya entropía por tanto es $A(m^s)$, se descompone en s elecciones sucesivas de forma que en cada una de ellas se elija entre m valores equiprobables. Por la condición 3, se cumple que:

$$\begin{aligned} A(m^s) &= A(m) + A(m^{s-1}) \\ A(m^{s-1}) &= A(m) + A(m^{s-2}) \\ &\vdots \\ A(m^2) &= A(m) + A(m) \end{aligned}$$

de donde se sigue que $A(m^s) = sA(m)$.

De manera análoga, $A(n^t) = tA(n)$. Podemos elegir t arbitrariamente grande y encontrar un s que satisfaga

$$m^s \leq n^t \leq m^{s+1}.$$

Así, tomando logaritmos y dividiendo por $t \log m$, se tiene que

$$\frac{s}{t} \leq \frac{\log n}{\log m} \leq \frac{s}{t} + \frac{1}{t}$$

por lo que la diferencia entre s/t y $\log n / \log m$ puede hacerse arbitrariamente pequeña:

$$\left| \frac{s}{t} - \frac{\log n}{\log m} \right| < \epsilon$$

donde ϵ es una cantidad arbitrariamente pequeña.

Por otra parte, por ser $A(n)$ una función monótona creciente de n:

$$\begin{aligned} &A(m^s) \leq A(n^t) \leq A(m^{s+1}) \\ &sA(m) \leq tA(n) \leq (s+1)A(m) \\ &\frac{s}{t} \leq \frac{A(n)}{A(m)} \leq \frac{s}{t} + \frac{1}{t} \end{aligned}$$

por lo que la diferencia entre s/t y $A(n)/A(m)$ puede hacerse arbitrariamente pequeña como antes:

$$\left| \frac{s}{t} - \frac{A(n)}{A(m)} \right| < \epsilon.$$

De todo lo anterior se concluye que

$$\left| \frac{A(n)}{A(m)} - \frac{\log n}{\log m} \right| < 2\epsilon$$

y por lo tanto, necesariamente, $A(n) = K \log n$, donde K debe ser una constante positiva para que se satisfaga la condición 2.

Consideremos ahora el caso más general de que las n posibilidades de elección arbitraria no sean equiprobables; pero supongamos que tales probabilidades son números racionales. En tal caso, dichas probabilidades serán de la forma $p_i = \dfrac{n_i}{\sum n_i}$ donde los n_i son enteros positivos.

La elección aleatoria de un elemento de entre $\sum n_i$ elementos equiprobables tendrá, según se ha visto hasta ahora, una entropía $K \log \sum n_i$. Pero si se descompusiese dicha elección en dos pasos, de forma que en el primer paso se escogiese entre n posibles conjuntos con $n_1, n_2, \ldots, n_n$ elementos cada uno (es decir, de probabilidades $p_1, p_2, \ldots, p_n$), y a continuación se eligiese un elemento del conjunto, al ser todas las elecciones equiprobables, por la condición 3 se tendría que

$$K \log \sum n_i = H(p_1, p_2, \ldots, p_n) + K \sum p_i \log n_i.$$

De la anterior igualdad, y teniendo en cuenta que $\sum p_i = 1$, se sigue que

$$\begin{aligned} H(p_1, p_2, \ldots, p_n) &= K \left(\sum p_i \log \sum n_i - \sum p_i \log n_i \right) \\ &= -K \sum p_i \log \frac{n_i}{\sum n_i} \\ &= -K \sum p_i \log p_i. \end{aligned}$$

Para finalizar, sólo hay que demostrar que la misma igualdad se obtendría en el caso de que los p_i fuesen irracionales, lo que se sigue de la continuidad de los p_i (condición 1), ya que existe un número racional que se aproxima tanto como se quiera a un número irracional cualquiera. ▶

2.3.3. Convexidad de la entropía

TEOREMA 2.7 (CONVEXIDAD DE LA ENTROPÍA). *Sea* $\mathbf{p} = (p_1, \ldots, p_n)$ *una distribución de probabilidades;* $H(\mathbf{p})$ *es una función convexa de* $\mathbf{p}$.

DEMOSTRACIÓN. Dado que por definición

$$H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log p_i$$

y dado que, por el teorema 2.2 (criterio de la segunda derivada), $-p_i \log p_i$ es una función convexa de p_i; entonces, en virtud del teorema 2.1, se tiene que $H(\mathbf{p})$ es convexa. ▶

COROLARIO 2.8. *En particular, para distribuciones de probabilidad binarias, dadas dos distribuciones de probabilidad* (p, q) *y* (p', q') *tales que* $|p' - q'| \leq |p - q|$, *se tiene que* $H(p) \leq H(p')$.

DEMOSTRACIÓN. Sean $\mathbf{p}_1 = (p, q)$ y $\mathbf{p}_2 = (q, p)$; y sea $\mathbf{p}' = (p', q')$, con $\mathbf{p}' = \alpha\mathbf{p}_1 + (1 - \alpha)\mathbf{p}_2, 0 < \alpha < 1$. Entonces, en virtud del teorema anterior:

$$H(\mathbf{p}') \geq \alpha H(\mathbf{p}_1) + (1 - \alpha)H(\mathbf{p}_2)$$
$$H(p') \geq \alpha H(p) + (1 - \alpha)H(p) = H(p).$$ ▶

2.3.4. Acotación de la entropía

Los teoremas que se establecen a continuación enuncian que la entropía de una variable aleatoria discreta tiene una cota inferior (el cero) y una cota superior (el logaritmo del cardinal del rango de la variable aleatoria), y nos dan además las condiciones para alcanzar tales cotas.

TEOREMA 2.9. *El valor de la entropía es o positivo o nulo; y es condición necesaria y suficiente para que* $H(X) = 0$ *que la distribución de probabilidades de la variable aleatoria* X *tenga un único elemento no nulo.*

DEMOSTRACIÓN. Es obvio que el valor de la entropía es o positivo o nulo, ya que, de acuerdo con su definición, la entropía es una suma de términos no negativos.

Es obvio también que si sólo hay un $p(x_i) \neq 0$ éste debe valer 1; y resulta entonces que $H(X) = 1 \cdot \log 1 = 0$.

Por otra parte, dado que la entropía es, por definición, una suma de términos no negativos, tal suma sólo valdrá cero cuando todos los términos sean nulos. De lo que se deduce que, necesariamente, habrá de cumplirse siempre que $p(x_i) = 0$ o que $p(x_i) = 1 \ \ \forall\, i$; y como todos los $p(x_i)$ deben sumar necesariamente uno, entonces existe un valor y sólo uno de i para el que $p(x_i) = 1$. ▶

Teorema 2.10. *Siendo n el número de elementos de la distribución de probabilidades de la variable aleatoria X, se verifica que*

$$H(X) \leq \log n.$$

Es además condición necesaria y suficiente para que $H(X) = \log n$ que la distribución de probabilidades de la variable aleatoria X sea uniforme, es decir, que $p(x_i) = \frac{1}{n} \ \ \forall\, i$.

Demostración. En virtud de la desigualdad de Gibbs se tiene que

$$\begin{aligned} H(X) &= \sum_{i=1}^{n} p(x_i) \log \frac{1}{p(x_i)} \\ &\leq \sum_{i=1}^{n} p(x_i) \log \frac{1}{1/n} \\ &= \log n \end{aligned}$$

cumpliéndose la igualdad si y sólo si $p(x_i) = 1/n \quad \forall i$. ▶

2.4. Entropía conjunta

En la sección anterior hemos definido la entropía de una variable aleatoria X. Se da el caso de que, a veces, interesa considerar fenómenos en los que aparecen dos variables aleatorias. Sea una de ellas la anterior variable aleatoria X; y sea Y otra variable aleatoria de rango discreto y finito $\mathcal{Y} = \{y_1, y_2, \ldots, y_m\}$, cuya función de masa de probabilidad es $p_Y(y) = \mathrm{P}(Y = y)$.

Consideradas conjuntamente las dos variables aleatorias, X e Y, tenemos una nueva variable aleatoria: la variable aleatoria bidimensional (X, Y), cuyo rango, también discreto y finito, será

$$\mathcal{X} \times \mathcal{Y} = \{(x_i, y_i) : \ x_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$$

y cuya función de masa de probabilidad será $p(x, y) = \mathrm{P}(X = x, Y = y)$.

Como, a la hora de calcular la entropía, lo único relevante son los valores de las probabilidades, pero no los valores que puede tomar la variable aleatoria, entonces no hay ninguna dificultad para extender la definición de entropía a variables aleatorias bidimensionales (o, en general, n-dimensionales). Así, la entropía de la variable aleatoria bidimensional (X, Y) vendrá dada por

$$H(X, Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) \log \frac{1}{p(x_i, y_j)}.$$

Definición 2.5 (Entropía conjunta). *Se define la entropía conjunta de las variables aleatorias X e Y como la entropía de la variable aleatoria bidimensional (X, Y).*

Por supuesto, la definición anterior puede extenderse sin ninguna dificultad para cualquier número arbitrario de variables aleatorias discretas.

2.4.1. Relación entre las entropías individuales y la conjunta

Vamos a probar a continuación que la entropía conjunta de dos variables aleatorias no puede superar a la suma de las entropías de dichas variables aleatorias consideradas por separado. Y, más aún, que es condición necesaria y suficiente para que la entropía conjunta valga lo mismo que la suma de las entropías individuales que ambas variables aleatorias sean independientes entre sí.

Teorema 2.11. *Se verifica en general que*

$$H(X, Y) \leq H(X) + H(Y)$$

siendo condición necesaria y suficiente que las variables aleatorias X e Y sean independientes entre sí para que se cumpla que

$$H(X, Y) = H(X) + H(Y).$$

Demostración. Dado que

$$p(x_i) = \sum_j p(x_i, y_j), \qquad p(y_j) = \sum_i p(x_i, y_j)$$

entonces:

$$H(X) = \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(x_i)}$$
$$H(Y) = \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(y_j)}.$$

Por tanto,

$$\begin{aligned} H(X) + H(Y) &= \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(x_i)p(y_j)} \\ &\geq \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(x_i, y_j)} \\ &= H(X, Y) \end{aligned}$$

donde la desigualdad se sigue de la desigualdad de Gibbs; por lo que se tiene, además, que se cumplirá la igualdad si y sólo si se verifica que $p(x_i, y_j) = p(x_i)p(y_j) \ \ \forall (x_i, y_j)$. ▶

Resulta fácil verificar que esta propiedad es extensible para cualquier número de variables aleatorias discretas.

Notas bibliográficas

El concepto de entropía se debe a Shannon [59], así como también el análisis de sus propiedades fundamentales. El término entropía lo introdujo Clausius en 1876 en el campo de la Termodinámica, donde este concepto sirve para enunciar la segunda ley de la Termodinámica. Shannon utilizó este término por la semejanza de $H(X)$ con la expresión de la entropía introducida por Boltzmann en 1896 en la Mecánica Estadística. La relación entre la entropía de la Teoría de la Información y la segunda ley de la Termodinámica se explora brevemente en [15].

Shannon extendió asimismo en [60] los conceptos de este capítulo al caso, matemáticamente un poco más complejo, de variables y vectores aleatorios continuos. Puede consultarse también en prácticamente cualquier libro de Teoría de la Información, siendo especialmente recomendables [26, 45, 15].

CAPÍTULO 3

Codificación de fuente

En este capítulo se va a analizar el problema de la codificación de fuente, entendido éste como el de decidir la forma más conveniente de representar los mensajes de la fuente para la consecución de una transmisión fiel y eficiente de los mismos si se ignoran los errores que se producen en el medio de transmisión, tal y como ya se expuso en el primer capítulo del libro.

Dada una fuente discreta con alfabeto finito $\mathcal{F}$ y dado un alfabeto de codificación también finito $\mathcal{D}$, se considerarán únicamente los códigos consistentes en asignar unívocamente a cada elemento de $\mathcal{F}^n$, para un valor de n dado, una secuencia de elementos de $\mathcal{D}$, que denominaremos *palabra del código*. Los códigos de este tipo reciben habitualmente la denominación de *códigos de bloques*.[1] La característica más notable de los códigos de bloques es que asignan siempre a cada secuencia de símbolos de longitud n de la fuente la misma secuencia de elementos del alfabeto de codificación, con independencia de lo que haya ocurrido con anterioridad, por lo que estos códigos también se denominan *códigos sin memoria*.

Ahora bien, no todos los códigos de bloques son igualmente convenientes para nuestro propósito. En primer lugar, el requisito de la transmisión fiel de los mensajes impone una restricción en la elección del código: los códigos de bloques válidos reciben el nombre de códigos unívocamente decodificables. Son códigos unívocamente decodificables todos aquéllos con los que siempre es posible recuperar sin error el mensaje original (el mensaje de la fuente) a partir de su versión codificada.[2]

[1] A veces se utiliza esta denominación con una acepción más restringida, para referirse únicamente a aquellos códigos de este tipo en que todas las secuencias de elementos de $\mathcal{D}$ consideradas tienen también la misma longitud.

[2] Esto no quiere decir que los códigos que no son unívocamente decodificables sean inútiles para la comunicación. Bien al contrario, existen algunos servicios de comunicación

Aun más, dentro de los códigos unívocamente decodificables hay un subconjunto especialmente conveniente: el de los códigos instantáneos. A definir tanto los códigos unívocamente decodificables, en general, como los códigos instantáneos, en particular, y a analizar su necesidad o conveniencia, dedicaremos el siguiente apartado de este capítulo.

Se denomina longitud de una palabra del código al número de símbolos de $\mathcal{D}$ que la forman. Pues bien, la síntesis de códigos instantáneos con palabras de longitud constante es bastante elemental. Pero, a veces, es más conveniente utilizar palabras de diferentes longitudes para la síntesis de un código. Así, en el apartado 3.2 se analiza la cuestión de la síntesis de códigos instantáneos con palabras de diferentes longitudes, y se define una medida de la eficiencia de los mismos.

Seguidamente, se establece el teorema de Shannon de codificación de fuente (si bien sólo para fuentes discretas sin memoria), resultado central de este capítulo, que establece la posibilidad (asintótica) de construir siempre códigos óptimos (códigos de eficiencia unidad).

Dado que la posibilidad de construir códigos óptimos es, en general, sólo asintótica, en el siguiente apartado se considera el problema de la asignación óptima de palabras a un conjunto dado de símbolos. Tales códigos, que no son necesariamente óptimos (en el sentido apuntado en el párrafo anterior), se denominan comúnmente *códigos compactos*. Pues bien, se verá que existe un método sistemático para la obtención de un código compacto: el algoritmo de Huffman.

Finalmente, en los dos últimos apartados del capítulo se extiende el teorema de Shannon para considerar modelos más generales de fuentes.

3.1. Códigos de bloques: códigos unívocamente decodificables y códigos instantáneos

Sea $\mathcal{F} = \{0, 1, \ldots, q-1\}$ el alfabeto de una fuente discreta dada, y sea $\mathcal{F}^n$ el producto cartesiano formado por n conjuntos $\mathcal{F}$, es decir, el conjunto formado por todas las n-tuplas de elementos de $\mathcal{F}$. Sea $\mathcal{D} = \{0, 1, \ldots, D-1\}$ el alfabeto de codificación.

que requieren del empleo de estos códigos. Esto ocurre cuando existe una restricción para la longitud del mensaje codificado que no satisface ningún código unívocamente decodificable. Por supuesto, en tales casos, la decodificación introducirá una distorsión en el mensaje original; y se tiene entonces otro problema, que aunque relacionado con el nuestro no consideraremos en este texto: el de la minimización de la distorsión del mensaje original para una restricción de longitud de codificación dada.

Definición 3.1 (Código de bloques). *Se llama código de bloques a aquél que a cada elemento de $\mathcal{F}^n$, para un valor de n cualquiera dado, le hace corresponder unívocamente una secuencia fija de elementos de $\mathcal{D}$.*

Así pues, matemáticamente, un código de bloques es una aplicación

$$f : \mathcal{F}^n \longrightarrow \bigcup_{i=1,2,\ldots} \mathcal{D}^i$$

siendo $\mathcal{D}^i$ el producto cartesiano formado por i conjuntos $\mathcal{D}$.

De acuerdo con la definición anterior, dos aplicaciones distintas con el mismo dominio de la aplicación y la misma imagen serían dos códigos de bloques distintos. Pero a veces únicamente es de nuestro interés el conjunto de secuencias de símbolos de $\mathcal{D}$ que se utilizan para la codificación, es decir, la imagen de la aplicación; y en tales casos también se suele denominar código a tal conjunto, recibiendo sus elementos la denominación de *palabras del código*.

Precisamente, únicamente del conjunto de palabras del código depende que sea posible siempre recuperar el mensaje original a partir de su versión codificada, es decir, que sea posible la transmisión fiable.

Efectivamente, resulta fácil deducir que es condición necesaria para la fiabilidad de la transmisión que la aplicación f sea inyectiva,[3] es decir, que todas las palabras del código sean distintas. Los códigos resultantes cuando la aplicación f es inyectiva se denominan *códigos no singulares*. Se puede establecer, por tanto, la siguiente definición.

Definición 3.2 (Código no singular). *Un código de bloques se denomina no singular cuando todas las palabras del código son distintas.*

Resulta fácil verificar, además, que si todas las palabras del código son de longitud constante, la condición de no singularidad del código es condición suficiente para la transmisión fiel.[4] Pero si las palabras del código son de distintas longitudes, la condición de no singularidad no es suficiente para la recuperación fidedigna del mensaje en el receptor; ilustrémoslo con un ejemplo.

Ejemplo 3.1. Sean $\mathcal{D} = \{0, 1\}$ y $\mathcal{F} = \{a, b, c, d\}$, y considérese el siguiente código de bloques no singular:

[3] Una función f se dice inyectiva si siempre ocurre que elementos distintos del conjunto de partida tienen imágenes distintas en el conjunto de llegada, $x \neq y \Rightarrow f(x) \neq f(y)$.

[4] Se está suponiendo implícitamente que hay tantas palabras en el código como símbolos tiene el alfabeto de la fuente.

$$a \to 0$$
$$b \to 01$$
$$c \to 001$$
$$d \to 0001$$

Cuando el decodificador se encuentra con la siguiente secuencia

$$001001$$

¿qué debe interpretar?, es decir, ¿cómo debe descomponer la secuencia de bits recibida en una secuencia de palabras del código? Obsérvese que existen múltiples posibilidades:

$$\begin{array}{lcl} & 0-01-0-01 & (abab) \\ \text{o bien} & 001-001 & (cc) \\ \text{o bien} & 0-01-001 & (abc) \end{array}$$

Como puede verse, en este caso la decodificación no es unívoca, y así la transmisión no puede ser fiable. ■

Pueden distinguirse, en consecuencia, dos grandes grupos de códigos: aquéllos en que todas las palabras del código constan de un mismo número de elementos de $\mathcal{D}$ (códigos de longitud constante) y aquéllos en que no (códigos de longitud variable).

Los primeros conducen a codificadores y decodificadores más sencillos; pero no siempre minimizan la representación de los mensajes de la fuente, pues es bien conocido que, en algunos casos, cuando las probabilidades no se reparten uniformemente, puede obtenerse una codificación más compacta de los mensajes de la fuente si se asignan palabras del código muy cortas a los símbolos más probables, aunque eso implique que haya que asignar palabras del código largas a los símbolos menos probables (como es el caso del código Morse).

Entonces, si las palabras del código son de longitud variable, no basta para la transmisión fiel con que sean distintas, sino que han de ser tales que mensajes distintos tengan siempre secuencias distintas de elementos del alfabeto de codificación de forma que no quepa más que una interpretación para cada mensaje recibido. Se dice que tales códigos son *códigos unívocamente decodificables* o de decodificación unívoca. Formalicemos matemáticamente esta restricción, introduciendo previamente el concepto de extensión de un código.

Definición 3.3 (Extensión de orden n de un código de bloques). *Considérese un código de bloques cualquiera de un cierto alfabeto discreto*

$\mathcal{F}$, y sea σ_i la palabra del código correspondiente a un símbolo $i \in \mathcal{F}$. La extensión de orden n del código de bloques anterior es el código que resulta de asignar a cada símbolo $(i_1, i_2, \ldots, i_n) \in \mathcal{F}^n$ la secuencia $(\sigma_{i_1}, \sigma_{i_2}, \ldots, \sigma_{i_n})$ de palabras del código original.

Es claro, de acuerdo con la definición anterior, que la extensión de orden n de un código de bloques es también un código de bloques. Y también que cualquier mensaje digital codificado, por largo que sea, puede considerarse como una única palabra del correspondiente código de bloques extendido.

Definición 3.4 (Código unívocamente decodificable). *Un código de bloques se dice unívocamente decodificable si y sólo si su extensión de orden n es no singular para cualquier valor finito de n.*

Restringiremos nuestra atención, por tanto, a los códigos unívocamente decodificables, ya que son los únicos en los que no hay posibilidad de ambigüedad en la descomposición de una secuencia de elementos de $\mathcal{D}$ en palabras del código. Y dentro de los códigos unívocamente decodificables centraremos nuestro interés en el subconjunto formado por aquellos códigos en los que no sólo se identifica sin ambigüedad cada palabra del código, sino que se hace tal identificación nada más recibir el último símbolo que la integra. Tales códigos, que reciben por tal motivo la denominación de códigos instantáneos, permiten simplificar el proceso de decodificación sin sacrificar a cambio eficiencia de codificación, tal y como se verá en el siguiente apartado (teoremas de Kraft y de McMillan).

Definición 3.5 (Código instantáneo). *Un código de bloques unívocamente decodificable se denomina instantáneo cuando es posible decodificar las palabras de una secuencia sin precisar el conocimiento de los símbolos que las suceden.*

Ejemplo 3.2. Un claro ejemplo de código unívocamente decodificable y no instantáneo es el siguiente:

$$\begin{aligned} a &\to 0 \\ b &\to 01 \\ c &\to 011 \\ d &\to 0111 \end{aligned}$$

Basta ver, para cerciorarse de que no puede haber ambigüedad en la decodificación, que todas las palabras del código empiezan por un único cero, diferenciándose unas de otras por el número de unos que siguen al cero: ninguno (a), uno (b), dos (c) o tres (d). Así pues, la estrategia de decodificación

es simple: después de recibir un cero, se cuenta el número de unos hasta el siguiente cero. Queda claro, además, que sólo se identifica cada una de las palabras del código al recibir el siguiente cero, es decir, el primer símbolo de la siguiente palabra del código, y por tanto este código no es instantáneo; lo que se puede deducir también simplemente de un examen de las palabras del código, ya que la primera palabra del código es prefijo de todas las demás, la segunda es prefijo de la tercera y la cuarta, y la tercera es prefijo de la cuarta. ■

EJEMPLO 3.3. Un claro ejemplo de código instantáneo es el siguiente:

$$a \to 0$$
$$b \to 10$$
$$c \to 110$$
$$d \to 1110$$

Basta ver, para cerciorarse de que no puede haber ambigüedad en la decodificación, que todas las palabras del código terminan con un único cero, diferenciándose unas de otras por el número de unos que preceden al cero: ninguno (a), uno (b), dos (c) o tres (d). Así pues, la estrategia de decodificación es simple: basta contar el número de unos hasta el siguiente cero. Y ahora, además, se identifican todas las palabras del código al recibir el cero, es decir, el último símbolo de la palabra del código. Esto se deduce también, como hemos dicho, del hecho de que ninguna palabra del código es prefijo de otra. ■

EJEMPLO 3.4. Una variante del código instantáneo anterior, más eficiente, es el código siguiente:

$$a \to 0$$
$$b \to 10$$
$$c \to 110$$
$$d \to 111$$

Obsérvese que no es necesario que todas las palabras del código finalicen con un cero. ¿Sabría cuál es ahora el proceso de decodificación? ■

A la vista de los ejemplos anteriores, es claro, además, que se cumple el siguiente teorema.

TEOREMA 3.1. *Es condición necesaria y suficiente para que un código sea instantáneo que ninguna palabra del código sea prefijo de otra, es decir, que ninguna palabra del código se forme añadiendo símbolos al final de otra palabra del código.*

En virtud de este teorema, por tanto, se puede verificar por simple inspección si un código dado es instantáneo (y por tanto unívocamente decodificable) o no. Obsérvese que el hecho de que un código no sea instantáneo no implica necesariamente que el código no sea unívocamente decodificable.

La figura 3.1 representa por medio de diagramas de Venn la relación entre los distintos tipos de códigos considerados.

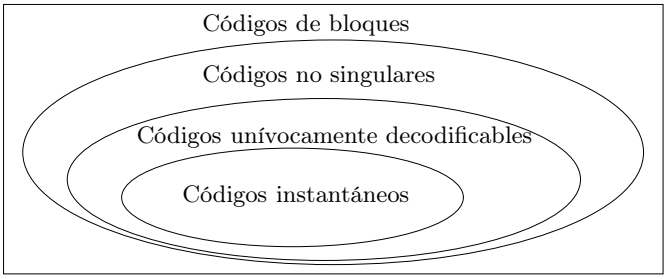


Figura 3.1.

A nosotros nos interesa, pues, en este libro la síntesis del código instantáneo más conveniente para representar los mensajes de una fuente dada. Y será mas conveniente aquél para el que sea menor el número de símbolos del alfabeto de codificación necesarios (en media) para representar cada símbolo de la fuente. Tal parámetro se denomina longitud del código.

Definición 3.6 (Longitud del código). *Sea $\mathcal{F}$ el alfabeto de la fuente considerada y sea $\mathcal{D}$ el alfabeto de codificación. Consideremos un código de bloques general dado por*

$$f : \mathcal{F}^n \longrightarrow \bigcup_{i=1,2,\ldots} \mathcal{D}^i$$

Sea L_n el número medio de símbolos de $\mathcal{D}$ que tienen los elementos de la imagen de f. Se define la longitud del código como

$$L = \frac{L_n}{n}\,.$$

Naturalmente, el valor de L_n, y por tanto el de L, depende no sólo de los elementos de la imagen de f, sino también de su probabilidad, que depende de la probabilidad del símbolo de $\mathcal{F}^n$ al que representa:

$$L_n = \sum_i p_i l_i$$

siendo l_i la longitud de la palabra del código que representa al símbolo $i \in \mathcal{F}^n$ y siendo $p_i = \mathrm{P}(X = i)$ la probabilidad de dicho símbolo.

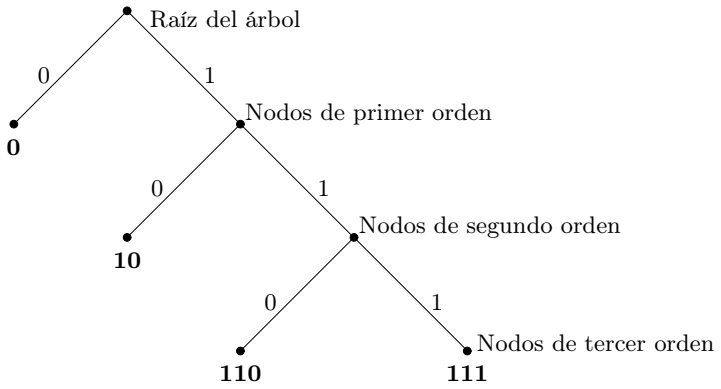


FIGURA 3.2.

3.2. Síntesis de códigos instantáneos con palabras de longitud variable

Una representación gráfica de las palabras de un código instantáneo puede obtenerse simplemente representando cada palabra del código por un nodo terminal de un árbol. A título de ejemplo, se muestra en la figura 3.2 el árbol que representa las palabras del código del ejemplo 3.4. Los dígitos sucesivos de cada palabra del código pueden entenderse como instrucciones para subir[5] desde la raíz del árbol hasta el nodo terminal que representa la letra de la fuente deseada.

Se define un árbol completo de orden n y alfabeto de tamaño D como un árbol con D nodos de orden i provenientes de cada nodo de orden $i-1$ para cada $1 \leq i \leq n$; es decir, un árbol completo de orden n es aquél en el que todas las ramas se extienden por igual hasta los nodos de orden n.

Obsérvese que, en un árbol completo, una fracción D^{-1} de los nodos de cada orden $i \geq 1$ provienen de cada uno de los D nodos de orden 1. De igual forma, una fracción D^{-2} de los nodos de cada orden $i \geq 2$ provienen de cada uno de los D^2 nodos de orden 2; y una fracción D^{-i} de los nodos de cada orden mayor o igual que i provienen de cada uno de los D^i nodos de orden i.

Así, un código instantáneo cuya palabra del código más larga tenga longitud $n_{\max}$ se corresponde con un subárbol de un árbol completo de orden $n_{\max}$. Y todo árbol con q ramas se corresponde con un código instantáneo de cualquier alfabeto fuente de q elementos. Habrá, naturalmente, múltiples

[5] En el caso de la figura 3.2, para bajar, dado que el árbol se ha representado invertido.

posibles códigos instantáneos, de los que nos interesará saber cuál es el de longitud mínima.

Dejando de lado momentáneamente la búsqueda de la mejor manera de construir códigos instantáneos, lo que sí parece claro es que la elección de un valor pequeño para unas l_i hace que otras l_i tengan que superar un cierto umbral. El teorema de Kraft formaliza este hecho.

Teorema 3.2 (de Kraft). *Es condición necesaria para que un código sea instantáneo que se verifique*

$$\sum_{i=0}^{q-1} D^{-l_i} \leq 1 \tag{3.1}$$

siendo l_i la longitud de la palabra del código correspondiente al símbolo $i \in \mathcal{F}$. Además, dicha condición es suficiente para construir un código instantáneo con palabras del código de longitudes l_i.

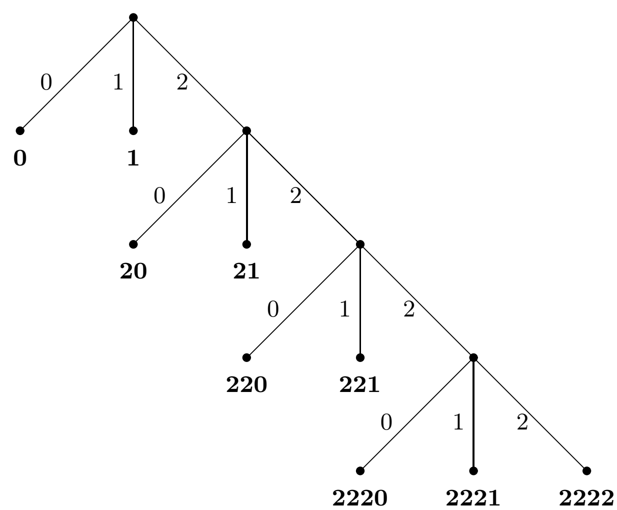
Demostración. Sean $l_0, \ldots, l_{q-1}$ tales que satisfagan la ecuación (3.1), y supongamos, para simplificar la notación, que $l_0 \leq l_1 \leq \cdots \leq l_{q-1}$.

Veamos primero cómo construir un código instantáneo con estas longitudes. Considérese el árbol completo de orden $n = l_{q-1}$ y elíjase un nodo cualquiera de orden l_0, llamémosle x_0; tenemos disponibles aún todos los nodos de orden mayor o igual que l_0 excepto los que provienen del nodo x_0, cuyo número será como mucho la fracción D^{-l_0} del total de nodos terminales del árbol completo. A continuación se selecciona cualquier nodo disponible de orden l_1, llamémosle x_1; después de esto permanecen todavía disponibles todos los nodos de orden mayor o igual que l_1 del árbol completo a excepción de los que provienen de x_0 o x_1, cuyo número será como mucho la fracción $D^{-l_0} + D^{-l_1}$ del total de nodos terminales del árbol completo. Continuando de esta manera, después de la asignación del j-ésimo nodo quedarán disponibles todos los nodos del árbol completo de orden igual o mayor que l_j a excepción de los que provienen de los nodos x_0 a x_j, cuyo número será como mucho la fracción $\sum_{i=0}^{j} D^{-l_i}$ del total de nodos del árbol completo. Por la desigualdad (3.1), esta fracción es siempre estrictamente menor que 1 para $j < q-1$, y por tanto siempre hay algún nodo disponible para ser asignado como siguiente nodo terminal.

Para probar la segunda parte del teorema, nótese que el árbol correspondiente a cualquier código instantáneo puede insertarse en un árbol completo cuyo orden sea igual a la longitud mayor de las palabras del código. Un nodo terminal de orden l_i en el árbol del código dará lugar a una fracción D^{-l_i} de los nodos terminales del árbol completo. Puesto que los conjuntos de nodos terminales del árbol completo provenientes de diferentes nodos terminales

del árbol del código son disjuntos, estas fracciones sumarán como mucho 1, obteniéndose la desigualdad (3.1). ▶

Ejemplo 3.5. Sea $D = 3$ y sean las longitudes $(1, 1, 2, 2, 3, 3, 4, 4, 4)$. Entonces, $\sum_i 3^{-l_i} = 1$. Y un código instantáneo cuyas longitudes sean las dadas es el que se representa en el siguiente árbol:



■

Como consecuencia inmediata de este teorema se puede demostrar una importante desigualdad.

Corolario 3.3. *La longitud de un código instantáneo, L, cumple que*

$$L \geq H_D(X).$$

Demostración. Para un código de bloques con $n = 1$ se tiene:

$$\begin{aligned}
H_D(X) - L_1 &= \sum_{i=0}^{q-1} p_i \log_D \frac{1}{p_i} - \sum_{i=0}^{q-1} p_i l_i \\
&= \sum_{i=0}^{q-1} p_i \log_D \frac{D^{-l_i}}{p_i} \\
&\leq \log \sum_{i=0}^{q-1} p_i \frac{D^{-l_i}}{p_i} \\
&\leq 0
\end{aligned}$$

verificándose la primera desigualdad en virtud de la desigualdad de Jensen y la segunda en virtud del teorema de Kraft.

Análogamente, para un valor arbitrario de n se tiene que

$$L_n \geq H_D(X^n)$$

siendo X^n la variable aleatoria que representa las n-tuplas de símbolos de la fuente.

Pero como, por otra parte, se sabe que $L_n = nL$; y que $H_D(X^n) = nH_D(X)$, por ser las componentes de X^n independientes e idénticamente distribuidas; entonces se obtiene el resultado del corolario. ▶

El resultado anterior nos proporciona una interesante interpretación del concepto de entropía:

> *La entropía de una fuente representa el número mínimo de símbolos del alfabeto de codificación que son necesarios (en media) para representar cada símbolo de la fuente.*

Podríamos preguntarnos ahora si no existirá algún código unívocamente decodificable no instantáneo para el que $L < H_D(X)$. Pues bien, el teorema de McMillan responde negativamente a esta pregunta.

Teorema 3.4 (de McMillan). *Es condición necesaria para que un código sea unívocamente decodificable que se cumpla*

$$\sum_{i=0}^{q-1} D^{-l_i} \leq 1.$$

Demostración. Sea $\mathcal{C}^k$ la k-ésima extensión del código unívocamente decodificable $\mathcal{C}$, es decir, el código formado por todas las secuencias que se pueden formar concatenando k palabras del código $\mathcal{C}$. Por definición, $\mathcal{C}^k$ es un código no singular.

Sean $\sigma_0, \sigma_1, \ldots, \sigma_{q-1}$ las palabras del código $\mathcal{C}$, y sean $l_0, l_1, \ldots, l_{q-1}$ sus respectivas longitudes.

Sea $\sigma_i^k \in \mathcal{C}^k$ una secuencia de k palabras del código $\mathcal{C}$ cuya longitud (igual a la suma de las longitudes de las k palabras del código C que la forman) representaremos por $|\sigma_i^k|$. Entonces se verifica que

$$\left(\sum_{i=0}^{q-1} D^{-l_i}\right)^k = \sum_{\sigma_i^k \in \mathcal{C}^k} D^{-|\sigma_i^k|}$$

donde $|\sigma_i^k|$ estará acotada inferiormente por $kl_{\min}$ y superiormente por $kl_{\max}$, siendo

$$l_{\min} = \text{mín}\,(l_i\,; 0 \leq i \leq q-1)$$
$$l_{\max} = \text{máx}\,(l_i\,; 0 \leq i \leq q-1).$$

Si se representa por n_l el número de palabras de $\mathcal{C}^k$ de longitud l, entonces resultará que

$$\left(\sum_{i=0}^{q-1} D^{-l_i}\right)^k = \sum_{l=kl_{\min}}^{kl_{\max}} n_l D^{-l}.$$

Y como, por ser $\mathcal{C}$ unívocamente decodificable, $n_l \leq D^l$, entonces

$$\left(\sum_{i=0}^{q-1} D^{-l_i}\right)^k \leq \sum_{l=kl_{\min}}^{kl_{\max}} 1$$

Por otra parte, al ser $kl_{\min} \geq 0$, se cumple que

$$\left(\sum_{i=0}^{q-1} D^{-l_i}\right)^k \leq kl_{\max}$$

de donde $\sum_{i=0}^{q-1} D^{-l_i} \leq (kl_{\max})^{\frac{1}{k}} \quad \forall k$.

Obsérvese por último que $(kl_{\max})^{\frac{1}{k}}$ es una función decreciente de k, y también que $\lim_{k\to\infty}(kl_{\max})^{\frac{1}{k}} = 1$; y se tiene como conclusión que

$$\sum_{i=0}^{q-1} D^{-l_i} \leq 1. \qquad \blacktriangleright$$

3.3. Códigos óptimos

Se ha establecido en el apartado anterior la existencia de una cota inferior para la longitud de un código instantáneo (unívocamente decodificable) de una fuente dada: $L \geq H_D(X)$. Es fácil asimismo verificar que tal cota es alcanzable codificando los símbolos de la fuente de uno en uno, pero sólo en el caso de que las probabilidades de los símbolos de la fuente cumplan cierta condición, que se establece a continuación.

Teorema 3.5. *Dado un código instantáneo (unívocamente decodificable) de longitud L, es condición necesaria y suficiente para que $L = H_D(X)$ que se cumpla que $p_i = D^{-l_i} \quad \forall i$.*

Demostración. Ya se ha visto en el apartado anterior que

$$H_D(X) - L = \sum_i p_i \log_D \frac{D^{-l_i}}{p_i}.$$

Resulta, pues, obvio que es condición suficiente para que $H_D(X) - L = 0$ que se cumpla $p_i = D^{-l_i} \quad \forall i$. Demostremos ahora que dicha condición es también necesaria.

Sea $q_i = \dfrac{D^{-l_i}}{\sum_i D^{-l_i}}$. Se verifica que $q_i > 0 \quad \forall i$, y $\sum_i q_i = 1$. Por tanto, en virtud de la desigualdad de Gibbs, se tiene que

$$\begin{aligned} H_D(X) &\leq \sum_i p_i \log_D \frac{1}{q_i} \\ &= \sum_i p_i \log_D \frac{1}{D^{-l_i}} + \sum_i p_i \log_D \sum_i D^{-l_i} \\ &= \sum_i p_i l_i + \log_D \sum_i D^{-l_i} \\ &= L + \log_D \sum_i D^{-l_i}. \end{aligned}$$

Y dado que, por el teorema de Kraft (teorema de McMillan), $\sum_i D^{-l_i} \leq 1$, entonces

$$\log_D \sum_i D^{-l_i} \leq 0$$

por lo que, para que se cumpla que $H_D(X) = L$, se tienen que cumplir simultáneamente las dos condiciones siguientes:

$$\begin{gathered} \textstyle\sum_i D^{-l_i} = 1 \\ p_i = q_i = \dfrac{D^{-l_i}}{\displaystyle\sum_i D^{-l_i}} \quad \forall i \end{gathered}$$

es decir, se tiene que cumplir que $D^{-l_i} = p_i \quad \forall i$. ▶

Una consecuencia interesante de este teorema es el siguiente corolario:

Corolario 3.6 (Fuente incompresible). *Si X es una variable aleatoria uniforme con un rango de D valores de probabilidad no nula, los mensajes de la fuente X no se pueden compactar.*

DEMOSTRACIÓN. Efectivamente, en este caso $L = H_D(X) = 1$, por lo que no se puede hacer nada mejor que dejar el mensaje de la fuente invariado. ▶

Desafortunadamente, en general $\log_D \frac{1}{p_i}$ no es una cantidad entera, y no se puede hacer siempre $l_i = \log_D \frac{1}{p_i} \quad \forall i$. En tales casos, para que se cumpla la inecuación de Kraft, podríamos tomar

$$l_i = \left\lceil \log_D \frac{1}{p_i} \right\rceil$$

ya que así se garantiza el cumplimiento de la condición $\sum_i D^{-l_i} \leq 1$; veámoslo:

$$l_i \geq \log_D \frac{1}{p_i} \Rightarrow D^{-l_i} \leq p_i$$

$$D^{-l_i} \leq p_i \Rightarrow \sum_i D^{-l_i} \leq \sum_i p_i = 1.$$

Por otra parte, los valores de l_i así elegidos verifican que

$$l_i < \log_D \frac{1}{p_i} + 1 \qquad \forall i$$

por lo que se puede obtener una cota superior para la longitud del código.

TEOREMA 3.7. *Dada una distribución de probabilidades $\{p_i\}$ cualquiera, siempre es posible construir un código instantáneo de longitud*

$$L < H_D(X) + 1$$

tomando $l_i = \lceil \log_D \frac{1}{p_i} \rceil$.

DEMOSTRACIÓN.

$$\begin{aligned} L &= \sum_i p_i l_i \\ &< \sum_i p_i \left(\log_D \frac{1}{p_i} + 1 \right) \\ &= \sum_i p_i \log_D \frac{1}{p_i} + \sum_i p_i \\ &= H_D(X) + 1. \end{aligned}$$

▶

3.3.1. Teorema de Shannon de codificación de fuente

Acabamos de ver que, para algunas fuentes particulares, se puede obtener fácilmente un código instantáneo de longitud $L = H_D(X)$. Pero esto no es siempre posible; lo que sí es posible siempre, sin embargo, es obtener un código instantáneo de longitud acotada superiormente, $L < H_D(X) + 1$, si bien en algunos casos, como muestra el siguiente ejemplo, ésta puede ser una cota muy conservadora.

Ejemplo 3.6.

$$\begin{aligned}&\mathbf{p} = (0{,}7, 0{,}1, 0{,}1, 0{,}1);\\ &H_2(\mathbf{p}) = 1{,}36;\\ &H_2(\mathbf{p}) + 1 = 2{,}36\end{aligned}$$ ■

Vamos a probar ahora un hecho trascendental: vamos a demostrar que siempre sería posible representar los símbolos de la fuente con un número de símbolos del alfabeto de codificación igual (en media) a la entropía.

Teorema 3.8 (Teorema de Shannon de codificación de fuente). *Sea X una fuente discreta sin memoria. Siempre es posible hallar un código instantáneo para los mensajes de dicha fuente cuya longitud sea tan próxima como se quiera a $H_D(X)$, o lo que es lo mismo, que se cumpla que*

$$L < H_D(X) + \frac{1}{n}$$

con n tan grande como se quiera.

Demostración. Codifiquemos los símbolos de la fuente en bloques de longitud n. En virtud del teorema anterior, se sabe que es posible hacerlo de forma que se cumpla que

$$L_n < H_D(X^n) + 1$$

siendo X^n la variable aleatoria que representa las n-tuplas de símbolos de la fuente.

Pero como, por otra parte, $L_n = nL$ por definición, y $H_D(X^n) = nH_D(X)$ por ser las componentes de X^n independientes e idénticamente distribuidas; entonces, dividiendo ambos términos de la desigualdad por n, se tiene que

$$L < H_D(X) + \frac{1}{n}.$$ ▶

En consecuencia, se dirá que un código de una fuente dada, X, es óptimo cuando se cumpla que $L = H_D(X)$. Además, el conocimiento del valor de la longitud de un código óptimo permite conocer la eficiencia de cualquier otro código sin más que comparar su longitud con la del código óptimo.

DEFINICIÓN 3.7 (EFICIENCIA DE UN CÓDIGO). *Si D es el número de elementos del alfabeto de codificación, se define la eficiencia de un código de una fuente X como*

$$\eta = \frac{H_D(X)}{L}.$$

Obviamente, se verifica que $0 \leq \eta \leq 1$.

Que la eficiencia de un código sea η significa que $H_D(X) = \eta L$, de donde se sigue que

$$L = H_D(X) + (1 - \eta)L$$

es decir, que la longitud de dicho código excede de la longitud del código óptimo en una cantidad $(1 - \eta)L$. En virtud de esto, $(1 - \eta)$ puede considerarse como una medida de la ineficiencia de un código, y la denominaremos redundancia del mismo.

DEFINICIÓN 3.8 (REDUNDANCIA DE UN CÓDIGO). *Se define la redundancia de un código como $1 - \eta$.*

Obsérvese que un codificador de una fuente dada estará, entonces, caracterizado por su eficiencia (o, equivalentemente, por su redundancia).

Además, se va a ver a continuación que la fuente equivalente que resulta de considerar conjuntamente la fuente dada y el codificador de fuente tiene una entropía igual a la eficiencia del codificador de fuente.

LEMA 3.9. *Considérese un código de fuente no singular de una fuente dada. La entropía por unidad de tiempo de la salida del codificador de fuente es igual a la entropía por unidad de tiempo de la fuente.*

DEMOSTRACIÓN. Sea X la variable aleatoria que representa el conjunto de posibles mensajes elementales que transforma el codificador de fuente. Sea Y la variable aleatoria que representa los distintos mensajes que se utilizan para codificar los símbolos de la variable X. Por tratarse de un código de fuente no singular, el número de elementos del rango de la variable aleatoria Y no puede ser inferior al número de elementos del rango de la variable aleatoria X; y a dos elementos distintos del rango de X se le asignan dos elementos distintos del rango de Y. En consecuencia, se cumple que $H(Y) = H(X)$ y que $v_y = v_x$, siendo v_x y v_y los regímenes de generación

de símbolos de las fuentes X e Y, respectivamente. Por tanto, se tiene, efectivamente, que $H(Y)v_y = H(X)v_x$. ▶

Teorema 3.10. *Considérese un código de fuente de eficiencia η de una fuente dada X. Sea D el número de elementos del alfabeto de codificación y sea Y la variable aleatoria discreta que representa la salida del codificador de fuente. Se tiene entonces que*

$$H_D(Y) = \eta.$$

Demostración. En virtud del lema anterior, se tiene que

$$H_D(X)v_f = H_D(Y)v_{cf}$$

siendo v_f y v_{cf} los regímenes de generación de símbolos de la fuente y del codificador de fuente, respectivamente.

Además, se cumple que

$$v_{cf} = v_f L_f = v_f \frac{H_D(X)}{\eta}$$

siendo L_f la longitud del código de fuente.

Por tanto, resulta que, efectivamente, $H_D(Y) = \eta$. ▶

Una consecuencia inmediata del teorema anterior es la siguiente.

Corolario 3.11. *Los mensajes que salen de un codificador de fuente óptimo no se pueden compactar adicionalmente.*

Demostración. Efectivamente, basta considerar $\eta = 1$ en el teorema anterior. ▶

3.4. Algoritmo de Huffman

La consecución de una eficiencia tan próxima a la unidad como se quiera se obtiene, en general, a través del aumento del valor de n, con independencia de que el método empleado para la obtención del código (elección de unas longitudes l_i para las palabras del código en función de las probabilidades de los elementos que vayan a representar de acuerdo con la fórmula $l_i = \left\lceil \log_D \frac{1}{p_i} \right\rceil$) sea o no óptimo. De hecho, es fácil comprobar que tal método no es siempre óptimo; ilustrémoslo con un ejemplo.

EJEMPLO 3.7. Considere el siguiente caso:

p_i	$l_i = \left\lceil \log \frac{1}{p_i} \right\rceil$
0,7	1
0,1	4
0,1	4
0,1	4

En un ejemplo sencillo como éste resulta evidente que se puede construir también un código instantáneo con palabras de longitudes $\{1, 2, 3, 3\}$, que tiene, claro está, una longitud menor:

p_i	C_i
0,7	0
0,1	10
0,1	110
0,1	111

■

En un caso tan sencillo como el anterior es fácil convencerse de que no existe un conjunto de longitudes $\{l_i\}$ con una longitud media menor que la del último código. Pero en un caso general, ¿cómo obtener las l_i con la menor longitud media posible?, o equivalentemente, ¿cómo construir un código cuya L sea la menor posible? Pues bien, en esta sección se va a explicar un procedimiento constructivo, descubierto por D. A. Huffman, para encontrar un conjunto óptimo de palabras para representar un conjunto dado de mensajes. Por óptimo se entenderá aquí que ningún otro conjunto de palabras del código tenga una longitud media menor. A tales códigos los denominaremos *códigos compactos*.

Sean los símbolos del alfabeto fuente $0, 1, \ldots, q-1$, y sean sus probabilidades respectivas $p_0, p_1, \ldots, p_{q-1}$. Supóngase, para simplificar la notación, que $p_0 \geq p_1 \geq \cdots \geq p_{q-1}$. Sea $\sigma_0, \sigma_1, \ldots, \sigma_{q-1}$ el conjunto de palabras del código para la fuente considerada, y sean $l_0, l_1, \ldots, l_{q-1}$ las longitudes respectivas de dichas palabras del código. Se verá a continuación que existe al menos un código compacto que cumple ciertas condiciones; y veremos también cómo usar dichas condiciones para encontrar el código.

Consideremos en primer lugar, por razón de simplicidad, el caso de códigos binarios; para a continuación generalizar el procedimiento para un alfabeto de codificación arbitrario.

3.4.1. Alfabeto de codificación binario

Lema 3.12. *Existe al menos un código instantáneo compacto en el que*

$$l_{q-1} \geq l_i \qquad \forall\, i = 0, 1, \ldots, q-2.$$

Demostración. Suponga que existe un i tal que $l_i > l_{q-1}$. Bastaría con intercambiar las palabras del código σ_i y σ_{q-1} para obtener el código compacto deseado, ya que tal intercambio no incrementa la longitud del código

$$\begin{aligned}\Delta L &= p_i l_{q-1} + p_{q-1} l_i - p_i l_i - p_{q-1} l_{q-1} \\ &= (p_i - p_{q-i})(l_{q-i} - l_i) \leq 0.\end{aligned}$$

▶

Teorema 3.13. *Para cualquier fuente con $q \geq 2$ existe un código instantáneo binario compacto en el que las dos palabras del código menos probables, σ_{q-2} y σ_{q-1}, tienen la misma longitud y difieren sólo en el último símbolo.*

Demostración. En cualquier código instantáneo binario compacto en el que l_{q-1} sea la longitud mayor, debe haber otra palabra del código σ_i $(i \neq q-1)$ de la misma longitud que σ_{q-1} y que difiera de esta última únicamente en el último bit; ya que de lo contrario, no siendo σ_i prefijo de σ_{q-1}, podría eliminarse el último bit de σ_{q-1} para obtener un código instantáneo de menor longitud que el dado, lo que es imposible.

Si no fuese $i = q-2$, resultaría que $l_i \geq l_{q-2}$ (recuérdese que $l_i = l_{q-1}$ es la mayor longitud). Y, de acuerdo con la demostración del lema anterior, podrían intercambiarse σ_i y σ_{q-2} sin incremento de la longitud del código, con lo que queda probado que siempre existe un código instantáneo binario compacto en el que $l_{q-2} = l_{q-1}$, difiriendo σ_{q-2} y σ_{q-1} únicamente en el último bit. ▶

Como corolario del teorema anterior, se puede probar fácilmente que el problema de la construcción de un código instantáneo compacto para un conjunto de q mensajes (C) se reduce al de construir $\sigma_0, \sigma_1, \ldots, \sigma_{q-3}$ y encontrar los $l_{q-1}-1$ dígitos comunes de σ_{q-2} y σ_{q-1}, es decir, construir un código compacto (C') para un conjunto de $q-1$ mensajes con probabilidades p'_i $(i = 0, 1, \ldots, q-2)$ tales que

$$p'_i = \begin{cases} p_i & i \leq q-3 \\ p_i + p_{i+1} & i = q-2. \end{cases}$$

Corolario 3.14. *Si C es un código instantáneo compacto, entonces C' también lo es.*

DEMOSTRACIÓN. Sean L' y L, respectivamente, las longitudes de los códigos C' y C. Se tiene que

$$\begin{aligned} L &= \sum_{i=0}^{q-3} p_i l_i + l_{q-1}(p_{q-2} + p_{q-1}) \\ &= \sum_{i=0}^{q-3} p'_i n'_i + (n'_{q-2} + 1)p'_{q-2} \\ &= \sum_{i=0}^{q-2} p'_i n'_i + p'_{q-2} \\ &= L' + p'_{q-2}. \end{aligned}$$

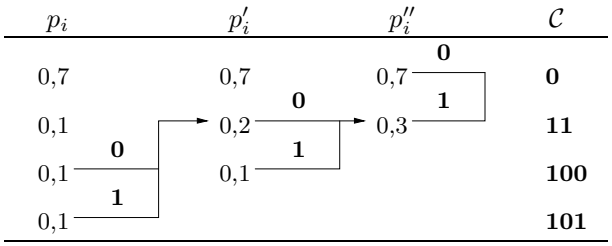
Según esto, L y L' difieren en un valor constante; y por tanto L no sería mínimo si L' no lo fuese. ▶

Si se aplica recurrentemente este procedimiento, nos encontramos con que el problema se reduce finalmente al problema de encontrar un código instantáneo compacto para una fuente binaria, problema que se resuelve de forma trivial asignando el símbolo 0 a uno de los elementos de la fuente y el símbolo 1 al otro.

EJEMPLO 3.8. Para una fuente cuaternaria con distribución de probabilidad

$$\mathbf{p} = (0{,}7, 0{,}1, 0{,}1, 0{,}1)$$

y un alfabeto de codificación binario, el algoritmo de Huffman se aplica del modo siguiente:



■

3.4.2. Alfabeto de codificación arbitrario

Al extender este procedimiento al caso más general de alfabetos de codificación no binarios surge un problema nuevo: el teorema 3.13 deja de ser

válido, surgiendo el problema de que haya más palabras que σ_{q-2} que difieran de σ_{q-1} en el último bit. Ahora bien, es fácil acotar el número máximo de palabras que pueden diferir de σ_{q-1} en el último bit.

Teorema 3.15. *El número de palabras del código que difieren de σ_{q-1} en el último bit es*

$$1 + R_{D-1}(q-2)$$

siendo $R_{D-1}(q-2)$ el resto de dividir $q-2$ entre $D-1$.

Demostración. El árbol completo más pequeño que representa un código instantáneo tiene D nodos terminales (que son todos los nodos de primer orden del árbol). Si este número de nodos no bastase para representar todas las palabras del código, entonces habría que extender el árbol a partir de alguno de los nodos terminales de primer orden. Por cada nodo de primer orden que se extienda completamente (es decir, que genere D nuevos nodos de segundo orden), se tendrán $D-1$ nuevos nodos terminales: los nuevos D nodos de segundo orden menos el nodo de primer orden del que éstos proceden (que ha dejado de ser nodo terminal). Realizando esta misma operación tantas veces como se precise, el número de nodos terminales del árbol será $D + m(D-1)$ para algún entero m.

Ahora bien, es ciertamente posible que alguno de los nodos terminales no se utilice para la construcción del código (porque haya más nodos terminales de los necesarios para el código); llamemos nodos libres a tales nodos terminales. Resulta claro también que en un código óptimo todos los nodos terminales libres deben tener el mismo orden que la longitud de la palabra del código más larga. O dicho de otra forma, intercambiando nodos terminales utilizados y libres, pueden disponerse los nodos libres de forma que difieran entre sí sólo en el último símbolo.

Así, el número de nodos libres del árbol correspondiente a un código óptimo, representémoslo por V, no pueden exceder de $D-2$, dado que, si se agrupasen juntos $D-1$ nodos libres, la palabra del código correspondiente al nodo utilizado restante podría acortarse sin violar la condición del prefijo. Como, por otra parte, el número de palabras del código más el número de nodos libres del correspondiente árbol es de la forma $D+m(D-1)$ para algún entero m, el número de nodos libres de tal árbol queda así unívocamente determinado:

$$V + q = D + m(D-1); \qquad 0 \leq V \leq D-2$$

Reescribiendo la igualdad anterior de la siguiente forma:

$$q - 2 = m(D-1) + D - 2 - V$$

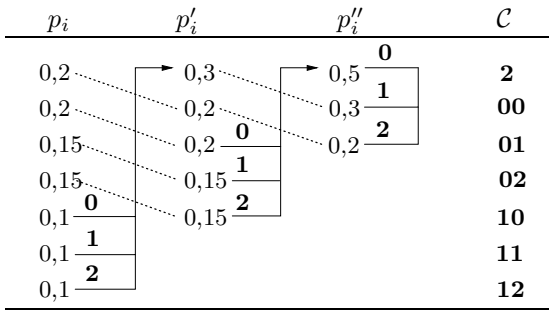
se observa que $D-2-V$ es el resto de dividir $q-2$ entre $D-1$:

$$D-2-V = R_{D-1}(q-2)$$

de donde se obtiene inmediatamente que $D-V = 2+R_{D-1}(q-2)$. ▶

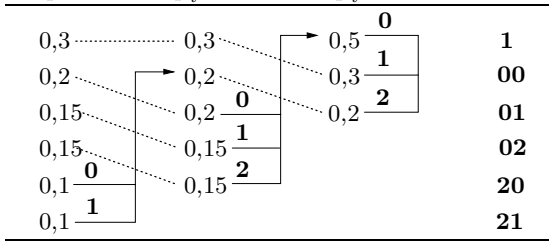
Así, el primer paso en el proceso de codificación consistirá en agrupar juntos los $2+R_{D-1}(q-2)$ nodos menos probables. A partir de ese paso, deja de haber nodos libres, y se puede hacer que los D nodos menos probables difieran entre sí en el último símbolo. Continuando de esta forma, finalmente se obtendrá un conjunto de D mensajes, para el que se obtiene fácilmente un código compacto por el simple procedimiento de asignar cada uno de los D símbolos del alfabeto de codificación a cada uno de dichos mensajes.

Ejemplo 3.9. Considere una fuente de alfabeto de 7 símbolos con probabilidades $\mathbf{p} = (0{,}2, 0{,}2, 0{,}15, 0{,}15, 0{,}1, 0{,}1, 0{,}1)$, y sea el alfabeto de codificación ternario. El algoritmo de Huffman proporciona el siguiente resultado:



Obsérvese que, en este caso, no hay nodos libres. ■

Ejemplo 3.10. Considere ahora una fuente de alfabeto de 6 símbolos con probabilidades $\mathbf{p} = (0{,}3, 0{,}2, 0{,}15, 0{,}15, 0{,}1, 0{,}1)$, y sea el alfabeto de codificación ternario como antes. Ahora, el algoritmo de Huffman proporciona el siguiente resultado:



Obsérvese que, en este caso, habría un nodo libre. ■

3.5. Tasa de entropía de un proceso estocástico*

En el apartado 3.3.1 se ha establecido el teorema de Shannon de codificación de fuente para el caso particular, más sencillo, de fuentes discretas sin memoria. Abordemos ahora el teorema de Shannon de codificación de fuente para un modelo más general de fuente. Si se aplican los resultados del corolario 3.3 y del teorema 3.7 al vector aleatorio n-dimensional $(X_1, X_2, \ldots, X_n)$, donde X_i es la variable aleatoria discreta que representa el símbolo i-ésimo emitido por la fuente, se tiene que

$$\frac{H(X_1, X_2, \ldots, X_n)}{n} \leq \frac{L_n}{n} < \frac{H(X_1, X_2, \ldots, X_n)}{n} + \frac{1}{n}.$$

Si existiese el límite

$$\lim_{n\to\infty} \frac{H(X_1, X_2, \ldots, X_n)}{n} \tag{3.2}$$

tal límite sería el mínimo número medio de símbolos del alfabeto de codificación por cada símbolo de la fuente.

Si $X_1, X_2, \ldots, X_n$ son variables aleatorias i.i.d. de entropía $H(X)$, entonces se cumple que $H(X_1, X_2, \ldots, X_n) = nH(X)$; y, por tanto,

$$\frac{H(X_1, X_2, \ldots, X_n)}{n} = H(X); \quad \forall n.$$

Ahora bien, en el caso más general de que $\{X_i\}_{i=1,2,\ldots}$ sea un proceso estocástico arbitrario, puede o no existir el límite (3.2). Cuando dicho límite existe recibe el nombre de *tasa de entropía* del proceso estocástico.

Definición 3.9 (Tasa de entropía). *Se define la tasa de entropía de un proceso estocástico de tiempo discreto* $\{X_i\}_{i=1,2,\ldots}$ *como*

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{H(X_1, X_2, \ldots, X_n)}{n}$$

cuando el límite existe.

Teorema 3.16. *Si el proceso estocástico* $\{X_i\}_{i=1,2,\ldots}$ *es estacionario, entonces existe el* $\lim_{n\to\infty} \frac{H(X_1, X_2, \ldots, X_n)}{n}$ *y se cumple que*

$$\lim_{n\to\infty} \frac{H(X_1, X_2, \ldots, X_n)}{n} = \lim_{n\to\infty} H(X_n \,|\, X_{n-1}, X_{n-2}, \ldots, X_1).$$

Demostración. Por la regla de la cadena,

$$\frac{H(X_1, X_2, \ldots, X_n)}{n} = \frac{1}{n} \sum_{i=1}^{n} H(X_i \,|\, X_{i-1}, X_{i-2}, \ldots, X_1).$$

Por otra parte, para un proceso estocástico estacionario

$$\begin{aligned} H(X_{n+1} \,|\, X_n, X_{n-1}, \ldots, X_1) &\leq H(X_{n+1} \,|\, X_n, X_{n-1}, \ldots, X_2) \\ &= H(X_n \,|\, X_{n-1}, \ldots, X_1) \end{aligned}$$

donde la igualdad se sigue de la estacionariedad del proceso. Es decir,

$$H(X_n \,|\, X_{n-1}, \ldots, X_1)$$

es una secuencia de números no negativos que decrece con n, por lo que tiene límite.

A continuación vamos a probar que si existe $\lim_{n\to\infty} H(X_n \,|\, X_{n-1}, \ldots, X_1)$, entonces se cumple que

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} H(X_i \,|\, X_{i-1}, \ldots, X_1) = \lim_{n\to\infty} H(X_n \,|\, X_{n-1}, \ldots, X_1).$$

Sean

$$\begin{aligned} H(X_i \,|\, X_{i-1}, X_{i-2}, \ldots, X_1) &= a_i \\ \lim_{n\to\infty} a_n &= a \\ \frac{1}{n} \sum_{i=1}^{n} a_i &= b_n. \end{aligned}$$

Por ser $\lim_{n\to\infty} a_n = a$, entonces existe un número $N(\epsilon)$ tal que

$$|a_n - a| \leq \epsilon \;\; \forall n \geq N(\epsilon).$$

De aquí que, $\forall n \geq N(\epsilon)$:

$$\begin{aligned} |b_n - a| &= \left| \frac{1}{n} \sum_{i=1}^{n} (a_i - a) \right| \\ &\leq \frac{1}{n} \sum_{i=1}^{n} |a_i - a| \\ &\leq \frac{1}{n} \sum_{i=1}^{N(\epsilon)} |a_i - a| + \frac{n - N(\epsilon)}{n} \epsilon \\ &\leq \frac{1}{n} \left[\sum_{i=1}^{N(\epsilon)} |a_i - a| - \epsilon N(\epsilon) \right] + \epsilon. \end{aligned}$$

De forma que, tomando n suficientemente grande ($n \to \infty$), podemos hacer $|b_n - a| \leq \epsilon$. ▶

3.6. Tasa de entropía de una cadena de Markov*

De entre todos los procesos estocásticos, los procesos de Markov merecen especial atención por su sencillez y utilidad.

En esencia, un proceso estocástico es una familia de variables aleatorias $X(t)$ donde t es un parámetro índice (típicamente temporal). Tres son las características definitorias de un proceso estocástico:

- El *espacio de estados*, o conjunto de posibles valores (estados) de las variables aleatorias $X(t)$, que puede ser un conjunto discreto (y en ese caso el proceso estocástico se denomina habitualmente *cadena*) o un conjunto continuo.
- El *parámetro índice (tiempo)*, que puede tomar valores en un conjunto discreto (y se tiene entonces un proceso de parámetro —tiempo— discreto) o continuo (y se tiene entonces un proceso de parámetro —tiempo— continuo). En el primer caso, es habitual referirse al proceso estocástico como *secuencia aleatoria* y representarla usando X_n en lugar de $X(t)$.
- Y por último, la más importante de todas, que es la *relación entre las variables aleatorias de la familia*. En general, para caracterizar un proceso estocástico hay que especificar la función de distribución conjunta de todas las variables aleatorias de la familia, lo que puede resultar una tarea inabordable. Afortunadamente, muchos procesos estocásticos de interés, como por ejemplo los procesos de Markov, admiten una descripción mucho más sencilla.

Un proceso de Markov es un proceso estocástico en el que el estado siguiente depende única y exclusivamente del estado actual; y si el espacio de estados del proceso es discreto suele denominarse, como ya hemos dicho, cadena de Markov. Además, como nos interesan únicamente los procesos de tiempo discreto, definiremos una cadena de Markov de la siguiente manera:

Definición 3.10 (Cadena de Markov). *Se dice que una secuencia estocástica* $\{X_i,\ i = 1, 2, \dots\}$ *de espacio de estados discreto* $\mathcal{X}$ *es una cadena de Markov si para todo valor de* n *y para cualesquiera* $x_1, \dots, x_{n+1} \in \mathcal{X}$ *se cumple que*

$$\mathrm{P}(X_{n+1} = x_{n+1} \,|\, X_n = x_n, X_{n-1} = x_{n-1}, \dots, X_1 = x_1) = \mathrm{P}(X_{n+1} = x_{n+1} \,|\, X_n = x_n).$$

La propiedad anterior, que define a una cadena de Markov, suele denominarse propiedad markoviana. Y en virtud de esta propiedad tenemos una

secuencia aleatoria en la que la dependencia se extiende hacia atrás en el tiempo una sola unidad.

La expresión $\mathrm{P}(X_{n+1} = j \,|\, X_n = i)$ se denomina *probabilidad de transición en un paso* y es la probabilidad condicional de que se produzca una transición del estado i en el instante de tiempo n al estado j en el instante de tiempo $n + 1$.

Si ocurre, como vamos a suponer de ahora en adelante, que las probabilidades de transición son independientes del valor de n, entonces se tiene una cadena de Markov *invariante con el tiempo* (u *homogénea*), y en ese caso se representará por P_{ij} la probabilidad de transición en un solo paso del estado i al estado j,

$$\mathrm{P}_{ij} = \mathrm{P}(X_{n+1} = j \,|\, X_n = i).$$

De manera análoga, se puede definir la probabilidad de transición en n pasos, que representaremos por $\mathrm{P}_{ij}^{(n)}$, como la probabilidad de que un proceso que está en el estado i llegue al estado j tras n transiciones, es decir, n instantes de tiempo más tarde. Esto es,

$$\mathrm{P}_{ij}^{(n)} = \mathrm{P}(X_{m+n} = j \,|\, X_m = i), \quad n > 0, \quad i, j \in \mathcal{X}.$$

Se dice que se puede acceder al estado j desde el estado i si para algún $n > 0$, $\mathrm{P}_{ij}^{(n)} > 0$. Se dice, además, que dos estados i y j se comunican si desde cualquiera de ellos se puede acceder al otro. Es trivial, entonces, la verificación de que la comunicación entre dos estados de una cadena de Markov es una relación de equivalencia. Esta relación de equivalencia permite agrupar los estados de una cadena de Markov en clases de equivalencia; y se dice que una cadena de Markov es irreducible si existe una única clase de equivalencia, es decir, si cada estado de la misma se comunica con todos los demás estados.

Definición 3.11 (Cadena de Markov irreducible). *Se dice que una cadena de Markov es irreducible si cada estado de la misma se comunica con todos los demás estados.*

Además, se dice que un subconjunto de los estados de una cadena de Markov es *cerrado* si ninguno de sus estados se comunica con ningún otro estado que no pertenezca al mismo subconjunto. Si el conjunto de los estados de una cadena de Markov es cerrado y no contiene ningún subconjunto propio que también lo sea, entonces se tiene una cadena de Markov irreducible. De lo contrario, la cadena se dice reducible; y los subconjuntos cerrados que no contengan otros subconjuntos cerrados se denominan *subcadenas de Markov irreducibles*. Estas subcadenas pueden estudiarse independientemente

del resto de estados, por lo que de ahora en adelante consideraremos únicamente cadenas irreducibles de Markov.

Una característica de los estados de la cadena que resulta de interés es la posibilidad de que el sistema vuelva al mismo estado en algún instante futuro, y la frecuencia con la que este hecho se produce. Si el número de pasos necesarios para alcanzar de nuevo el estado j sólo puede ser un número d ($d > 1$) y sus múltiplos enteros, entonces se dice que el estado es *periódico*, siendo su periodo d. En caso contrario, se dice que el estado es *aperiódico*. Se puede demostrar también que la periodicidad de un estado es una propiedad de clase, de forma que en una cadena de Markov irreducible o todos los estados son aperiódicos o todos son periódicos (y con idéntico periodo).

Definición 3.12 (Cadena de Markov aperiódica). *Una cadena de Markov es aperiódica si todos los estados de la misma son aperiódicos.*

Representemos por $f_j^{(n)}$ la probabilidad de que el primer regreso al estado j se produzca al cabo de exactamente n instantes de tiempo. Está claro que la probabilidad de regresar alguna vez al estado j está dada por

$$f_j = \sum_{n=1}^{\infty} f_j^{(n)}.$$

Los estados de una cadena de Markov se pueden clasificar ahora en función del valor obtenido para f_j:

- si $f_j = 1$, se dice que el estado es *recurrente*;
- si $f_j < 1$, se dice que el estado es *transitorio*.

Se puede demostrar que también la recurrencia, al igual que la periodicidad, es una propiedad de clase. Por tal motivo, los estados de una cadena de Markov irreducible o son todos transitorios o son todos recurrentes. Ahora bien, puesto que si un estado j es recurrente, con probabilidad uno se vuelve alguna vez a él, con probabilidad uno el número de visitas a j será infinito; y por tanto será infinito el número medio de regresos a j. Por otra parte, si el estado j es transitorio, hay una probabilidad $(1 - f_j)$ de que nunca se regrese de nuevo a él; por lo que el número de visitas a ese estado es una variable aleatoria geométrica de media finita $1/(1 - f_j)$. Lo que nos lleva a la conclusión de que en una cadena de Markov de estados finitos no todos los estados pueden ser transitorios; ya que si así fuese, pasado un tiempo finito no se visitaría ningún estado, lo que es imposible. En consecuencia, en una cadena de Markov irreducible de estados finitos, todos los estados son recurrentes.

Para los estados recurrentes de una cadena de Markov se puede definir el tiempo medio de recurrencia de un estado j como

$$T_j = \sum_{n=1}^{\infty} n f_j^{(n)}$$

y, de acuerdo con este parámetro, los estados recurrentes pueden clasificarse a su vez en:

- *recurrentes positivos*, si $T_j < \infty$;
- *recurrentes nulos*, en caso contrario.

También la recurrencia positiva (nula) es una propiedad de clase. Y todos los estados de una cadena de Markov irreducible de estados finitos son recurrentes positivos.

Definición 3.13 (Cadena de Markov ergódica). *Un estado aperiódico y recurrente positivo se denomina ergódico. Y se dice que una cadena de Markov es ergódica cuando todos sus estados son ergódicos.*

Por consiguiente, una cadena de Markov de estados finitos, irreducible y aperiódica es ergódica.

Definamos ahora $\pi_j^{(n)}$ como la probabilidad de que en el instante de tiempo n el sistema se encuentre en el estado j, esto es,

$$\pi_j^{(n)} = \mathrm{P}(X_n = j).$$

Una distribución de probabilidad de los estados que sea independiente del instante de tiempo n se llama *distribución estacionaria*. Y recibe tal nombre porque si el estado inicial de una cadena de Markov se elige de forma que la distribución sea estacionaria, entonces la cadena de Markov es un *proceso estocástico estacionario*. En tales casos, una cuestión fundamental es la obtención de la distribución de probabilidad estacionaria $\{\pi_j\}$, que se aborda en el siguiente teorema.

Teorema 3.17. *Si la cadena de Markov es ergódica, entonces la distribución estacionaria es única y la distribución de X_n tiende a la distribución estacionaria según $n \to \infty$, sin importar cual sea la distribución inicial. Además, la distribución se obtiene como la única solución no negativa del*

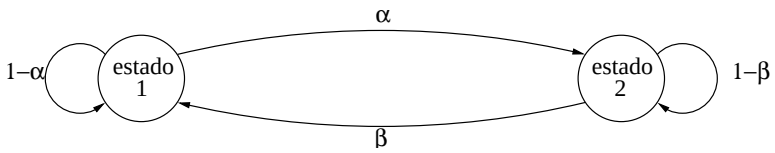
siguiente sistema de ecuaciones lineales:

$$\pi_j = \sum_i \pi_i \, \mathrm{P}_{ij}$$
$$\sum_j \pi_j = 1.$$

Ejemplo 3.11. Considere una cadena de Markov de dos estados con matriz de probabilidades de transición

$$P = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}.$$

Representemos la distribución estacionaria por un vector $\pi = (\pi_1, \pi_2)$ cuyas componentes serán las probabilidades estacionarias del estado 1 y del estado 2, respectivamente.



Entonces las probabilidades estacionarias deben cumplir:

$$\pi_1 \alpha = \pi_2 \beta$$
$$\pi_2 = 1 - \pi_1,$$

cuya solución es $\pi_1 = \dfrac{\beta}{\alpha+\beta}, \pi_2 = \dfrac{\alpha}{\alpha+\beta}$. ■

Teorema 3.18 (Tasa de entropía de una cadena de Markov). *Sea una cadena de Markov estacionaria, con distribución estacionaria π y matriz de transición P. Entonces la tasa de entropía está dada por*

$$H(\mathcal{X}) = -\sum_{i,j} \pi_i \, \mathrm{P}_{ij} \log \mathrm{P}_{ij} \, .$$

Demostración. Por definición, la tasa de entropía de la cadena de Markov es

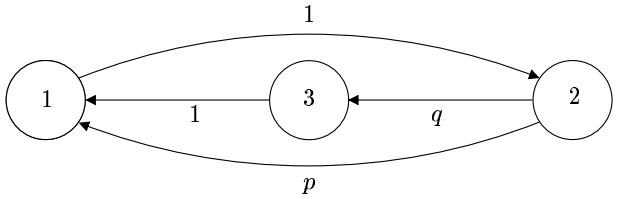
$$\begin{aligned} H(\mathcal{X}) &= \lim_{n\to\infty} H(X_n \,|\, X_{n-1}, \ldots, X_1) = \lim_{n\to\infty} H(X_n \,|\, X_{n-1}) \\ &= H(X_2 \,|\, X_1) \\ &= \sum_i \pi_i \Big(-\sum_j \mathrm{P}_{ij} \log \mathrm{P}_{ij} \Big). \end{aligned}$$

▶

Ejemplo 3.12. Consideremos una fuente binaria. Supongamos que se ha observado que dicha fuente genera uno de los símbolos (digamos el símbolo 0) en mayor cantidad que el otro (digamos que el 60 % de las veces se genera el símbolo 0). De acuerdo con esta observación, se podría modelar la fuente en cuestión como una variable aleatoria binaria X, con $\mathrm{P}(X = 0) = 0{,}6$. En tal caso, dicha fuente tendrá una entropía $H(X) = 0{,}97$ bits; por lo que, si se dispone de un codificador de fuente ideal, un mensaje de la fuente de longitud n podría representarse perfectamente con $0{,}97n$ símbolos binarios, pero no con menos.

Pero supongamos ahora que una observación más detallada de las características de la fuente nos permite darnos cuenta de una peculiaridad de la misma: nunca genera de forma consecutiva dos símbolos 1, ni más de dos símbolos 0. Esto significa, obviamente, que existe una fuerte correlación entre símbolos consecutivos (memoria); y, con toda seguridad, tal correlación permitirá una representación más compacta de los mensajes de la fuente que sean suficientemente largos.

A la luz de lo que se acaba de exponer en este apartado, la fuente que estamos considerando puede modelarse de forma más precisa mediante una cadena de Markov de tres estados: un estado (estado 1) en el que transmite un símbolo 1, y otros dos estados (estado 2 y estado 3) en los que transmite un símbolo 0. El diagrama de transiciones de esta cadena de Markov sería el siguiente:



Por tanto, se tiene que:

$$\begin{aligned} \pi_1 &= p\pi_2 + \pi_3 \\ q\pi_2 &= \pi_3 \end{aligned}$$

o lo que es lo mismo, $\pi_1 = \pi_2 = \pi_3/q$, de donde, teniendo en cuenta que $\pi_1 + \pi_2 + \pi_3 = 1$, se tiene que:

$$\begin{aligned} \pi_1 &= \frac{1}{2+q} = \pi_2 \\ \pi_3 &= \frac{q}{2+q}. \end{aligned}$$

La tasa de entropía vendrá dada por

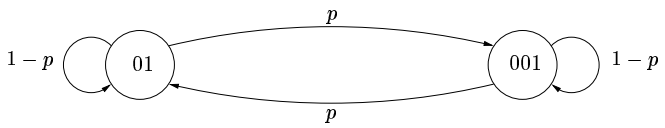
$$H(\mathcal{X}) = \pi_2 H(q) = \frac{1}{2+q} H(q).$$

Y como, en este caso, se tiene que:

$$\pi_1 = 0{,}4$$
$$\pi_2 = \pi_1 = \frac{1}{2+q}$$

resulta que $q = 0{,}5$, por lo que $H(\mathcal{X}) = 0{,}4$ bits; con lo que un mensaje de la fuente de longitud n (suficientemente grande) podría representarse sólo con $0{,}40n$ símbolos binarios. Obsérvese el notabilísimo aumento de la eficiencia de la representación de los mensajes de la fuente. ■

Ejemplo 3.13. Existe otra cadena de Markov que puede emplearse, alternativamente, para el modelado de la fuente anterior. Observe que podríamos modelar igualmente la fuente en cuestión mediante una cadena de Markov de dos estados: un estado 1, en el que se transmitiría una secuencia de dos símbolos diferentes, 01 (o 10, según cuál fuese el primer bit del mensaje); y un estado 2, en el que se transmitiría una secuencia de tres símbolos, 001 (o 100). El diagrama de transiciones de esta cadena de Markov sería el siguiente



donde $H(\mathcal{X}) = \dfrac{\beta}{\alpha+\beta} H(\alpha) + \dfrac{\alpha}{\alpha+\beta} H(\beta) = 0{,}4$ bits.

Obsérvese que esta cadena de Markov consta de dos estados en los que se puede permanecer indefinidamente, por lo que este modelo es equivalente a una fuente binaria sin memoria cuyos símbolos sean las secuencias de dos y tres bits que se transmiten, respectivamente, en los estados 1 y 2 de la cadena de Markov anterior. Y nótese, entonces, cómo una adecuada definición de los símbolos permite recoger cierto grado de memoria con un modelo de fuente sin memoria. Modelaríamos entonces la fuente como una variable aleatoria binaria X de rango $\mathcal{X} = \{01, 001\}$ y $\mathrm{P}(X = 01) = 0{,}5$. Así definida X, resulta que $H(X) = 1$ bit. Es decir, se necesita, en media, un bit para representar cada símbolo de esta fuente. Pero los símbolos de esta fuente tienen una longitud media de 2,5 bits; lo que quiere decir que un mensaje suficientemente largo de la fuente puede reducirse hasta un 40 % de su longitud original, que es el mismo resultado de antes. ■

Notas bibliográficas

Shannon estableció [59] el teorema de codificación de fuente que lleva su nombre para el caso de fuentes markovianas, si bien de una forma diferente a como se ha presentado aquí, basándose en un resultado original hoy conocido como propiedad de equipartición asintótica. Esta denominación, así como una generalización de la misma para procesos estacionarios ergódicos con alfabeto finito, se debe a McMillan [48].

La desigualdad de Kraft aparece en [37], y la demostración de que también se cumple para cualquier código unívocamente decodificable se debe a McMillan [49] (la demostración de este teorema aquí empleada aparece en [35].

El procedimiento de codificación de Huffman y la demostración de que es óptimo aparecen en [34].

3.A. La propiedad de equipartición asintótica

Shannon formuló su famoso teorema de codificación de fuente [59] de una forma diferente a como lo hemos hecho en este capítulo, a través de un resultado original que ahora se conoce como *propiedad de equipartición asintótica*, y que no es sino una consecuencia directa de la ley de los grandes números (para secuencias de variables aleatorias independientes e idénticamente distribuidas) o del teorema ergódico (para procesos estacionarios).

Nosotros nos limitaremos aquí a considerar el caso más sencillo de las secuencias de variables aleatorias independientes e idénticamente distribuidas y la convergencia en probabilidad. Para este caso particular, la propiedad de equipartición asintótica se formaliza con el siguiente teorema.

Teorema 3.19. *Si $X_1, X_2, \ldots,$ son variables aleatorias independientes y todas ellas con idéntica distribución que la variable aleatoria X, entonces se verifica la siguiente convergencia en probabilidad*

$$-\frac{1}{n}\log p(X_1, X_2, \ldots, X_n) \to H(X).$$

Demostración. Las funciones de variables aleatorias independientes son también variables aleatorias independientes. Así, puesto que las X_i son variables aleatorias independientes e idénticamente distribuidas, también lo son las $\log p(X_i)$. De aquí que, por la ley débil de los grandes números,

$$\begin{aligned} -\frac{1}{n}\log p(X_1, X_2, \ldots, X_n) &= -\frac{1}{n}\sum_{i=1}^{n}\log p(X_i) \\ &\to -\operatorname{E}[\log p(X)] \quad \text{en probabilidad} \\ &= H(X). \end{aligned}$$

▶

El teorema anterior implica que se puede estimar, con una probabilidad tan próxima a uno como se quiera, la entropía $H(X)$ a partir de una secuencia suficientemente grande de muestras independientes de la variable aleatoria X. O lo que es lo mismo, que las secuencias de muestras de X se pueden dividir en dos clases: aquéllas en las que la proporción de los distintos símbolos converge en probabilidad a la distribución de probabilidades de X, y todas las demás. Las primeras, que se denominan *secuencias típicas*, tienen una probabilidad conjunta tan próxima a uno como se quiera sin más que hacer n arbitrariamente grande; y las otras, en cambio, ocurren con una probabilidad arbitrariamente próxima a cero. Las primeras, además, tendrán probabilidades aproximadamente iguales, hecho que da origen a la denominación del teorema. El conjunto de las secuencias típicas recibe el nombre de *conjunto típico* y se puede formalizar de la siguiente manera.

DEFINICIÓN 3.14 (CONJUNTO TÍPICO). *El conjunto típico $T_\epsilon^{(n)}$ con respecto a la variable aleatoria X, de rango $\mathcal{X}$, es el conjunto de secuencias $(x_1, x_2, \ldots, x_n) \in \mathcal{X}^n$ que tienen la siguiente propiedad:*

$$2^{n(H_2(X)+\epsilon)} \leq p(x_1, x_2, \ldots, x_n) \leq 2^{n(H_2(X)-\epsilon)}.$$

Se puede demostrar fácilmente (véase [15]) que, como consecuencia de la propiedad de equipartición asintótica, el conjunto típico tiene las siguientes propiedades:

TEOREMA 3.20.

1. *Si $(x_1, x_2, \ldots, x_n) \in T_\epsilon^{(n)}$, entonces*

$$H(X) - \epsilon \leq -\frac{1}{n} \log p(x_1, x_2, \ldots, x_n) \leq H(X) + \epsilon.$$

2. *$\mathrm{P}(T_\epsilon^{(n)}) > 1 - \epsilon$ para n suficientemente grande.*

3. *El número de elementos del conjunto típico es $|T_\epsilon^{(n)}| \leq 2^{n(H_2(X)+\epsilon)}$.*

4. *$|T_\epsilon^{(n)}| \geq (1-\epsilon)2^{n(H_2(X)-\epsilon)}$ para n suficientemente grande.*

Es decir, más informalmente, el conjunto típico tiene probabilidad casi uno, todos sus elementos son aproximadamente equiprobables y el número de los mismos es aproximadamente $2^{nH_2(X)}$.

Teniendo en cuenta las propiedades anteriores, resulta fácil diseñar un procedimiento para la codificación (que supondremos binaria, sin pérdida de generalidad) de secuencias arbitrariamente largas de símbolos independientes de X de forma que la longitud del código se aproxime tanto como se quiera a $H_2(X)$. Basta asignar un número de secuencia a cada uno de los elementos tanto del conjunto típico como de su complementario, diferenciando los números de secuencia idénticos correspondientes a elementos distintos con un prefijo de un bit: por ejemplo, un 0 para los números de secuencia correspondientes a secuencias del conjunto típico, y un 1 para los otros. Ahora, si estos números de secuencia se representan mediante secuencias de bits de longitud fija:

- las palabras del código correspondientes a los elementos del conjunto típico tendrán una longitud $l_1 \leq \lceil nH_2(X) + \epsilon \rceil + 1$, ya que $|T_\epsilon^{(n)}| \leq 2^{n(H_2(X)+\epsilon)}$ y $nH_2(X) + \epsilon$ puede no ser una cantidad entera;
- las palabras del código correspondientes a los demás elementos tendrán una longitud $l_2 \leq n\lceil \log_2 |\mathcal{X}| \rceil + 1$.

Así, si n es suficientemente grande para que $\mathrm{P}(T_\epsilon^{(n)}) > 1-\epsilon$, la longitud media del código, L, es

$$\begin{aligned}
L &= \sum_{(x_1,\ldots,x_n)\in T_\epsilon^{(n)}} l_1 p(x_1,\ldots,x_n) + \sum_{(x_1,\ldots,x_n)\notin T_\epsilon^{(n)}} l_2 p(x_1,\ldots,x_n) \\
&< \sum_{(x_1,\ldots,x_n)\in T_\epsilon^{(n)}} (nH_2(X)+\epsilon+2) p(x_1,\ldots,x_n) \\
&+ \sum_{(x_1,\ldots,x_n)\notin T_\epsilon^{(n)}} (n\log_2|\mathcal{X}|+2) p(x_1,\ldots,x_n) \\
&= \mathrm{P}(T_\epsilon^{(n)})\big(nH_2(X)+\epsilon+2\big) + \big(1-\mathrm{P}(T_\epsilon^{(n)})\big)(n\log_2|\mathcal{X}|+2) \\
&< n(H_2(X)+\epsilon) + \epsilon n\log_2|\mathcal{X}| + 2 \\
&= n(H_2(X)+\epsilon')
\end{aligned}$$

donde $\epsilon' = \epsilon + \epsilon\log_2|\mathcal{X}| + \frac{2}{n}$ puede hacerse arbitrariamente pequeña con una elección apropiada de ϵ y n.

Otra manera de ver lo anterior es la siguiente. Podrían utilizarse secuencias de bits de la misma longitud N para representar las distintas secuencias de longitud n, de forma que, según el valor de N, podría haber en el código menos palabras de las necesarias para una codificación sin pérdida. Definamos P_e como la probabilidad de todas las secuencias de la fuente de longitud n para las que no se ha proporcionado una palabra del código.

Si se elige N de forma que sea $N \geq n(H_2(X)+\epsilon)$, entonces

$$2^N \geq 2^{n(H_2(X)+\epsilon)}$$

y resulta, por tanto, que existen palabras suficientes en el código al menos para todas las secuencias del conjunto típico; por lo que se tendrá que

$$P_e < \epsilon.$$

En cambio, si se elige N de forma que $N < nH_2(X)$, entonces se puede demostrar que $P_e \to 1$ a medida que $n \to \infty$. Veámoslo. Elijamos un valor de N que satisfaga $N \leq n(H_2(X)-2\epsilon)$. Entonces el número total de palabras del código, 2^N, será como mucho $2^{n(H_2(X)-2\epsilon)}$. Puesto que cada secuencia del conjunto típico (secuencia típica) tiene una probabilidad de, como máximo, $2^{-n(H_2(X)-\epsilon)}$, la probabilidad de todas las secuencias típicas para las que hay palabras del código es como mucho

$$2^{-n(H_2(X)-\epsilon)} \times 2^{n(H_2(X)-2\epsilon)} = 2^{-n\epsilon}.$$

Por otra parte, la probabilidad de todas las secuencias no pertenecientes al conjunto típico es, como mucho, ϵ. Así pues, la probabilidad del conjunto

de secuencias de la fuente, típicas o no, para las que no hay palabras del código será tal que

$$1 - P_e \leq \epsilon + 2^{-n\epsilon}.$$

En la discusión precedente, se ha elegido un alfabeto de codificación binario por simple comodidad notacional. Todo lo anterior es igualmente válido para un alfabeto de codificación de D símbolos (D arbitrario) sin más que sustituir $H_2(X)$ por $H_D(X)$.

Resumiendo todo lo anterior, puede enunciarse el teorema de codificación de fuente de la siguiente manera:

TEOREMA 3.21 (TEOREMA DE CODIFICACIÓN DE FUENTE). *Sea una fuente sin memoria discreta X y considere una codificación de secuencias de n símbolos de la fuente en secuencias de N símbolos de un alfabeto de codificación de D símbolos. Asignemos cada palabra del código a una única secuencia de símbolos de la fuente y sea P_e la probabilidad de que ocurra una secuencia de la fuente a la que no se le ha asignado ninguna palabra del código. Entonces, para cualquier $\epsilon > 0$, si*

$$\frac{N}{n} \geq H_D(X) + \epsilon$$

P_e puede hacerse arbitrariamente pequeño sin más que hacer n suficientemente grande. A la inversa, si

$$\frac{N}{n} \leq H_D(X) - \epsilon$$

entonces $P_e \to 1$ si n se hace suficientemente grande.

3.B. Técnicas de compresión

Aun siendo el método de Huffman el que proporciona el conjunto de palabras del código de longitud media mínima para un conjunto de símbolos de la fuente dado, la consecución de un código eficiente (de longitud próxima a la entropía) requiere, en general, que la unidad elemental de codificación sea no un símbolo de la fuente sino una secuencia de símbolos (de cuanta mayor longitud mejor). En realidad, existen otros métodos de codificación de secuencias de símbolos que dan resultados comparables a los del método de Huffman y presentan, además, grandes ventajas de índole práctica; como la codificación aritmética, por ejemplo, que es otro método que también se basa en el conocimiento de la naturaleza estadística de la fuente; o como el método Lempel–Ziv, que es de naturaleza completamente distinta a los anteriores, pues ignora por completo las características de la fuente.

3.B.1. Codificación aritmética

El inconveniente del método de Huffman es que requiere el cálculo previo de las probabilidades de todas las secuencias de la fuente de una longitud prefijada. La codificación aritmética, en cambio, es un método que puede extenderse fácilmente a longitudes mayores de las secuencias de la fuente sin rehacer todos los cálculos.

La codificación aritmética se basa en utilizar como palabra del código correspondiente a una cierta secuencia de la fuente de cualquier longitud n (representémoslas por x^n) una representación en base D (donde D es el número de símbolos del alfabeto de codificación) del número que expresa el valor de su función de distribución, $F(x^n)$. Y la ventaja de este método radica en que, si existe un procedimiento sencillo para calcular $p(x^n)$ para cualquier x^n, entonces $F(x^n)$ se puede calcular de manera incremental, es decir, se puede calcular $F(x^n)$ conocidos $F(x^{n-1})$ y el n-ésimo símbolo de la secuencia que se está codificando. Así se pueden codificar y decodificar los símbolos de la fuente de manera secuencial, es decir, a medida que se reciben.

Consideremos en primer lugar la cuestión de la representación D-aria de $F(x)$ para un símbolo cualquiera de la fuente x, que, sin pérdida de generalidad, se puede suponer perteneciente al conjunto $\mathcal{X} = \{1, 2, \ldots, m\}$. En general, $F(x)$ es un número real expresable sólo por un número infinito de bits, de manera que habrá que usar un valor aproximado, es decir, con un cierto número finito de sus símbolos $l(x)$, dependiendo obviamente la precisión de la aproximación del número de símbolos empleados en la representación. Representemos por $F_{l(x)}(x)$ dicha aproximación a $F(x)$. Se tiene

entonces que

$$F(x) - F_{l(x)}(x) < \frac{1}{D^{l(x)}}.$$

Por otra parte, si $l(x) = \lceil \log_D \frac{1}{p(x)} \rceil + 1$, entonces

$$\frac{1}{D^{l(x)}} < \frac{1}{2}\big(F(x) - F(x-1)\big).$$

Se pone así de manifiesto que las representaciones aproximadas de $F(x)$ para los distintos valores de la variable x pertenecen a conjuntos disjuntos, lo que garantiza la univocidad de la decodificación.

Por otra parte, puesto que se usan con este procedimiento $\lceil \log_D \frac{1}{p(x)} \rceil + 1$ símbolos para representar el símbolo x de probabilidad $p(x)$, entonces la longitud de este código será

$$L = \sum_x \left(\left\lceil \log_D \frac{1}{p(x)} \right\rceil + 1 \right) p(x) < H(X) + 2.$$

Los detalles para el cálculo paso a paso de $F(x^n)$, así como en general todas las cuestiones prácticas de implementación del método exceden con mucho el ámbito del presente capítulo. El lector interesado puede consultar alguna obra especializada en técnicas de compresión de datos, como por ejemplo [58].

3.B.2. Codificación de fuente universal

Hasta ahora hemos supuesto que se conocían de antemano las características de la fuente; pero rara vez esto es así. Los métodos de codificación que se emplean cuando no se conocen a priori las características de la fuente se denominan *universales*.

Son ejemplos de este tipo de métodos versiones adaptativas (también llamadas dinámicas) de los métodos ya comentados: el método de Huffman y la codificación aritmética. En ambos casos, la idea esencial del método consiste en la estimación dinámica (a medida que se procesan los símbolos de los mensajes tanto en el codificador como en el codificador) de las características estadísticas de la fuente.

Pero el método más ampliamente usado de codificación universal (puede considerarse actualmente el método estándar para compresión de ficheros), el denominado método Lempel–Ziv, es de naturaleza totalmente diferente ya que no se basa en la estimación de las características de la fuente; está basado, en cambio, en el análisis gramatical de la secuencia de la fuente.

Existen varias versiones del método [38, 79, 81], ligeramente diferentes entre sí, pero la regla más sencilla de análisis es la siguiente: el siguiente término es el término más corto que todavía no se ha observado. Siendo así, el siguiente término estará compuesto por un término anterior (y podrá representarse simplemente mediante un número de índice que indicará su posición en una lista ordenada de términos —diccionario—) más un símbolo adicional. Los detalles de implementación se discuten en [73] y [9]. Sorprendentemente, pese a su sencillez y aun sin tener en cuenta las características estadísticas de la fuente, el método Lempel–Ziv codifica cualquier fuente estacionaria ergódica a su tasa de entropía, como se demuestra en [80, 15, 76].

CAPÍTULO 4

Información mutua y capacidad de canal

En este capítulo se introduce el concepto de *entropía condicional* de una fuente X como la cantidad de información que nos proporcionan los mensajes de dicha fuente una vez que ya se conoce el mensaje de otra fuente Y que pueda guardar alguna relación con la primera. Se verá que esta información no puede ser mayor que la que nos proporcionarían los mensajes de la fuente X si se ignorase el mensaje de la fuente Y.

La reducción en la cantidad de información que nos proporcionan los mensajes de una fuente X después de conocer los mensajes de otra fuente Y puede interpretase, en consecuencia, como la información que la fuente Y proporciona sobre la fuente X. Dicha información, que, como se verá, es exactamente la misma que la que la fuente X proporciona sobre la fuente Y, se denomina por este motivo *información mutua*.

El concepto de información mutua es útil para determinar la información que sería necesaria para conocer el mensaje que se ha enviado a través de un canal (fuente X), después de ver el mensaje que ha llegado al destino (fuente Y). Si no hiciese falta información adicional, significaría que sabríamos con certeza qué mensaje se envió, por lo que dicha transmisión habría sido absolutamente fiable. Por contra, cuanta más información adicional se requiriese, menos fiable habría sido la transmisión. Esta cantidad de información depende, obviamente, tanto de las características de los mensajes que se envían a través del canal (es decir, de las características de la fuente) como de las propias características del canal de transmisión. Pero veremos también que, si se consideran todas las posibles fuentes que pueden hacer uso del canal, existe un valor máximo para la información que se puede trans-

mitir, que es un parámetro característico del canal. Tal parámetro recibe el nombre de *capacidad* del canal. La capacidad de un canal es un parámetro fundamental del mismo, ya que, como se verá en el capítulo siguiente, la capacidad del canal es una medida de la cantidad de información que se puede transmitir fielmente a través del mismo por unidad de tiempo. Tal resultado, con seguridad el más trascendental de la Teoría de la Información, se conoce habitualmente como teorema de Shannon de codificación de canales ruidosos.

Este capítulo se dedicará enteramente a las definiciones de entropía condicional, información mutua y capacidad, así como también al análisis de sus propiedades fundamentales.

4.1. Entropía condicional

Cuando se consideran dos variables aleatorias, X e Y, el hecho de conocer que $Y = y_j$ puede modificar las probabilidades a priori de los distintos valores de la variable aleatoria X.

Representemos por $p(x_i \,|\, y_j) = \mathrm{P}(X = x_i \,|\, Y = y_j)$ los distintos valores de la distribución de probabilidades a posteriori de la variable aleatoria X; y representemos por $H(X \,|\, Y = y_j)$ la entropía de dicha distribución de probabilidades. Entonces, esta entropía será:

$$H(X \,|\, Y = y_j) = \sum_{i=1}^{n} p(x_i \,|\, y_j) \log \frac{1}{p(x_i \,|\, y_j)}.$$

Así, para cada valor y_j de la variable aleatoria Y, se tiene una entropía $H(X \,|\, Y = y_j)$, por lo que tal entropía puede considerarse una función de la variable aleatoria Y, siendo su valor medio

$$\sum_{j=1}^{m} p(y_j) H(X \,|\, Y = y_j).$$

Tal valor medio recibe la denominación de entropía condicional.

Definición 4.1 (Entropía condicional). *Se define la entropía condicional de la variable aleatoria X, dada Y, como*

$$\begin{aligned} H(X \,|\, Y) &= \sum_{j=1}^{m} p(y_j) H(X \,|\, Y = y_j) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) \log \frac{1}{p(x_i \,|\, y_j)}. \end{aligned}$$

Ejemplo 4.1. Sea (X, Y) una variable aleatoria bidimensional con la siguiente distribución conjunta:

$Y \setminus X$	1	2	3	4
1	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{32}$
2	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{32}$	$\frac{1}{32}$
3	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
4	$\frac{1}{4}$	0	0	0

La distribución marginal de la variable X es $(1/2, 1/4, 1/8, 1/8)$, y la de la variable Y es $(1/4, 1/4, 1/4, 1/4)$. Por lo tanto, se tiene que

$$\begin{aligned} H_2(X) &= 7/4 \text{ bits}, \\ H_2(Y) &= 2 \text{ bits}, \\ H_2(X, Y) &= 27/8 \text{ bits}. \end{aligned}$$

Además,

$$\begin{aligned} H_2(X \,|\, Y) &= \sum_{i=1}^{4} \mathrm{P}(Y = i) H_2(X \,|\, Y = i) \\ &= \frac{1}{4} H_2\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right) + \frac{1}{4} H_2\left(\frac{1}{4}, \frac{1}{2}, \frac{1}{8}, \frac{1}{8}\right) \\ &+ \frac{1}{4} H_2\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right) + \frac{1}{4} H_2\left(1, 0, 0, 0\right) \\ &= \frac{1}{4} \times \frac{7}{4} + \frac{1}{4} \times \frac{7}{4} + \frac{1}{4} \times 2 + \frac{1}{4} \times 0 \\ &= \frac{11}{8} \text{ bits} \end{aligned}$$

$$\begin{aligned} H_2(Y \,|\, X) &= \sum_{i=1}^{4} \mathrm{P}(X = i) H_2(Y \,|\, X = i) \\ &= \frac{1}{2} H_2\left(\frac{1}{4}, \frac{1}{8}, \frac{1}{8}, \frac{1}{2}\right) + \frac{1}{4} H_2\left(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}, 0\right) \\ &+ \frac{1}{8} H_2\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{2}, 0\right) + \frac{1}{8} H_2\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{2}, 0\right) \\ &= \frac{1}{2} \times \frac{7}{4} + \frac{1}{4} \times \frac{3}{2} + \frac{1}{8} \times \frac{3}{2} + \frac{1}{8} \times \frac{3}{2} \\ &= \frac{13}{8} \text{ bits}. \end{aligned}$$

■

4.1.1. Propiedades de la entropía condicional

De acuerdo con la definición anterior, la entropía condicional es un promedio de entropías, por lo que adopta siempre un valor no negativo, que será nulo únicamente en el caso de que todas las entropías que se promedian lo sean, cosa que sólo puede ocurrir si el valor que adopta la variable aleatoria X depende funcionalmente del valor que adopta la variable aleatoria Y.

Obsérvese además, en el ejemplo anterior, que, en general, $H(X \mid Y) \neq H(Y \mid X)$. Sin embargo, se cumple el siguiente teorema:

TEOREMA 4.1 (REGLA DE LA CADENA).

$$H(X) + H(Y \mid X) = H(Y) + H(X \mid Y) = H(X, Y).$$

DEMOSTRACIÓN.

$$\begin{aligned} H(Y) + H(X \mid Y) &= \sum_j p(y_j) \log \frac{1}{p(y_j)} + \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(x_i \mid y_j)} \\ &= \sum_i \sum_j p(x_i, y_j) \left(\log \frac{1}{p(y_j)} + \log \frac{1}{p(x_i \mid y_j)} \right) \\ &= \sum_i \sum_j p(x_i, y_j) \left(\log \frac{1}{p(y_j) \cdot p(x_i \mid y_j)} \right) \\ &= \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(x_i, y_j)} \\ &= H(X, Y). \end{aligned}$$

De manera análoga se probaría que $H(X) + H(Y \mid X) = H(X, Y)$. ▶

Asimismo, modificando ligeramente la demostración anterior, se demuestra fácilmente el siguiente corolario:

COROLARIO 4.2. $H(X, Y \mid Z) = H(X \mid Z) + H(Y \mid X, Z)$.

Siendo $H(X, Y)$ la información que proporcionan las variables aleatorias X e Y conjuntamente, y siendo $H(Y)$ la información que proporciona la variable aleatoria Y por sí sola, la propiedad anterior permite interpretar $H(X \mid Y)$ como la información suministrada por la variable aleatoria X si ya se conoce el resultado de la variable aleatoria Y. A continuación veremos también que, como parece que debería ocurrir, esta información no puede ser mayor que $H(X)$; es decir, que nunca los mensajes de otra fuente Y

pueden aumentar nuestra incertidumbre acerca de una fuente cualquiera dada X.

Teorema 4.3. *Se cumple en general que*

$$H(X \mid Y) \leq H(X)$$

siendo condición necesaria y suficiente que las variables aleatorias X e Y sean independientes entre sí para que se cumpla la igualdad.

Demostración. La desigualdad se sigue inmediatamente de la regla de la cadena:

$$H(X, Y) = H(Y) + H(X \mid Y)$$

y de la relación entre las entropías individuales y la conjunta:

$$H(X, Y) \leq H(X) + H(Y).$$

Por otra parte, se cumplirá la igualdad si y sólo si se cumple la igualdad en la desigualdad anterior, cosa que, como ya se sabe, ocurre si y sólo si las variables aleatorias X e Y son independientes entre sí. ▶

4.1.2. Desigualdad de Fano

Un caso en el que aparecen dos fenómenos aleatorios relacionados es precisamente el de la transmisión de mensajes digitales a través de un canal discreto. En tal caso, se suele representar por X la aleatoriedad de los mensajes que se introducen en el canal y por Y la de los mensajes que salen del mismo. La variable aleatoria X depende únicamente de la fuente de los mensajes, pero la variable aleatoria Y depende tanto de la fuente como del canal. Precisamente, el rango de Y coincidirá con el alfabeto de salida del canal; y su función de probabilidad, $p_Y(y_j) = \mathrm{P}(Y = y_j)$, dependerá tanto de la función de probabilidad de X, $p_X(x_i) = \mathrm{P}(X = x_i)$, como de la matriz de probabilidades de transición del canal:

$$p(y_j) = \sum_i p(x_i, y_j) = \sum_i p(y_j \mid x_i) p(x_i).$$

Si el canal fuese ideal, es decir, si la entrada pudiese deducirse con certeza de la salida, se verificaría que

$$H(X \mid Y) = H(Y \mid X) = 0.$$

Por contra, si el canal no fuese ideal, se tendría necesariamente que $H(X \mid Y) > 0$. En tal caso, aun después de conocido el mensaje que sale

del canal, no se podría conocer con absoluta certeza qué mensaje habría enviado la fuente, siendo $H(X \mid Y)$ la medida de la cantidad de información adicional que sería precisa para averiguarlo. Obviamente, al no conocer con certeza el mensaje enviado por la fuente, habrá una cierta probabilidad de cometer un fallo al deducir el mensaje enviado por la fuente del obtenido a la salida del canal; y dicha probabilidad de error debería ser tanto más pequeña cuanto menor fuese el valor de $H(X \mid Y)$. La desigualdad de Fano cuantifica esta idea.

TEOREMA 4.4 (DESIGUALDAD DE FANO). *Sean X e Y dos variables aleatorias con el mismo rango, siendo n el número de elementos del mismo. Sea $\mathrm{P}_e = \mathrm{P}(Y \neq X)$. Se cumple que*

$$H(X \mid Y) \leq H(\mathrm{P}_e) + \mathrm{P}_e \log(n-1).$$

DEMOSTRACIÓN. Definamos una variable aleatoria error:

$$E = \begin{cases} 0 & \text{si } Y = X \\ 1 & \text{si } Y \neq X \end{cases}$$

De acuerdo con la regla de la cadena, se tiene:

- por un lado:

$$\begin{aligned} H(E, X \mid Y) &= H(X \mid Y) + H(E \mid X, Y) \\ &= H(X \mid Y) \end{aligned}$$

  ya que la incertidumbre sobre el error es nula si se conocen los valores de X e Y;

- y por otro lado:

$$\begin{aligned} H(E, X \mid Y) &= H(E \mid Y) + H(X \mid E, Y) \\ &\leq H(\mathrm{P}_e) + \mathrm{P}_e \log(n-1) \end{aligned}$$

  por las dos desigualdades siguientes:

$$\begin{aligned} H(E \mid Y) &\leq H(E) = H(\mathrm{P}_e) \\ H(X \mid E, Y) &= \mathrm{P}(E=0) H(X \mid E=0, Y) + \mathrm{P}(E=1) H(X \mid E=1, Y) \\ &\leq (1 - \mathrm{P}_e) 0 + \mathrm{P}_e \log(n-1). \end{aligned}$$

  Obsérvese que $H(X \mid E=1, Y) \leq \log(n-1)$ puesto que, al ser $E=1$, los posibles valores de la variable X son todos los que difieren del valor de Y. ▶

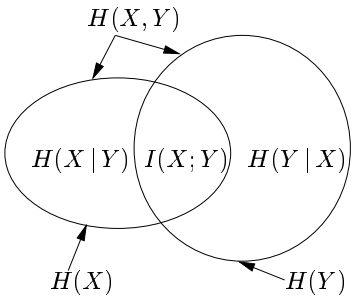


Figura 4.1. Relación entre la información mutua y la entropía.

La desigualdad de Fano admite una formulación intuitiva simple utilizando la propiedad de partición: la información necesaria para determinar el valor de X, una vez conocido el de Y, se puede descomponer en dos partes: la primera sería la información necesaria para saber si ha habido o no error; y la segunda (únicamente necesaria si ha habido error) sería la información necesaria para determinar cuál de las $n-1$ posibles alternativas de X restantes ha ocurrido, que dependerá obviamente de las probabilidades de estas alternativas, pero que está acotada superiormente por $\log(n-1)$.

4.2. Información mutua

Sean X e Y dos variables aleatorias discretas, tal y como hemos venido considerando hasta ahora. Según se ha visto, $H(X)$ es una medida de nuestra incertidumbre a priori acerca del resultado del experimento aleatorio representado por la variable aleatoria X; y $H(X\,|\,Y)$ es una medida de la incertidumbre a posteriori.

Tal y como se ha visto también en la sección anterior, se cumple que $H(X\,|\,Y) \leq H(X)$, y por tanto la diferencia $H(X) - H(X\,|\,Y)$ será la reducción de la incertidumbre debida al conocimiento del resultado del experimento representado por la variable aleatoria Y. Se dice por eso que $H(X) - H(X\,|\,Y)$ es la información que la variable aleatoria Y nos proporciona sobre la variable aleatoria X.

Pero, además, resulta que:

$$\begin{aligned} H(X) - H(X\,|\,Y) &= H(X) + H(Y) - \big(H(Y) + H(X\,|\,Y)\big) \\ &= H(X) + H(Y) - \big(H(X) + H(Y\,|\,X)\big) \\ &= H(Y) - H(Y\,|\,X) \end{aligned}$$

obteniéndose la primera igualdad sin más que sumar y restar $H(Y)$, y la segunda de la aplicación de la regla de la cadena.

Es decir, X da tanta información sobre Y como la que Y da sobre X; y por ese motivo esa información se denomina información mutua.

Definición 4.2 (Información mutua). *Dadas dos variables aleatorias discretas, X e Y, se define la información mutua $I(X;Y)$ como*

$$I(X;Y) = H(X) - H(X \mid Y).$$

Ejemplo 4.2. La información mutua entre las variables aleatorias definidas por la distribución conjunta del ejemplo 4.1 es

$$I(X;Y) = H(X) - H(X \mid Y) = \frac{7}{4} - \frac{11}{8} = \frac{3}{8} \text{ bits.} \qquad \blacksquare$$

4.2.1. Propiedades de la información mutua

De acuerdo con la definición dada, se tiene que

$$\begin{aligned} I(X;Y) &= \sum_i p(x_i) \log \frac{1}{p(x_i)} - \sum_i \sum_j p(x_i, y_j) \log \frac{1}{p(x_i \mid y_j)} \\ &= \sum_i \sum_j p(x_i, y_j) \left(\log \frac{1}{p(x_i)} - \log \frac{1}{p(x_i \mid y_j)} \right) \\ &= \sum_i \sum_j p(x_i, y_j) \log \frac{p(x_i \mid y_j)}{p(x_i)} \\ &= \mathrm{E}\left[\log \frac{p(x_i \mid y_j)}{p(x_i)} \right] \end{aligned}$$

donde queda puesto de manifiesto explícitamente que la información mutua depende tanto de la distribución de las probabilidades a priori como de la distribución de las probabilidades a posteriori de una cualquiera de las variables aleatorias.

Es fácil comprobar que esta función, al igual que la entropía, tiene las siguientes características:

- nunca es negativa,
- tiene una cota superior, y
- es convexa con respecto a la distribución de las probabilidades a priori (cóncava con respecto a la distribución de las probabilidades a posteriori).

Teorema 4.5. *El valor de $I(X;Y)$ es o positivo o nulo, siendo condición necesaria y suficiente para que $I(X;Y) = 0$ que las variables aleatorias X e Y sean independientes entre sí.*

Demostración. Es $I(X;Y) \geq 0$ ya que $H(X \mid Y) \leq H(X)$.

Por otra parte, por definición, $I(X;Y) = 0$ si y sólo si $H(X) = H(X \mid Y)$, lo cual a su vez es cierto si y sólo si las variables aleatorias X e Y son independientes entre sí. ▶

Teorema 4.6.

$$I(X;Y) \leq \min\big(H(X), H(Y)\big).$$

Demostración. Se sigue inmediatamente de las siguientes desigualdades:

$$\begin{aligned} I(X;Y) &= H(X) - H(X \mid Y) \leq H(X) \\ I(X;Y) &= H(Y) - H(Y \mid X) \leq H(Y). \end{aligned}$$ ▶

Teorema 4.7. *$I(X;Y)$ es convexa con respecto a la distribución de probabilidad $(p(x_1), p(x_2), \ldots, p(x_n))$.*

Demostración.

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y \mid X) \\ &= H(Y) - \sum_i p(x_i) H(Y \mid X = x_i) \\ &= H(Y) - \sum_i p(x_i) \sum_j p(y_j \mid x_i) \log \frac{1}{p(y_j \mid x_i)}. \end{aligned}$$

Si las probabilidades $p(y_j \mid x_i)$ son fijas, entonces las probabilidades $p(y_j)$ son funciones lineales de las probabilidades $p(x_i)$, ya que

$$p(y_j) = \sum_i p(x_i, y_j) = \sum_i p(y_j \mid x_i) p(x_i).$$

Se tiene pues que:

- $H(Y)$ es una función convexa con respecto a la distribución de probabilidades $(p(x_1), p(x_2), \ldots, p(x_n))$, ya que lo es con respecto a la distribución de probabilidades $(p(y_1), p(y_2), \ldots, p(y_m))$;
- $\sum_i p(x_i) H(Y \mid X = x_i)$ es una función lineal de las probabilidades $p(x_i)$,

y la diferencia de una función convexa y una función lineal resulta ser una función convexa. ▶

Esta propiedad resulta muy importante ya que, en virtud de la convexidad de $I(X;Y)$ con respecto a $\mathbf{p}$, $I(X;Y)$ tiene un máximo.

TEOREMA 4.8. *$I(X;Y)$ es cóncava con respecto a las distribuciones de probabilidad $(p(y_1 \mid x_i), p(y_2 \mid x_i), \ldots, p(y_m \mid x_i))$.*

DEMOSTRACIÓN. Dadas dos familias de distribuciones de probabilidad condicionales $(p_1(y_j \mid x_i))$ y $p_2(y_j \mid x_i))$, y dado un número real λ tal que $0 < \lambda < 1$, se definen las siguientes probabilidades condicionales:

$$p_\lambda(y_j \mid x_i) = \lambda p_1(y_j \mid x_i) + (1-\lambda) p_2(y_j \mid x_i).$$

Tales probabilidades caracterizan a una nueva variable aleatoria, Y_λ, de distribución marginal dada por las probabilidades:

$$\begin{aligned} p_\lambda(y_j) &= \sum_i p(x_i) p_\lambda(y_j \mid x_i) \\ &= \lambda \sum_i p_1(y_j \mid x_i) p(x_i) + (1-\lambda) \sum_i p_2(y_j \mid x_i) p(x_i) \\ &= \lambda p_1(y_j) + (1-\lambda) p_2(y_j). \end{aligned}$$

Si ahora se calcula la información mutua entre esta nueva variable aleatoria Y_λ y la variable aleatoria X, se tiene que

$$\begin{aligned} I(X;Y_\lambda) &= \sum_i p(x_i) \sum_j p_\lambda(y_j \mid x_i) \log \frac{p_\lambda(y_j \mid x_i)}{p_\lambda(y_j)} \\ &= \sum_i p(x_i) \sum_j p_\lambda(y_j \mid x_i) \log \frac{\lambda p_1(y_j \mid x_i) + (1-\lambda) p_2(y_j \mid x_i)}{\lambda p_1(y_j) + (1-\lambda) p_2(y_j)}. \end{aligned}$$

Pero como $f(x) = x \log x$ es una función cóncava en x:

$$\begin{aligned} &\sum_j p_\lambda(y_j \mid x_i) \log \frac{\lambda p_1(y_j) + (1-\lambda) p_2(y_j)}{\lambda p_1(y_j \mid x_i) + (1-\lambda) p_2(y_j \mid x_i)} \\ &\leq \lambda \sum_j p_1(y_j \mid x_i) \log \frac{p_1(y_j \mid x_i)}{p_1(y_j)} + (1-\lambda) \sum_j p_2(y_j \mid x_i) \log \frac{p_2(y_j \mid x_i)}{p_2(y_j)}. \end{aligned}$$

Aplicando esta desigualdad a cada sumando de $I(X;Y_\lambda)$, resulta:

$$I(X;Y_\lambda) \leq \lambda I(X;Y_1) + (1-\lambda) I(X;Y_2).$$ ▶

4.3. Información mutua condicional

El hecho de conocer una tercera variable aleatoria Z puede hacer variar la distribución marginal o conjunta de dos variables aleatorias dadas, X e Y.

Definición 4.3. *Sean X, Y y Z tres variables aleatorias discretas. Se dice que X e Y son condicionalmente independientes dado Z si la distribución condicional de Y, cuando se conoce Z, es independiente de la de X:*

$$\mathrm{P}(X = x, Y = y \,|\, Z = z) = \mathrm{P}(X = x \,|\, Z = z)\,\mathrm{P}(Y = y \,|\, Z = z).$$

Definición 4.4 (Información mutua condicional). *La información mutua condicional de las variables aleatorias X e Y, dada la variable Z, es*

$$I(X;Y \,|\, Z) = H(X \,|\, Z) - H(X \,|\, Y, Z).$$

Se puede verificar fácilmente que la información mutua condicional conserva todas las propiedades de la información mutua. En particular, se cumple el siguiente teorema.

Teorema 4.9.

$$I(X;Y \,|\, Z) \geq 0$$

con igualdad estricta si y solamente si, dada Z, las variables X e Y son condicionalmente independientes.

Además, la información mutua satisface también una regla de la cadena:

Teorema 4.10.

$$I(X,Y;Z) = I(X;Z) + I(Y;Z \,|\, X).$$

Demostración.

$$\begin{aligned} I(X,Y;Z) &= H(X,Y) - H(X,Y \,|\, Z) \\ &= H(X) + H(Y \,|\, X) - \big(H(X \,|\, Z) + H(Y \,|\, X,Z)\big) \\ &= \big(H(X) - H(X \,|\, Z)\big) + \big(H(Y \,|\, X) - H(Y \,|\, X,Z)\big) \\ &= I(X;Z) + I(Y;Z \,|\, X). \end{aligned}$$

▶

4.4. El teorema de procesamiento de la información

Dado que la información mutua es una medida de la cantidad de información que una variable, Y, aporta sobre otra, X, cabría preguntarse: ¿existe alguna transformación de Y que nos proporcione mayor cantidad de información sobre X que $I(X;Y)$? El teorema de procesamiento de la información niega tal posibilidad: ninguna manipulación determinista o aleatoria de las observaciones incrementa la cantidad de información de éstas.

TEOREMA 4.11. *Sea Z el resultado de cualquier transformación de Y que sea condicionalmente independiente de X. Entonces*

$$I(X;Z) \leq I(X;Y).$$

DEMOSTRACIÓN.

$$I(X;Y,Z) = I(X;Z) + I(X;Y \,|\, Z) = I(X;Y) + I(X;Z \,|\, Y).$$

Pero, como X y Z son condicionalmente independientes, $I(X;Z \,|\, Y) = 0$; y como $I(X;Y \,|\, Z) \geq 0$, se tiene que

$$I(X;Z) \leq I(X;Y). \qquad \blacktriangleright$$

La inecuación es simétrica con respecto a X y Z, y del mismo modo se prueba que $I(Y;Z) \geq I(X;Z)$.

4.5. Información mutua entre vectores aleatorios

La definición de información mutua se puede aplicar, al igual que la definición de entropía, al caso más general de variables aleatorias discretas n-dimensionales, también denominadas vectores aleatorios discretos de dimensión n. Y de la misma manera que existe una relación entre las entropías individuales y la conjunta, se puede probar que existen ciertas relaciones entre la información mutua entre vectores aleatorios y las informaciones mutuas entre las componentes de los vectores.

Sean pues $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ e $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_n)$ dos vectores aleatorios discretos de dimensión n. Se tiene que la información mutua entre ellos está dada por:

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= \mathrm{E}\left[\log \frac{p(\mathbf{x} \,|\, \mathbf{y})}{p(\mathbf{x})}\right] \\ &= \mathrm{E}\left[\log \frac{p(\mathbf{y} \,|\, \mathbf{x})}{p(\mathbf{y})}\right]. \end{aligned}$$

Teorema 4.12. *Si se verifica que*

$$\mathrm{P}\big((y_1, y_2, \ldots, y_n) \,|\, (x_1, x_2, \ldots, x_n)\big) = \prod_{i=1}^{n} p(y_i \,|\, x_i),$$

entonces se cumple que

$$I(\mathbf{X}; \mathbf{Y}) \leq \sum_{i=1}^{n} I(X_i; Y_i).$$

Demostración.

$$I(\mathbf{X}; \mathbf{Y}) = \mathrm{E}\left[\log \frac{p(\mathbf{y} \,|\, \mathbf{x})}{p(\mathbf{y})}\right] = \mathrm{E}\left[\log \frac{\prod_{i=1}^{n} p(y_i \,|\, x_i)}{p(\mathbf{y})}\right]$$

donde la primera igualdad se obtiene de la definición de información mutua y la segunda se sigue inmediatamente de la hipótesis del teorema.

Por otra parte,

$$\begin{aligned}
\sum_{i=1}^{n} I(X_i; Y_i) &= \sum_{i=1}^{n} \mathrm{E}\left[\log \frac{p(y_i \,|\, x_i)}{p(y_i)}\right] \\
&= \mathrm{E}\left[\sum_{i=1}^{n} \log \frac{p(y_i \,|\, x_i)}{p(y_i)}\right] \\
&= \mathrm{E}\left[\log \frac{\prod_{i=1}^{n} p(y_i \,|\, x_i)}{\prod_{i=1}^{n} p(y_i)}\right].
\end{aligned}$$

Entonces, se tiene que

$$\begin{aligned}
I(\mathbf{X}; \mathbf{Y}) - \sum_{i=1}^{n} I(X_i; Y_i) &= \mathrm{E}\left[\log \frac{\prod_{i=1}^{n} p(y_i)}{p(\mathbf{y})}\right] \\
&\leq \log \mathrm{E}\left[\frac{\prod_{i=1}^{n} p(y_i)}{p(\mathbf{y})}\right] \\
&= 0
\end{aligned}$$

siguiéndose la desigualdad de la demostración de la desigualdad de Jensen; y la última igualdad, del hecho de que se cumple que

$$\mathrm{E}\left[\frac{\prod_{i=1}^{n} p(y_i)}{p(\mathbf{y})}\right] = \sum_{\mathbf{y}} p(\mathbf{y}) \frac{\prod_{i=1}^{n} p(y_i)}{p(\mathbf{y})} = \sum_{\mathbf{y}} \prod_{i=1}^{n} p(y_i) = 1. \qquad \blacktriangleright$$

Si se considera que $(Y_1, Y_2, \ldots, Y_n)$ representa las n salidas de un canal correspondientes a las entradas $(X_1, X_2, \ldots, X_n)$, este teorema dice que, cuando el canal no tiene memoria, $\mathbf{Y}$ proporciona sobre $\mathbf{X}$ una cantidad de información que no puede ser mayor que la cantidad total de información suministrada acerca de cada X_i por la correspondiente Y_i.

Existen, sin embargo, unas condiciones en las que ocurre todo lo contrario, tal y como establece el siguiente teorema.

TEOREMA 4.13. *Si se verifica que las componentes del vector aleatorio* $\mathbf{X}$ *son independientes entre sí, entonces se cumple que*

$$I(\mathbf{X};\mathbf{Y}) \geq \sum_{i=1}^{n} I(X_i;Y_i).$$

DEMOSTRACIÓN.

$$I(\mathbf{X};\mathbf{Y}) = \mathrm{E}\left[\log \frac{p(\mathbf{x}\,|\,\mathbf{y})}{p(\mathbf{x})}\right] = \mathrm{E}\left[\log \frac{p(\mathbf{x}\,|\,\mathbf{y})}{\prod_{i=1}^{n} p(x_i)}\right]$$

donde la primera igualdad se obtiene de la definición de información mutua y la segunda se sigue inmediatamente de la hipótesis del teorema.

Por otra parte,

$$\begin{aligned}\sum_{i=1}^{n} I(X_i;Y_i) &= \sum_{i=1}^{n} \mathrm{E}\left[\log \frac{p(x_i\,|\,y_i)}{p(x_i)}\right] \\ &= \mathrm{E}\left[\sum_{i=1}^{n} \log \frac{p(x_i\,|\,y_i)}{p(x_i)}\right] \\ &= \mathrm{E}\left[\log \frac{\prod_{i=1}^{n} p(x_i\,|\,y_i)}{\prod_{i=1}^{n} p(x_i)}\right].\end{aligned}$$

Entonces, se tiene que

$$\begin{aligned}\sum_{i=1}^{n} I(X_i;Y_i) - I(\mathbf{X};\mathbf{Y}) &= \mathrm{E}\left[\log \frac{\prod_{i=1}^{n} p(x_i\,|\,y_i)}{p(\mathbf{x}\,|\,\mathbf{y})}\right] \\ &\leq \log \mathrm{E}\left[\frac{\prod_{i=1}^{n} p(x_i\,|\,y_i)}{p(\mathbf{x}\,|\,\mathbf{y})}\right] \\ &= 0\end{aligned}$$

siguiéndose la desigualdad de la demostración de la desigualdad de Jensen;

y la última igualdad, del hecho de que se cumple que

$$\begin{aligned}
\mathrm{E}\left[\frac{\prod_{i=1}^{n} p(x_i \,|\, y_i)}{p(\mathbf{x}|\mathbf{y})}\right] &= \sum_{\mathbf{x}} \sum_{\mathbf{y}} p(\mathbf{x},\mathbf{y}) \frac{\prod_{i=1}^{n} p(x_i \,|\, y_i)}{p(\mathbf{x}\,|\,\mathbf{y})} \\
&= \sum_{\mathbf{x}} \sum_{\mathbf{y}} p(\mathbf{y}) \prod_{i=1}^{n} p(x_i \,|\, y_i) \\
&= \sum_{\mathbf{y}} p(\mathbf{y}) \sum_{\mathbf{x}} \prod_{i=1}^{n} p(x_i \,|\, y_i) \\
&= \sum_{\mathbf{y}} p(\mathbf{y}) \\
&= 1.
\end{aligned}$$

▶

Como consecuencia inmediata de los dos teoremas anteriores se tiene el siguiente

Corolario 4.14. *Es condición suficiente para que se cumpla*

$$I(\mathbf{X};\mathbf{Y}) = \sum_{i=1}^{n} I(X_i;Y_i)$$

que se verifiquen simultáneamente las dos siguientes condiciones:

a) las componentes del vector aleatorio $\mathbf{X}$ *son independientes entre sí;*

b) $\mathrm{P}\big((y_1, y_2, \ldots, y_n) \,|\, (x_1, x_2, \ldots, x_n)\big) = \prod_{i=1}^{n} p(y_i \,|\, x_i)$.

Resulta también interesante notar que estas dos condiciones son también condiciones necesarias. Basta para ello verificar en los dos teoremas anteriores que, como consecuencia de la desigualdad de Gibbs, es condición necesaria y suficiente para que se cumpla la igualdad que:

- las componentes del vector aleatorio $\mathbf{Y}$ sean independientes entre sí, en el teorema 4.12;
- $\mathrm{P}\,(\mathbf{x}\,|\,\mathbf{y}) = \prod_{i=1}^{n} p(x_i \,|\, y_i)$, en el teorema 4.13.

4.6. Capacidad de canal: concepto y propiedades

Como se recordará, las características de un canal discreto sin memoria se especifican mediante una matriz de probabilidades de transición

$$Q_{r\times s} = \begin{pmatrix} p(y_1 \,|\, x_1) & \cdots & p(y_j \,|\, x_1) & \cdots & p(y_s \,|\, x_1) \\ \vdots & & \vdots & & \vdots \\ p(y_1 \,|\, x_i) & \cdots & p(y_j \,|\, x_i) & \cdots & p(y_s \,|\, x_i) \\ \vdots & & \vdots & & \vdots \\ p(y_1 \,|\, x_r) & \cdots & p(y_j \,|\, x_r) & \cdots & p(y_s \,|\, x_r) \end{pmatrix}$$

donde $p(y_j \,|\, x_i)$ representa la probabilidad de que se obtenga el símbolo y_j a la salida del canal cuando se ha transmitido el símbolo x_i.

Para un canal dado, la información que los mensajes recibidos proporcionan sobre los enviados (que está dada por la información mutua $I(X;Y)$) dependerá de la naturaleza de los mensajes enviados, es decir, de la variable aleatoria X. Se sabe también que la información mutua $I(X;Y)$ es una función convexa de las probabilidades de la distribución de la variable aleatoria X, por lo que existe una distribución de las probabilidades de X para la que la información mutua alcanza su máximo valor. Tal valor máximo se denomina capacidad del canal.

Definición 4.5 (Capacidad del canal). *Sea X la variable aleatoria discreta que representa la introducción de un símbolo en el canal, y sea Y la variable aleatoria discreta que representa la salida de un símbolo del canal. Se define la capacidad del canal como el máximo valor posible que puede adoptar la información mutua entre las variables aleatorias X e Y cuando se consideran todas las posibles distribuciones de probabilidad de la variable aleatoria X:*

$$C = \max_{p(x)} I(X;Y).$$

Las propiedades de la capacidad del canal se deducen inmediatamente de las propiedades de la información mutua.

Teorema 4.15. $C \geq 0$.

Demostración. Se sigue inmediatamente del hecho de que $I(X;Y) \geq 0$. ▶

Teorema 4.16. $C \leq \max H(X)$.

Demostración. Se sigue inmediatamente de los dos resultados siguientes:

$$I(X;Y) = H(X) - H(X \mid Y)$$
$$H(X \mid Y) \geq 0. \qquad \blacktriangleright$$

En virtud del teorema anterior, habida cuenta de que $H(X) \leq \log r$, resulta que, expresando la información en unidades de base r, el valor de C no puede exceder de la unidad.

4.7. Cálculo de la capacidad

La obtención de la capacidad del canal es un problema de cálculo, en general complejo, pero para el que existen métodos de resolución bien conocidos. La exposición de tales métodos cae, no obstante, fuera del ámbito de este libro. Aun así, para muchos casos de interés práctico, el cálculo de la capacidad es una cuestión sencilla. Así sucede, por ejemplo, para todos los canales con alfabeto de entrada binario como el del ejemplo siguiente.

Ejemplo 4.3. Considérese el canal de matriz de probabilidades de transición

$$Q = \begin{pmatrix} 1-p & p \\ 0 & 1 \end{pmatrix}; \qquad p < \frac{1}{2}.$$

Sea $\alpha = \mathrm{P}(X = 1)$. En consecuencia, resultará que $\mathrm{P}(X = 0) = 1 - \alpha$. A partir de aquí resulta fácil establecer tanto la distribución de probabilidades conjunta de X e Y:

$$\begin{aligned} \mathrm{P}(X = 0, Y = 0) &= 1 - \alpha \\ \mathrm{P}(X = 1, Y = 0) &= \alpha p \\ \mathrm{P}(X = 0, Y = 1) &= 0 \\ \mathrm{P}(X = 1, Y = 1) &= \alpha(1-p) \end{aligned}$$

como la distribución marginal de Y:

$$\begin{aligned} \mathrm{P}(Y = 0) &= 1 - \alpha(1-p) \\ \mathrm{P}(Y = 1) &= \alpha(1-p). \end{aligned}$$

Así, resulta que

$$\begin{aligned} I(X;Y) &= \sum_i \sum_j p(x_i, y_j) \log \frac{p(x_i, y_j)}{p(x_i)p(y_j)} \\ &= (1-\alpha) \log \frac{1}{1-\alpha(1-p)} + \alpha p \log \frac{p}{1-\alpha(1-p)} + \alpha(1-p) \log \frac{1}{\alpha}. \end{aligned}$$

Se tiene entonces que, en este caso, $I(X;Y)$ es una función de una variable real, α, cuya maximización es trivial: se obtiene el valor de α que maximiza $I(X;Y)$, α_{max}, resolviendo la ecuación $\frac{\mathrm{d}I(X;Y)}{\mathrm{d}\alpha} = 0$. Así, resulta que:

$$\alpha_{\mathrm{máx}} = \frac{1}{1-p+p^{\frac{-p}{1-p}}}$$

y que

$$C = I(X;Y)|_{\alpha=\alpha_{\mathrm{máx}}} = \log\left[1+(1-p)p^{\frac{p}{1-p}}\right]. \qquad \blacksquare$$

Existen también, aun para canales de alfabeto de entrada no binario, casos particulares en los que la maximización se simplifica por aplicación de las propiedades de la información mutua.

Ejemplo 4.4. Considérese el canal de matriz de probabilidades de transición

$$Q_{M\times(M+1)} = \begin{pmatrix} 1-p & 0 & 0 & \cdots & 0 & p \\ 0 & 1-p & 0 & \cdots & 0 & p \\ \vdots & & & & & \\ 0 & 0 & 0 & \cdots & 1-p & p \end{pmatrix}; \qquad p < \frac{1}{2}.$$

Un canal de este tipo suele denominarse canal M-ario con borrado.

Sea $p_i = \mathrm{P}(X = i)$. En este caso particular sólo tienen probabilidad no nula $2M$ pares de los $M(M+1)$ pares posibles: son los pares de la forma (i,i) y los pares de la forma (i,M), para los que se tiene que

$$\begin{aligned} \mathrm{P}(X=i, Y=i) &= p_i(1-p) \\ \mathrm{P}(X=i, Y=M) &= p_i p \end{aligned}$$

de donde resulta la distribución marginal de Y

$$\begin{aligned} \mathrm{P}(Y=i) &= \mathrm{P}(X=i, Y=i) = p_i(1-p) \\ \mathrm{P}(Y=M) &= \sum_i \mathrm{P}(X=i, Y=M) = p. \end{aligned}$$

Así, resulta que

$$\begin{aligned} I(X;Y) &= \sum_i \sum_j p(x_i, y_j) \log \frac{p(x_i, y_j)}{p(x_i)p(y_j)} \\ &= \sum_{i=1}^{M} (1-p) p_i \log\left(\frac{1}{p_i}\right) \\ &= (1-p)H(X) \end{aligned}$$

que será máxima cuando lo sea $H(X)$. Pero de las propiedades de la entropía se sabe que el máximo de $H(X)$ se alcanza cuando la distribución de la variable aleatoria X es uniforme ($p_i = 1/M \quad \forall i$), y vale $H(X) = \log M$; por lo que, en este caso, se obtiene directamente la capacidad del canal:

$$C = (1 - p) \log M. \qquad \blacksquare$$

Vamos a considerar, por último, una serie de casos especiales de canales con propiedades bien conocidas.

4.7.1. Canales simétricos

Definición 4.6 (Canal simétrico). *Se dice que un canal discreto sin memoria es simétrico cuando su matriz de probabilidades de transición cumple las dos propiedades siguientes:*

- *todas sus filas tienen los mismos elementos, y*
- *todas sus columnas tienen los mismos elementos.*

Obsérvese que, en la definición anterior, no se exige que todas las filas sean iguales, pues los elementos de las mismas pueden disponerse en distinto orden. Lo mismo cabe decir de las columnas. Obsérvese, además, que filas y columnas no tienen por qué tener los mismos elementos (ni siquiera tienen por qué tener el mismo número de elementos).

Ejemplo 4.5. Un ejemplo de canal simétrico sería el siguiente:

$$Q = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}; \qquad p < \frac{1}{2}.$$

Este canal se denomina canal binario simétrico. $\blacksquare$

Ejemplo 4.6. Es también simétrico un canal de la forma

$$Y = X + Z \pmod K$$

donde los alfabetos de X y Z son ambos $\{0, 1, \ldots, K-1\}$, y donde Z es independiente de X. Resulta fácil verificar que tanto las filas como las columnas de la matriz de probabilidades de transición no son más que permutaciones de los elementos de la distribución de probabilidad de la variable aleatoria Z. $\blacksquare$

TEOREMA 4.17. *Un canal discreto sin memoria simétrico con r entradas y s salidas alcanza su capacidad cuando las entradas son equiprobables; y el valor de la capacidad es*

$$C = \log s - H(Y \mid X = x_i) \qquad \forall i.$$

DEMOSTRACIÓN. De acuerdo con la definición de información mutua,

$$I(X;Y) = H(Y) - H(Y \mid X).$$

Y, por la definición de entropía condicional,

$$\begin{aligned} H(Y \mid X) &= \sum_{i=1}^{r} p(x_i) H(Y \mid X = x_i) \\ &= H(Y \mid X = x_i) \sum_{i=1}^{r} p(x_i) \\ &= H(Y \mid X = x_i) \end{aligned}$$

ya que $H(Y \mid X = x_i)$ es constante $\forall i$ al ser la entropía de una cualquiera de las filas de la matriz de probabilidades de transición del canal.

Por tanto, se tiene que $H(Y \mid X) = H(Y \mid X = x_i)$, constante $\forall i$. Y así, por ser $H(Y \mid X)$ constante, $I(X;Y)$ será máxima cuando lo sea $H(Y)$.

Se sabe, por las propiedades de la entropía, que $H(Y) \leq \log s$, alcanzándose la igualdad únicamente en el caso de que los distintos valores de la variable aleatoria Y sean equiprobables. Así que habrá que preguntarse si los distintos valores de la variable aleatoria Y pueden ser equiprobables, es decir, si existe alguna distribución de la variable aleatoria X que conduzca a valores de Y equiprobables. Y es fácil verificar que sí existe tal distribución, y es la distribución uniforme; es decir, que, en un canal simétrico, los valores de Y son equiprobables cuando lo son los de la variable aleatoria X. En efecto:

$$\begin{aligned} p(y_j) &= \sum_{i=1}^{r} p(x_i, y_j) \\ &= \sum_{i=1}^{r} p(y_j \mid x_i) p(x_i) \\ &= p(x_i) \sum_{i=1}^{r} p(y_j \mid x_i) \end{aligned}$$

por ser $p(x_i)$ constante $\forall i$. Y como, por ser el canal simétrico, la suma $\sum_{i=1}^{r} p(y_j \mid x_i)$ es constante $\forall j$, entonces también $p(y_j)$ es constante $\forall j$. ▶

Si se examina cuidadosamente la demostración anterior, puede apreciarse que las condiciones de simetría de la definición que hemos dado para canal simétrico podrían relajarse un poco sin alterar el resultado que acabamos de obtener, a saber:

- bastaría que todas las filas de la matriz Q tuviesen la misma entropía; y
- bastaría que todas las columnas de la matriz Q sumasen lo mismo.

Por tal motivo estas dos condiciones podrían constituir una definición alternativa de la simetría del canal.

4.7.2. Yuxtaposición de canales

Puede darse el caso de que la matriz de probabilidades de transición de un canal sea de la forma

$$Q_{r\times s} = \begin{pmatrix} Q^1_{r'\times s'} & 0 \\ 0 & Q^2_{(r-r')\times(s-s')} \end{pmatrix}.$$

Un canal de este tipo puede considerarse formado por la yuxtaposición de dos canales independientes (con matrices de probabilidades de transición Q^1 y Q^2) que se usan aleatoriamente.

Teorema 4.18. *La capacidad de un canal formado por la yuxtaposición de dos subcanales de capacidades C_1 y C_2 (medidas ambas en bits) es*

$$C = \log_2\left(2^{C_1} + 2^{C_2}\right)$$

obteniéndose dicha capacidad cuando la probabilidad de uso del canal de capacidad C_1 está dada por

$$p = \frac{2^{C_1}}{2^{C_1} + 2^{C_2}}.$$

Demostración. Sean X e Y las variables aleatorias discretas que representan, respectivamente, los símbolos que se introducen en el canal y que salen del mismo en un momento dado. Considere que se pueden establecer particiones binarias en ambas variables aleatorias de forma que las entradas de un elemento de la partición de X sólo pueden dar salidas de un elemento de la partición de Y, y viceversa; y sean X_i e Y_i las variables aleatorias que representan sendos elementos de las particiones de X e Y que están en correspondencia (es decir, sean X_i e Y_i las variables aleatorias discretas que

representan, respectivamente, los símbolos que se introducen en el subcanal i-ésimo y que salen del mismo en un momento dado).

Por definición, la capacidad del canal será

$$C = \max_{p(x)} I(X;Y).$$

A su vez,

$$I(X;Y) = H(X) + H(Y) - H(X,Y).$$

Y como, en virtud de la propiedad de partición, se tiene:

$$\begin{aligned} H(X) &= H(p) + pH(X_1) + (1-p)H(X_2) \\ H(Y) &= H(p) + pH(Y_1) + (1-p)H(Y_2) \\ H(X,Y) &= H(p) + pH(X_1,Y_1) + (1-p)H(X_2,Y_2) \end{aligned}$$

entonces resulta que

$$I(X;Y) = H(p) + pI(X_1;Y_1) + (1-p)I(X_2;Y_2).$$

Finalmente, por ser la maximización de las $I(X_i;Y_i)$ independiente del valor de p, se tiene que

$$\max I(X;Y) = \max \left(H(p) + pC_1 + (1-p)C_2\right)$$

y derivando esta expresión con respecto a p e igualando a 0 se obtiene el valor de p que maximiza la información mutua:

$$p = \frac{2^{C_1}}{2^{C_1} + 2^{C_2}}.$$

Así, resulta que la capacidad C vale

$$\begin{aligned} & H\left(\frac{2^{C_1}}{2^{C_1}+2^{C_2}}\right) + \frac{2^{C_1}}{2^{C_1}+2^{C_2}}C_1 + \frac{2^{C_2}}{2^{C_1}+2^{C_2}}C_2 \\ &= \frac{2^{C_1}}{2^{C_1}+2^{C_2}} \log_2 \frac{2^{C_1}+2^{C_2}}{2^{C_1}} + \frac{2^{C_2}}{2^{C_1}+2^{C_2}} \log_2 \frac{2^{C_1}+2^{C_2}}{2^{C_2}} + \frac{C_1 2^{C_1} + C_2 2^{C_2}}{2^{C_1}+2^{C_2}} \\ &= \log_2 \left(2^{C_1} + 2^{C_2}\right). \end{aligned}$$

▶

Ejemplo 4.7.

$$Q_{4\times 4} = \begin{pmatrix} 1-p & p & 0 & 0 \\ p & 1-p & 0 & 0 \\ 0 & 0 & 1-p & p \\ 0 & 0 & p & 1-p \end{pmatrix}; \qquad p < \frac{1}{2}.$$

En este caso, $C_1 = C_2 = 1 - H_2(p)$ bits; y por tanto $C = 2 - H_2(p)$ bits, obteniéndose dicha capacidad para entradas equiprobables. ■

4.7.3. Canales en serie

Cabe la posibilidad de concatenar varios canales para formar un nuevo canal. Consideraremos aquí únicamente la concatenación de dos canales de forma que la salida del primero se introduzca directamente en el segundo, sin ningún procesamiento intermedio. Será preciso, entonces, que ambos canales tengan idénticas tasas de transferencia de datos, y que el alfabeto de entrada del segundo canal sea idéntico al alfabeto de salida del primero.

Resulta evidente que tal concatenación de canales producirá los mismos resultados que un único canal con un alfabeto de entrada igual al del primer canal y un alfabeto de salida igual al del segundo con tal de que las probabilidades de transición del canal único sean las mismas que las obtenidas extremo a extremo para los canales concatenados. Diremos que un tal canal único es el canal equivalente a la concatenación de los dos canales dados. Y es inmediato verificar que la matriz de probabilidades de transición del canal equivalente no es más que el producto de las matrices de probabilidades de transición de los canales que se concatenan:

$$Q^e = Q^1 \times Q^2$$

siendo Q^e la matriz del canal equivalente, Q^1 la matriz del primero de los canales que se concatenan y Q^2 la matriz del segundo.

Ejemplo 4.8. Considere dos canales binarios simétricos idénticos. En este caso, se tiene que

$$Q^1 = Q^2 = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}; \qquad p < \frac{1}{2}$$

y el canal equivalente de la concatenación de los dos canales dados tendrá la siguiente matriz de probabilidades de transición:

$$Q^e = \begin{pmatrix} p^2+(1-p)^2 & 2p(1-p) \\ 2p(1-p) & p^2+(1-p)^2 \end{pmatrix} = Q^1 \times Q^2.$$

Es decir, el canal equivalente es otro canal binario simétrico, pero con probabilidad de error $2p(1-p)$. ■

Notas bibliográficas

La idea de la información mutua y su relación con la capacidad del canal fue desarrollada por Shannon en [59] y [60]. La desigualdad de Fano, en la que se basa la demostración del teorema inverso de codificación de canales ruidosos, que se verá en el capítulo siguiente, apareció publicada en [17].

Un libro ya clásico de Teoría de la Información, recomendable por su legibilidad, en el que los tópicos de este tema se tratan con cierta amplitud es [1].

Los conceptos de este capítulo se pueden extender para el caso de variables y vectores aleatorios continuos, si bien un tratamiento riguroso de esta materia es difícil. El tratamiento más completo de la misma se encuentra en los tres primeros capítulos de [50].

CAPÍTULO 5

El teorema de codificación de canales ruidosos

5.1. Introducción

Este tema está dedicado por entero al establecimiento e interpretación del que es sin lugar a dudas el resultado central de la Teoría de la Información: el llamado teorema de Shannon de codificación de canales ruidosos. De forma sintética, este teorema establece la posibilidad de establecer una comunicación fiable a través de cualquier canal con un coste (por símbolo de la fuente) acotado e inversamente proporcional a la capacidad del canal.

5.2. El teorema de Shannon de codificación de canales ruidosos

En el tema precedente se ha definido la capacidad del canal como el máximo valor de la información que, en media, cada símbolo que sale del mismo proporciona sobre el que realmente se ha introducido. Si se multiplica este valor, C, por el régimen de transmisión de símbolos del canal, v_c, se obtiene el máximo de dicha información por unidad de tiempo; representemos esta última cantidad por C^t.

Por su definición, tanto C como C^t son parámetros característicos del canal. Pero hasta ahora nada se ha dicho sobre si tales parámetros tienen o no algún significado práctico para la resolución de las cuestiones de interés en este libro. Ese es precisamente el objeto de este capítulo, ya que la trascendencia de la capacidad del canal se deriva del teorema de Shannon

de codificación de canales ruidosos. Este teorema, tal y como fue enunciado originalmente por el propio Shannon, consta de dos proposiciones:

- Tenga un canal discreto capacidad C^t y una fuente discreta entropía H^t. Si $H^t < C^t$, existe un sistema de codificación tal que la salida de la fuente puede transmitirse sobre el canal con una frecuencia de errores arbitrariamente pequeña.
- Si $H^t > C^t$, lo anterior es imposible.

En lo que sigue consideraremos ambas proposiciones por separado y se denominarán, respectivamente, teorema directo y teorema inverso.

5.2.1. Teorema directo

Ocupémonos en primer lugar de la primera proposición. Se va a demostrar que si la entropía de la fuente (por unidad de tiempo) es menor que la capacidad del canal (también por unidad de tiempo), entonces es posible, con ciertas condiciones, una recepción arbitrariamente fiable.

El método de demostración no se basará en la obtención de un código que tenga las propiedades deseadas, sino en mostrar que tal código puede existir. Bastará, pues, encontrar un procedimiento cualquiera para la construcción de una cierta clase de códigos (por lo que pueden imponerse tantas restricciones como se deseen al codificador y al decodificador), promediar la frecuencia de errores entre todos los códigos considerados y mostrar que esta frecuencia puede hacerse arbitrariamente pequeña. Siendo así, como al menos uno de los códigos habrá de tener una frecuencia de errores al menos tan pequeña como el promedio, entonces la frecuencia de los errores de este código podrá hacerse también arbitrariamente pequeña.

Pues bien, la primera restricción que impondremos será precisamente la de realizar la descomposición (ya comentada en el primer capítulo del libro) de la pareja codificador–decodificador en un codificador–decodificador de fuente y en un codificador–decodificador de canal.

Codificación de fuente: El codificador de fuente transformará los mensajes de la fuente en secuencias de símbolos binarios. Y conocido el teorema de Shannon de codificación de fuente (que garantiza la posibilidad de realizar una codificación de fuente asintóticamente óptima), para realizar la codificación de canal puede ignorarse la naturaleza de la fuente, ya que, tras una codificación de fuente óptima, siempre se tiene que los distintos símbolos del mensaje son equiprobables. De esta

forma, el proceso de codificación del canal se independiza de la fuente, y dependerá únicamente del canal, lo que, por lo demás, también resulta muy conveniente en la práctica.

Por otra parte, en virtud del mencionado teorema de codificación de fuente, el codificador de fuente óptimo transforma los símbolos de la fuente en una secuencia de símbolos binarios, de forma que existe una relación conocida entre las velocidades a la que generan los símbolos la fuente, v_f, y el codificador de fuente, v_{cf}:

$$v_{cf} = H_2(X)v_f$$

siendo $H_2(X)$ la entropía de la fuente (expresada en bits).

Codificación de canal: En cuanto al proceso de codificación de canal, restringiremos nuestra atención a la codificación en bloques.

- En transmisión, se separa la secuencia de símbolos que se va a enviar en bloques de símbolos del mismo tamaño. Sea k el número de símbolos de cada bloque. Existen entonces $M = 2^k$ diferentes bloques posibles; y el codificador debe asignar a cada uno de tales bloques otro bloque de símbolos del alfabeto de entrada del canal, que se supondrá de longitud también fija n, al que llamaremos palabra del código. Representemos cada una de las $M = 2^k$ palabras del código por $\mathbf{x}_i = (x_i^1, x_i^2, \ldots, x_i^n)$. Un código tal se denominará código (M, n). Nótese que, en virtud de la existencia de un proceso de codificación de fuente óptimo, pueden suponerse equiprobables todas las palabras del código aquí consideradas.

  Un parámetro típico de un código de bloques tal es la tasa R, que se define como $R = k/n$ y representa el número de símbolos del mensaje a transmitir que se envían por cada uso del canal; o lo que es lo mismo, dicha tasa es la razón entre las velocidades de los símbolos a la salida del codificador, v_c, y a su entrada, v_{cf}:

  $$R = \frac{v_{cf}}{v_c}$$

  por lo que, en las condiciones en que $v_{cf} = H_2(X)v_f$, es equivalente la condición $R < C(\text{bits})$ a la condición

  $$H_2(X)v_f < C(\text{bits})v_c$$

  es decir, que la entropía de la fuente por unidad de tiempo sea menor que la capacidad del canal por unidad de tiempo.

- En recepción, el decodificador debe adivinar, para cada bloque de símbolos de longitud n, qué palabra del código se envió; y

decimos que se ha producido un error de decodificación si no la adivina correctamente.

Puesto que no nos interesan los símbolos particulares que constituyen una determinada palabra del código, puede considerarse la codificación simplemente como una correspondencia entre los enteros comprendidos entre 1 y M y las palabras del código $\mathbf{x}_1$ a $\mathbf{x}_M$: si se quiere enviar el mensaje i, se transmite la palabra $\mathbf{x}_i$. El decodificador, a partir del bloque recibido como consecuencia de la transmisión de $\mathbf{x}_i$, produce un entero i'. Y habrá ocurrido un error si $i' \neq i$. Sea $\mathrm{P}_{e,i}$ la probabilidad de tal error. Dicha probabilidad (que será, obviamente, función tanto del canal de transmisión empleado como de los métodos de codificación y decodificación) no tiene por qué ser la misma para todas las palabras del código. Ahora bien, por transmisión fiable se entiende aquí que la probabilidad de error en la transmisión de cualquiera de las palabras del código puede hacerse arbitrariamente pequeña, o lo que es lo mismo, puede hacerse arbitrariamente pequeña la mayor de todas las $\mathrm{P}_{e,i}$, llamémosla P_e.

Es bien sabido que se puede hacer P_e tan pequeña como se quiera disminuyendo R. Pero lo que ahora se va a probar es algo que a primera vista resulta verdaderamente sorprendente: no es necesario reducir indefinidamente R para hacer P_e tan pequeña como se quiera; basta que $R < C$ (expresada C en bits). Así, para cualquier $R < C$, puede hacerse P_e tan pequeña como se quiera sin que disminuya R, incrementando el valor de n.

Obsérvese finalmente que, dado que $R = k/n$, y habida cuenta de que $k = \log_2 M$, resulta que

$$R = \frac{\log_2 M}{n}$$

o, equivalentemente,

$$M = 2^{nR}$$

expresiones ambas que ligan el número de palabras del código, M, con su tasa, R. Obviamente, esta tasa R no puede tomar cualquier valor. No obstante, a veces interesa considerar una variación continua de R, y en dichos casos será necesario corregir la expresión anterior de la siguiente forma:

$$M = \left\lceil 2^{nR} \right\rceil .$$

Teniendo en cuenta toda la exposición anterior, el teorema directo de codificación de canal puede enunciarse como sigue:

TEOREMA 5.1. *Si se considera una fuente binaria de entropía máxima, para cualquier tasa $R < C(bits)$, existe una secuencia de códigos $(\lceil 2^{nR} \rceil, n)$ cuya máxima probabilidad de error $\mathrm{P}_e \xrightarrow[n\to\infty]{} 0$.*

Demostración. Queremos investigar la mínima probabilidad conseguible de decodificar con error como una función de la tasa R, de la longitud de los bloques n y del canal. Se verá que existe una cota superior para esta probabilidad, que decae exponencialmente con la longitud de los bloques para todas las tasas por debajo de la capacidad. Esta cota se obtiene analizando un conjunto de códigos, no únicamente un buen código. Esta peculiar aproximación está motivada por el hecho de que, para valores interesantes de n y R, no se conoce ninguna forma de encontrar códigos que minimicen la probabilidad de decodificar con error, e, incluso si pudiesen encontrarse tales códigos, el número de posibles secuencias recibidas haría prohibitivo un cálculo directo de la probabilidad de error.

Para definir el conjunto de los códigos de bloques que se van a considerar, sea $Q_n(\mathbf{x})$ una asignación arbitraria de probabilidades al conjunto de secuencias de longitud n que van a enviarse a través del canal, y elijamos todas las palabras del código independientemente con estas mismas probabilidades. Así, en el conjunto de códigos considerado, la probabilidad de un código particular, digamos $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$, es

$$\prod_{i=1}^{m} Q_n(\mathbf{x}_i).$$

Obsérvese que no se excluye la posibilidad de que en el código haya palabras repetidas, aun cuando tal código sería claramente inadecuado. Cada código del conjunto tiene su propia probabilidad de error de decodificación. Acotaremos la esperanza, sobre todo el conjunto de códigos, de esta probabilidad de error. Como al menos un código del conjunto debe tener una probabilidad de error tan pequeña como el promedio de las probabilidades del conjunto, ésta será una cota superior para la probabilidad de error del mejor código.

Caso de código de dos palabras

Consideremos un canal discreto arbitrario donde $\mathrm{P}(\mathbf{y} \,|\, \mathbf{x})$ representa la probabilidad de que se reciba una secuencia $\mathbf{y}$ cuando se ha enviado la secuencia $\mathbf{x}$. Sean $\mathbf{x}_1$ y $\mathbf{x}_2$ dos palabras del código de longitud n, y supongamos que se usa la regla de decodificación de máxima verosimilitud, decodificando el mensaje 1 si $\mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1) > \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)$, y decodificando el mensaje 2 en caso contrario.

La probabilidad de error cuando se envía el mensaje 1 es

$$\mathrm{P}_{e,1} = \sum_{\mathbf{y} \in \mathbf{Y}_1^C} \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)$$

siendo $\mathbf{Y}_1^C$, que es el complemento de $\mathbf{Y_1}$, el conjunto de todas las secuencias de longitud n que no se decodifican como el mensaje 1.

Para cualquier $\mathbf{y} \in \mathbf{Y}_1^C$, y para cualquier $s > 0$, se tiene que $\mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^s \geq \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^s$; y por tanto, se cumple que

$$\mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1) \leq \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^{1-s}\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^s; \quad 0 < s < 1.$$

Entonces, se puede escribir:

$$\mathrm{P}_{e,1} \leq \sum_{\mathbf{y}} \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^{1-s}\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^s; \quad 0 < s < 1.$$

De la misma forma:

$$\mathrm{P}_{e,2} \leq \sum_{\mathbf{y}} \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^{1-r}\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^r; \quad 0 < r < 1.$$

Obsérvese que, si en la cota de $\mathrm{P}_{e,2}$ se sustituye r por $(1-s)$, se obtiene la misma cota que para $\mathrm{P}_{e,1}$. Por tanto, sin pérdida de generalidad, se puede escribir:

$$\mathrm{P}_{e,i} \leq \sum_{\mathbf{y}} \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^{1-s}\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^s; \quad i = 1, 2; \quad 0 < s < 1.$$

Si se considera ahora el conjunto de códigos donde las palabras del código se seleccionan independientemente usando la asignación de probabilidades $Q_n(\mathbf{x})$, entonces la probabilidad del código de palabras $\mathbf{x}_1$ y $\mathbf{x}_2$ es $Q_n(\mathbf{x}_1)Q_n(\mathbf{x}_2)$. Así, la probabilidad de error media sobre el conjunto de códigos está acotada por:

$$\begin{aligned}
\mathrm{E}\left[\mathrm{P}_{e,i}\right] &= \sum_{\mathbf{x}_1} \sum_{\mathbf{x}_2} Q_n(\mathbf{x}_1) Q_n(\mathbf{x}_2)\, \mathrm{P}_{e,i}(\mathbf{x}_1, \mathbf{x}_2) \\
&\leq \sum_{\mathbf{x}_1} \sum_{\mathbf{x}_2} \sum_{\mathbf{y}} Q_n(\mathbf{x}_1) Q_n(\mathbf{x}_2)\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^{1-s}\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^s \\
&= \sum_{\mathbf{y}} \left[\sum_{\mathbf{x}_1} Q_n(\mathbf{x}_1)\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)^{1-s} \right] \left[\sum_{\mathbf{x}_2} Q_n(\mathbf{x}_2)\, \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2)^s \right].
\end{aligned}$$

Y tomando $s = 1/2$:

$$\mathrm{E}\left[\mathrm{P}_{e,i}\right] \leq \sum_{\mathbf{y}} \left[\sum_{\mathbf{x}} Q_n(\mathbf{x}) \sqrt{\mathrm{P}(\mathbf{y} \,|\, \mathbf{x})} \right]^2.$$

Ahora bien, como, si el canal no tiene memoria, se cumple que

$$\mathrm{P}(\mathbf{y} \,|\, \mathbf{x}) = \prod_{i=1}^{n} \mathrm{P}(y_i \,|\, x_i)$$

entonces, si se elige $Q_n(\mathbf{x}) = \prod_{i=1}^{n} Q(x_i)$, se tiene:

$$\begin{aligned}
\mathrm{E}\left[\mathrm{P}_{e,i}\right] &\leq \sum_{y_1} \cdots \sum_{y_n} \left[\sum_{x_1} \cdots \sum_{x_n} \prod_{i=1}^{n} Q(x_i) \sqrt{\mathrm{P}(y_i \,|\, x_i)} \right]^2 \\
&= \sum_{y_1} \cdots \sum_{y_n} \left[\prod_{i=1}^{n} \sum_{x_i} Q(x_i) \sqrt{\mathrm{P}(y_i \,|\, x_i)} \right]^2 \\
&= \prod_{i=1}^{n} \sum_{y_i} \left[\sum_{x_i} Q(x_i) \sqrt{\mathrm{P}(y_i \,|\, x_i)} \right]^2 \\
&= \left[\sum_{j=0}^{J-1} \left(\sum_{k=0}^{K-1} Q(k) \sqrt{\mathrm{P}(j \,|\, k)} \right)^2 \right]^n
\end{aligned}$$

siguiéndose la última igualdad del hecho de que todos los x_i tienen el mismo recorrido (que es el alfabeto de entrada del canal, que hemos supuesto dado por $\{0, 1, \ldots, K-1\}$) y también todos los y_i tienen el mismo recorrido (que es el alfabeto de salida del canal, que hemos supuesto dado por $\{0, 1, \ldots, J-1\}$). Se tiene, entonces, una cota para la probabilidad de error media cuando el código tiene sólo dos palabras. Dicha cota es función de las probabilidades con las que se eligen independientemente los diferentes símbolos de las palabras del código, $Q(k)$, y de las probabilidades de transición del canal (supuesto sin memoria), $\mathrm{P}(j \,|\, k)$.

Código con un número arbitrario de palabras

Existe una forma alternativa de derivar la cota obtenida para $\mathrm{E}\left[\mathrm{P}_{e,i}\right]$ que es más fácilmente generalizable para códigos con un número arbitrario de palabras. Observe que $\mathrm{E}\left[\mathrm{P}_{e,i}\right]$ es un promedio sobre $\mathbf{x}_1$, $\mathbf{x}_2$ e $\mathbf{y}$. Cuando se envía el mensaje 1 codificado como $\mathbf{x}_1$, $\mathbf{y}$ ocurre con probabilidad $\mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)$, y suponemos que ha ocurrido un error si $\mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_2) \geq \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)$. Así:

$$\mathrm{E}\left[\mathrm{P}_{e,1}\right] = \sum_{\mathbf{x}_1} Q_n(\mathbf{x}_1) \sum_{\mathbf{y}} \mathrm{P}(\mathbf{y} \,|\, \mathbf{x}_1)\, \mathrm{P}(\text{error} \,|\, i = 1, \mathbf{x}_1, \mathbf{y})$$

donde $\mathrm{P}(\text{error} \,|\, i = 1, \mathbf{x}_1, \mathbf{y})$ es la probabilidad, sobre el conjunto de posibilidades de $\mathbf{x}_2$, de que se produzca un error cuando se envía el mensaje 1

codificado como $\mathbf{x}_1$ y se recibe $\mathbf{y}$, por lo que:

$$\begin{aligned}\mathrm{P}(\text{error}\,|\,i=1,\mathbf{x}_1,\mathbf{y}) &= \sum_{\mathbf{x}_2:\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_2)\geq\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_1)} Q_n(\mathbf{x}_2) \\ &\leq \sum_{\mathbf{x}_2} Q_n(\mathbf{x}_2)\left[\frac{\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_2)}{\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_1)}\right]^s,\ \text{para cualquier } s>0.\end{aligned}$$

Siguiendo esta misma línea argumental, para un código con un número arbitrario de palabras M, se tiene que

$$\mathrm{E}\left[\mathrm{P}_{e,i}\right] = \sum_{\mathbf{x}_i}\sum_{\mathbf{y}} Q_n(\mathbf{x}_i)\,\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_i)\,\mathrm{P}(\text{error}\,|\,i,\mathbf{x}_i,\mathbf{y})$$

y que

$$\mathrm{P}(\text{error}\,|\,i,\mathbf{x}_i,\mathbf{y}) \leq \mathrm{P}\left(\bigcup_{i'\neq i} A_{i'}\right)$$

siendo $A_{i'}$, para cada $i'\neq i$, el evento consistente en la elección de la palabra del código $\mathbf{x}_{i'}$ de forma que $\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_{i'})\geq\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_i)$; y dándose la desigualdad (en lugar de la igualdad) por el hecho de que un decodificador de máxima verosimilitud no necesariamente comete error si $\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_{i'})=\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_i)$ para algún i.

Por otra parte, es fácil verificar que se cumple que:

$$\mathrm{P}\left(\bigcup_{i'\neq i} A_{i'}\right) \leq \left[\sum_{i'\neq i}\mathrm{P}(A_{i'})\right]^\rho,\ \text{para cualquier } \rho: 0<\rho\leq 1$$

$$\begin{aligned}\mathrm{P}(A_{i'}) &= \sum_{\mathbf{x}_{i'}:\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_{i'})\geq\mathrm{P}(\mathbf{y}\,|\,\mathbf{x})} Q_n(\mathbf{x}_{i'}) \\ &\leq \sum_{\mathbf{x}_{i'}} Q_n(\mathbf{x}_{i'})\left[\frac{\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_{i'})}{\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_i)}\right]^s,\ \text{cualquier } s>0.\end{aligned}$$

Así pues, dado que hay $M-1$ casos posibles de valores $i'\neq i$, se cumple que:

$$\mathrm{P}(\text{error}\,|\,i,\mathbf{x}_i,\mathbf{y}) \leq \left\{(M-1)\sum_{\mathbf{x}} Q_n(\mathbf{x})\left[\frac{\mathrm{P}(\mathbf{y}\,|\,\mathbf{x})}{\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_i)}\right]^s\right\}^\rho$$

de donde se sigue que

$$\mathrm{E}\left[\mathrm{P}_{e,i}\right] \leq (M-1)^\rho \sum_{\mathbf{y}}\left[\sum_{\mathbf{x}_i} Q_n(\mathbf{x}_i)\,\mathrm{P}(\mathbf{y}\,|\,\mathbf{x}_i)^{1-s\rho}\right]\left[\sum_{\mathbf{x}} Q_n(\mathbf{x})\,\mathrm{P}(\mathbf{y}\,|\,\mathbf{x})^s\right]^\rho.$$

Finalmente, tomando $1 - s\rho = s$, es decir, $s = 1/(1+\rho)$, y teniendo en cuenta que $\mathbf{x}_i$ no es más que una variable auxiliar de suma, se tiene que

$$\mathrm{E}\left[\mathrm{P}_{e,i}\right] \leq (M-1)^{\rho} \sum_{\mathbf{y}} \left[\sum_{\mathbf{x}} Q_n(\mathbf{x})\, \mathrm{P}(\mathbf{y} \mid \mathbf{x})^{1/(1+\rho)} \right]^{1+\rho}, \quad 0 < \rho \leq 1.$$

Es fácil verificar que también se cumple la desigualdad anterior en el caso particular de que $\rho = 0$, sin más que verificar que, para ese valor de ρ, el segundo término de la desigualdad vale 1.

Si se particulariza esta desigualdad, al igual que hicimos para el caso de código de dos palabras, para el caso particular de canal sin memoria y elección independiente de los símbolos de cada palabra del código de acuerdo con una distribución de probabilidades $Q(k), k = 0, 1, \ldots, K-1$, se obtiene que

$$\begin{aligned}
\mathrm{E}\left[\mathrm{P}_{e,i}\right] &\leq (M-1)^{\rho} \sum_{y_1} \cdots \sum_{y_n} \left[\sum_{x_1} \cdots \sum_{x_n} \prod_{i=1}^{n} Q(x_i)\, \mathrm{P}(y_i \mid x_i)^{1/(1+\rho)} \right]^{1+\rho} \\
&= (M-1)^{\rho} \prod_{i=1}^{n} \sum_{y_i} \left[\sum_{x_i} Q(x_i)\, \mathrm{P}(y_i \mid x_i)^{1/(1+\rho)} \right]^{1+\rho} \\
&= (M-1)^{\rho} \left[\sum_{j=0}^{J-1} \left(\sum_{k=0}^{K-1} Q(k)\, \mathrm{P}(j \mid k)^{1/(1+\rho)} \right)^{1+\rho} \right]^{n}.
\end{aligned}$$

Si ahora se tiene en cuenta que $R = \frac{\log_K M}{n}$, entonces $M = K^{nR}$; por lo que, para R fija, M varía exponencialmente con n. Pero, como la variación continua con n, para valores de R fijos, da a veces valores no enteros para K^{nR}, debe tomarse como número de palabras del código $M = \lceil K^{nR} \rceil$, es decir, el menor número entero mayor o igual que K^{nR}. Así, dado que $M - 1 < K^{nR}$, se tiene que

$$\mathrm{E}\left[\mathrm{P}_{e,i}\right] \leq K^{nR\rho} \left[\sum_{j=0}^{J-1} \left(\sum_{k=0}^{K-1} Q(k)\, \mathrm{P}(j \mid k)^{1/(1+\rho)} \right)^{1+\rho} \right]^{n}; \quad 0 \leq \rho \leq 1.$$

Es decir, se tiene que

$$\mathrm{E}\left[\mathrm{P}_{e,i}\right] \leq K^{-n[E_0(\rho,\mathbf{Q}) - \rho R]}, \quad 0 \leq \rho \leq 1$$

donde

$$\begin{aligned}
E_0(\rho, \mathbf{Q}) &= -\log_K \sum_{j=0}^{J-1} \left[\sum_{k=0}^{K-1} Q(k)\, \mathrm{P}(j \mid k)^{1/(1+\rho)} \right]^{1+\rho} \\
\mathbf{Q} &= \left(Q(0), Q(1), \ldots, Q(K-1)\right).
\end{aligned}$$

Como la desigualdad anterior es válida para cada mensaje del código, vemos que el promedio de las probabilidades de error, considerados todos los posibles mensajes, con probabilidades arbitrarias $\mathrm{P}(i)$, satisface

$$\mathrm{E}[\mathrm{P}_e] = \sum_{i=1}^{M} \mathrm{P}(i)\,\mathrm{E}[\mathrm{P}_{e,i}] \leq K^{-n[E_0(\rho,\mathbf{Q})-\rho R]}.$$

Finalmente, puesto que ρ y $\mathbf{Q}$ son arbitrarios, obtenemos la cota más cercana eligiendo ρ y $\mathbf{Q}$ de forma que se maximice $E_0(\rho,\mathbf{Q}) - \rho R$. Esto nos lleva a la siguiente definición de exponente de codificación aleatoria:

$$E_a(R) = \max_{0\leq\rho\leq 1,\mathbf{Q}} [E_0(\rho,\mathbf{Q}) - \rho R]$$

de forma que se cumple que

$$\mathrm{E}[\mathrm{P}_e] \leq K^{-nE_a(R)}.$$

Análisis de $E_a(R)$

Veamos, finalmente, que $E_a(R)$ es una función positiva, decreciente y cóncava $\forall 0 \leq R < C$. Para comprender la naturaleza de $E_a(R)$ hay que analizar la variación con ρ de $E_0(\rho,\mathbf{Q})$, pudiendo observarse que se cumple que:

1. $E_0(\rho,\mathbf{Q}) \geq 0; \quad \rho \geq 0$ (cumpliéndose la igualdad si y sólo si $\rho = 0$);
2. $0 < \dfrac{\partial E_0(\rho,\mathbf{Q})}{\partial \rho} \leq I(\mathbf{Q}); \quad \rho \geq 0$ (cumpliéndose la igualdad en la segunda desigualdad de la cadena si $\rho = 0$);
3. $\dfrac{\partial^2 E_0(\rho,\mathbf{Q})}{\partial \rho^2} \leq 0; \quad \rho \geq 0$ (cumpliéndose la igualdad si y sólo si $\forall j,k$: $Q(k)\,\mathrm{P}(j\,|\,k) > 0$, se tiene que $I(\mathbf{Q})$ es determinista);

donde $I(\mathbf{Q})$ es la información mutua para el canal considerado, que es función únicamente de $\mathbf{Q}$:

$$\begin{aligned} I(\mathbf{Q}) &= \sum_{k=0}^{K-1}\sum_{j=0}^{J-1} Q(k)\,\mathrm{P}(j\,|\,k)\log\frac{\mathrm{P}(j\,|\,k)}{\sum_i Q(i)\,\mathrm{P}(j\,|\,i)} \\ &= \left.\frac{\partial E_0(\rho,\mathbf{Q})}{\partial \rho}\right|_{\rho=0}. \end{aligned}$$

Teniendo en cuenta las propiedades anteriormente enumeradas, es fácil maximizar $E_0(\rho, \mathbf{Q}) - \rho R$ sobre ρ para obtener $E_a(R, \mathbf{Q})$:

$$E_a(R, \mathbf{Q}) = \max_{0 \leq \rho \leq 1} \left[E_0(\rho, \mathbf{Q}) - \rho R \right].$$

El máximo se obtendrá para aquel valor de ρ ($0 \leq \rho \leq 1$) que satisfaga la ecuación

$$\frac{\partial E_0(\rho, \mathbf{Q})}{\partial \rho} - R = 0$$

puesto que $\dfrac{\partial^2 E_0(\rho, \mathbf{Q})}{\partial \rho^2} \leq 0, \quad \rho \geq 0.$

Además, como $\dfrac{\partial E_0}{\partial \rho}$ es continua y decreciente con ρ, el rango de variación de R para ese máximo será:

$$\left.\frac{\partial E_0(\rho, \mathbf{Q})}{\partial \rho}\right|_{\rho=1} \leq R \leq \left.\frac{\partial E_0(\rho, \mathbf{Q})}{\partial \rho}\right|_{\rho=0} = I(\mathbf{Q})$$

por lo que, para los valores de R en ese rango, se tiene que

$$R = \frac{\partial E_0(\rho, \mathbf{Q})}{\partial \rho}, \quad 0 \leq \rho \leq 1 \tag{5.1}$$

$$E_a(R, \mathbf{Q}) = E_0(\rho, \mathbf{Q}) - \rho \frac{\partial E_0(\rho, \mathbf{Q})}{\partial \rho}. \tag{5.2}$$

Por otra parte,

- Si $\left.\dfrac{\partial E_0}{\partial \rho}\right|_{\rho=1} > R$, entonces la función es estrictamente creciente con ρ en el intervalo $0 \leq \rho \leq 1$, por lo que alcanzará su máximo para $\rho = 1$, que será
$$E_a(R, \mathbf{Q}) = E_0(1, \mathbf{Q}) - R.$$

- Si $I(\mathbf{Q}) < R$, entonces la función es estrictamente decreciente con ρ en el intervalo $0 \leq \rho \leq 1$, por lo que alcanzará su máximo para $\rho = 0$, que será
$$E_a(R, \mathbf{Q}) = 0.$$

En resumen, pues, $E_a(R, \mathbf{Q})$ es estrictamente decreciente con R, y positiva para todo $R < I(\mathbf{Q})$.

Además, derivando con respecto a ρ las dos expresiones de (5.1)-(5.2) se tiene que:

$$\frac{\partial R}{\partial \rho} = \frac{\partial^2 E_0}{\partial \rho^2}$$
$$\frac{\partial E_a}{\partial \rho} = -\rho \frac{\partial^2 E_0}{\partial \rho^2}$$

de donde se sigue que:

$$\frac{\partial E_a(R, \mathbf{Q})}{\partial R} = -\rho$$
$$\frac{\partial^2 E_a(R, \mathbf{Q})}{\partial R^2} = -\left[\frac{\partial^2 E_0}{\partial \rho^2}\right]^{-1} \geq 0$$

por lo que $E_a(R, \mathbf{Q})$ es cóncava en $R \quad \forall R \geq 0$.

Así, dado que

$$E_a(R) = \max_{\mathbf{Q}} E_a(R, \mathbf{Q})$$

y que $E_a(R, \mathbf{Q})$ es decreciente con R, positiva $\forall R < I(\mathbf{Q})$ y cóncava $\forall R \geq 0$, entonces también es $E_a(R)$ decreciente con R, positiva $\forall R < I(\mathbf{Q}) = C$ y cóncava $\forall R \geq 0$. ▶

5.2.2. Teorema inverso

El teorema directo establece, según acabamos de ver, que si la entropía (por unidad de tiempo) de la fuente es menor que la capacidad del canal (por unidad de tiempo), entonces puede reducirse la probabilidad de error en la transmisión de los mensajes a un valor arbitrariamente pequeño usando una codificación adecuada. Pues bien, en este apartado se va a establecer el resultado inverso, es decir, la imposibilidad de conseguir una probabilidad arbitrariamente pequeña cuando la entropía de la fuente es mayor que la capacidad del canal.

TEOREMA 5.2. *Si la entropía (por unidad de tiempo) de la fuente es mayor que la capacidad del canal (por unidad de tiempo), entonces no existe ningún esquema de codificación con el que pueda reducirse la probabilidad de error en la transmisión de los mensajes a un valor arbitrariamente pequeño.*

DEMOSTRACIÓN. A diferencia de la demostración del teorema directo, en la que había libertad para escoger el esquema de codificación, aquí es inadmisible la imposición de restricción alguna en el proceso de codificación. Para cualquier método de codificación, en las condiciones enunciadas, debe haber

una cota inferior no nula para la probabilidad de error en la transmisión de los mensajes, sea cual sea la naturaleza de éstos.

Así pues, en lo que sigue se supondrá que:

- un mensaje de la fuente será una secuencia de símbolos del alfabeto de la fuente de longitud arbitraria L, $\mathbf{u} = (u_1, u_2, \ldots, u_L)$;
- el codificador transforma cada mensaje $\mathbf{u}$ en una secuencia arbitraria de símbolos del alfabeto de entrada del canal de longitud N, $\mathbf{x} = (x_1, x_2, \ldots, x_N)$;
- la secuencia $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ es el mensaje que sale del canal cuando se transmite la secuencia $\mathbf{x}$;
- la salida del decodificador correspondiente a la secuencia $\mathbf{y}$ es también una secuencia de símbolos del alfabeto de la fuente de longitud L, $\mathbf{v} = (v_1, v_2, \ldots, v_L)$.

El objetivo del sistema de comunicación es conseguir que la secuencia $\mathbf{v}$ reproduzca la secuencia $\mathbf{u}$. Si $u_i \neq v_i$, entonces se dice que se ha producido un error en la transmisión del i-ésimo símbolo del mensaje. Sea la probabilidad de tal error $P_{e,i}$, y definamos la probabilidad de error media por símbolo en la transmisión del mensaje como

$$P_e = \frac{1}{L} \sum_{i=1}^{L} P_{e,i}.$$

Pues bien, el presente teorema establece que cuando la entropía de la fuente es mayor que la capacidad del canal, existe una cota inferior no nula para P_e, es decir, no puede hacerse P_e arbitrariamente pequeña. Nótese que el hecho de que pudiese hacerse P_e arbitrariamente pequeña no implicaría necesariamente una transmisión sin errores; en cambio, no puede haber transmisión sin errores si no es posible hacer P_e arbitrariamente pequeña.

Sea $\mathbf{U} = (U_1, U_2, \ldots, U_L)$ la variable aleatoria L-dimensional que representa todos los mensajes de longitud L que se pueden enviar, y sea $\mathbf{V} = (V_1, V_2, \ldots, V_L)$ la variable aleatoria L-dimensional que representa todos los mensajes de longitud L que se pueden recibir. Por la regla de la cadena, se cumple que

$$\begin{aligned} H(\mathbf{U} \,|\, \mathbf{V}) &= H(U_1 \,|\, \mathbf{V}) + H(U_2 \,|\, U_1\mathbf{V}) + \cdots + H(U_L \,|\, U_1 \ldots U_{L-1}\mathbf{V}) \\ &\leq \sum_{i=1}^{L} H(U_i \,|\, V_i) \end{aligned}$$

siguiéndose la desigualdad del hecho de que $H(X \mid Z) \geq H(X \mid Y, Z)$ (véase el apartado 4.3).

Ahora, aplicando la desigualdad de Fano a cada uno de los términos del sumatorio anterior, si el alfabeto de la fuente tiene M elementos, se tiene que

$$H(\mathbf{U} \mid \mathbf{V}) \leq \sum_{i=1}^{L} [P_{e,i} \log(M-1) + H(P_{e,i})]$$

$$\frac{1}{L} H(\mathbf{U} \mid \mathbf{V}) \leq P_e \log(M-1) + \frac{1}{L} \sum_{i=1}^{L} H(P_{e,i})$$

y dado que, como consecuencia de la convexidad de la entropía, se cumple que

$$\frac{1}{L} \sum_{i=1}^{L} H(P_{e,i}) \leq H(P_e)$$

se tiene finalmente que

$$\frac{1}{L} H(\mathbf{U} \mid \mathbf{V}) \leq P_e \log(M-1) + H(P_e).$$

Por otra parte, en virtud del teorema de procesamiento de la información, se tiene que

$$I(\mathbf{U}; \mathbf{V}) \leq I(\mathbf{X}; \mathbf{Y})$$

siendo $\mathbf{X}$ e $\mathbf{Y}$ las variables aleatorias que representan, respectivamente, las secuencias que entran y salen del canal para el envío de un mensaje.

En virtud de las dos anteriores desigualdades, se tiene que

$$\begin{aligned} P_e \log(M-1) + H(P_e) &\geq \frac{1}{L} H(\mathbf{U} \mid \mathbf{V}) \\ &= \frac{1}{L} H(\mathbf{U}) - \frac{1}{L} I(\mathbf{U}; \mathbf{V}) \\ &\geq \frac{1}{L} H(\mathbf{U}) - \frac{1}{L} I(\mathbf{X}; \mathbf{Y}) \\ &\geq \frac{1}{L} H(\mathbf{U}) - \frac{N}{L} C \end{aligned}$$

verificándose la última desigualdad para el caso de canales discretos sin memoria.

Es posible ahora relacionar N y L a través de las velocidades de generación de símbolos de la fuente, v_f, y de transmisión de símbolos del canal, v_c, de forma que se tiene que

$$N \leq \frac{v_c}{v_f} L.$$

Y, por otra parte, se tiene que

$$\frac{1}{L}H(\mathbf{U}) \geq H(\mathcal{U})$$

siendo $H(\mathcal{U})$ la tasa de entropía de la fuente (véase el apartado 3.5). ▶

5.3. Interpretación del teorema

Dado el papel central que juega este teorema en la Teoría de la Información, y dado que su demostración resulta farragosa en exceso, en este apartado vamos a detenernos a comentar sucintamente las implicaciones del mismo, que lógicamente estarán todas ellas relacionadas entre sí. Esencialmente, este teorema establece que la exigencia de fiabilidad de la comunicación impone:

- una cota superior al régimen de transmisión de la fuente;
- una cota inferior para el coste de transmisión de los símbolos de la fuente;
- una cota inferior para la cantidad de redundancia que es preciso introducir en los mensajes codificados;

y a estas cotas es posible acercarse tanto como se quiera.

Régimen de transmisión de la fuente. La capacidad del canal impone una cota superior al régimen de transmisión de símbolos de la fuente para la posibilidad de una comunicación fiable. Tal cota superior, a la que es posible aproximarse tanto como se desee, viene dada por

$$\frac{C}{H(X)}v_c$$

donde v_c representa el régimen de transmisión de símbolos del canal y $H(X)$ representa la entropía de la fuente expresada en las mismas unidades que la capacidad del canal C.

Efectivamente, puesto que para que sea posible la comunicación fiable ha de cumplirse que $H^t < C^t$, es decir, $v_f H(X) < v_c C$, entonces habrá de ser

$$v_f < \frac{C}{H(X)}v_c.$$

Coste de transmisión. Este teorema establece también que el coste de transmisión por símbolo (de la fuente) tiene una cota inferior, a la que es posible aproximarse tanto como se quiera. Veámoslo. El coste de transmisión por símbolo es, en media, directamente proporcional al número de símbolos de canal que es necesario transmitir, en media, por cada símbolo de la fuente. Y dicho número viene dado simplemente por la razón v_c'/v_f, donde v_f representa el régimen de generación de los símbolos de la fuente y v_c' representa el régimen de transmisión por el canal de los símbolos que representan el mensaje de la fuente. Ahora bien, como deben cumplirse las dos condiciones siguientes:

$$v_c' \leq v_c$$
$$v_f < \frac{C}{H(X)} v_c$$

entonces resulta que

$$\frac{v_c'}{v_f} > \frac{H(X)}{C}.$$

Redundancia Este teorema asegura, además, que se puede realizar la comunicación fiable de forma óptima (es decir, con coste mínimo) separando la codificación en dos procesos independientes, codificación de fuente y codificación de canal, al garantizar la existencia de un código de canal óptimo: un código que, con una cantidad mínima de redundancia (es decir, con una longitud mínima), permite la comunicación fiable.

El código de canal tendrá una longitud, L_c, que vendrá dada por

$$L_c = \frac{v_c}{v_f L_f}$$

donde L_f representa la longitud del código de fuente. En virtud del teorema de Shannon de codificación de fuente, el valor óptimo de L_f es

$$L_f = H_2(X)$$

supuesto que el alfabeto de codificación es binario.

Si se tiene en cuenta ahora que, en virtud del teorema de codificación de canal, para que sea posible la comunicación fiable ha de cumplirse que

$$v_f < \frac{C}{H(X)} v_c$$

entonces se deduce la posibilidad de la comunicación fiable con un código de canal de longitud

$$L_c > \frac{1}{C}$$

siempre que se exprese C en unidades de base el número de elementos del alfabeto de codificación de fuente.

Sentido del término capacidad Por último, el hecho de que alguno de los símbolos que se transmiten por el canal no contengan información de la fuente implica que la fuente sólo dispone realmente de una fracción del tiempo de uso del canal. Y esa fracción es precisamente, en el mejor de los casos, la capacidad del canal (expresada en unidades de base el número de elementos del alfabeto de entrada del canal). Efectivamente, si se supone que el alfabeto de codificación de fuente coincide con el alfabeto de entrada del canal, tal fracción es simplemente

$$\frac{1}{L_c} = C$$

es decir, la razón entre el número de símbolos del canal que serían necesarios para representar cada símbolo del codificador de fuente si el canal fuese ideal (1 símbolo) y el número mínimo de símbolos del canal que son necesarios cuando no lo es (L_c símbolos).

En definitiva, por tanto, la capacidad (por unidad de tiempo) mide efectivamente los símbolos de información que puede transmitir el canal por unidad de tiempo, propiedad que confiere al término de capacidad verdadero sentido.

Ejemplo 5.1. Suponga que se quieren transmitir de forma fiable los mensajes de una fuente binaria sin memoria dada, de entropía $H(X) = 0{,}5$ bits, sobre un cierto canal binario sin memoria ruidoso, de capacidad $C = 0{,}8$ bits y régimen de transmisión $v_c = 1$ símbolo por unidad de tiempo.

- De acuerdo con lo que se acaba de exponer, el régimen de transmisión de la fuente puede aproximarse tanto como se desee a

$$v_f = \frac{C}{H(X)} v_c = 1{,}6 \text{ símbolos fuente/unidad de tiempo},$$

valor que en ningún caso se podrá superar si se mantiene la exigencia de la fiabilidad de la transmisión.

- ¿Cómo es posible que sea $v_f > v_c$? La respuesta es sencilla: porque no se introducen directamente en el canal los símbolos generados por la fuente, sino que los mensajes de la fuente se codifican previamente. Supongamos que tal codificación se hace en dos etapas: se emplea en primer lugar un codificador de fuente, y a continuación un codificador de canal. Supongamos también que ambos codificadores son óptimos, puesto que sabemos de su existencia por los teoremas de Shannon de codificación de fuente y de canal.

El código de fuente óptimo tiene una longitud $L_f = H_2(X) = 0{,}5$. Eso significa que el codificador de fuente compacta los mensajes de la fuente, reduciendo el número de sus símbolos a la mitad. En consecuencia, el régimen de transmisión de símbolos del codificador de fuente, v_{cf}, es la mitad del régimen de transmisión de símbolos de la fuente:

$$v_{cf} = v_f/2 = 0{,}8.$$

Se tiene, ahora sí, un régimen inferior al del canal. Y como el régimen de la fuente no puede aumentarse, so pena de falibilidad, se puede concluir que el codificador de canal tiene que introducir símbolos redundantes a razón de 0,2 símbolos por unidad de tiempo. Es decir, la longitud del código de canal óptimo es, como se esperaba,

$$L_c = 1/0{,}8 = 1/C(\text{bits}).$$

- Por último, si se supone que la transmisión de símbolos por un canal dado tiene un coste unitario de k, entonces el coste de transmisión de los símbolos de la fuente no podrá en ningún caso ser inferior a $\frac{5}{8}k$ (pudiéndonos acercar a ese coste mínimo tanto como queramos), ya que el número de símbolos de canal que representan, en media, cada símbolo de la fuente es

$$\frac{v_c}{v_f} > \frac{H(X)}{C} = \frac{5}{8}.$$ ■

Notas bibliográficas

El teorema de codificación de canales ruidosos se debe a Shannon [59] y es, indudablemente, el resultado más importante de la Teoría de la Información. No obstante, en el artículo anterior no se demuestra el teorema inverso, que se sigue directamente de la desigualdad de Fano. La primera demostración rigurosa del teorema directo fue dada por Feinstein [18]. Posteriormente, Shannon [61] y Wolfowitz [77] proporcionaron demostraciones más sencillas. El método de demostración usado en este capítulo se debe a Gallager [25].

5.A. Decodificación óptima

Vamos a considerar en este apéndice la bondad de la regla de máxima verosimilitud como criterio de decodificación.

En un canal discreto y sin memoria dado, representemos la palabra del código transmitida con el símbolo genérico $\mathbf{u} = (u_1, \ldots, u_n)$, el vector recibido con el símbolo genérico $\mathbf{y} = (y_1, \ldots, y_n)$ y la palabra del código estimada por el decodificador con el signo $\hat{\mathbf{u}}$. Empleando esta notación, la probabilidad de cometer una equivocación en el decodificador se podrá escribir como

$$\mathrm{P}_{\mathrm{error}} = \mathrm{P}(\mathbf{u} \neq \hat{\mathbf{u}}).$$

Por la fórmula de la probabilidad total, si condicionamos sobre el valor del vector recibido y aplicamos la hipótesis de que el canal carece de memoria tendremos que

$$\mathrm{P}_{\mathrm{error}} = \sum_{\mathbf{y} \in \mathbf{Y}} \mathrm{P}(\mathbf{u} \neq \hat{u} \,|\, \mathbf{y})\, \mathrm{P}(\mathbf{y})$$

en donde $\mathbf{Y}$ es el conjunto de todos los vectores que se pueden recibir, $\mathrm{P}(\mathbf{y})$ es la probabilidad de que se haya recibido el vector $\mathbf{y}$ a la salida del canal discreto y $\mathrm{P}(\mathbf{u} \neq \hat{u} \,|\, \mathbf{y})$ es la probabilidad de error al decodificar cuando se ha recibido $\mathbf{y}$. Adviértase que son las suposiciones de canal y código sin memoria (ambas) las que de manera implícita nos permiten escribir que el vector estimado depende solamente del vector recibido actual, por lo que $\mathrm{P}(\mathbf{u} \neq \hat{\mathbf{u}} \,|\, \mathbf{y})$ tiene perfecto sentido. Si, como parece lógico, interesa minimizar la probabilidad de equivocación, tengamos en cuenta dos observaciones:

- $\mathrm{P}(\mathbf{y})$ depende sólo del modelo de canal y de la distribución de probabilidad $\{\mathrm{P}(\mathbf{u})\}$ de los mensajes, pero es independiente del criterio de decisión;
- y $\mathrm{P}(\hat{\mathbf{u}} \neq \mathbf{u} \,|\, \mathbf{y})$ solamente es función, para cualquier $\mathbf{y}$ fijo, de la regla de decodificación.

En consecuencia, para elegir la regla de decodificación que hace mínima la probabilidad de equivocación en el receptor es condición necesaria y suficiente adoptar la regla que hace mínimo cada uno de los términos $\mathrm{P}(\hat{u} \neq \mathbf{u} \mid \mathbf{y})$, para todo $\mathbf{y} \in \mathbf{Y}$.

Ahora bien, fijo el parámetro $\mathbf{y}$

$$\min_{\hat{\mathbf{u}}} \mathrm{P}(\hat{\mathbf{u}} \neq \mathbf{u} \,|\, \mathbf{y}) = 1 - \max_{\hat{\mathbf{u}}} \mathrm{P}(\hat{\mathbf{u}} = \mathbf{u} \,|\, \mathbf{y}).$$

Por tanto, el mínimo global se obtiene cuando, para cada posible vector recibido, $\mathbf{y}$, se elige como estimación de vector transmitido aquél que hace máxima la probabilidad[1] $\mathrm{P}(\mathbf{u}\,|\,\mathbf{y})$

$$\hat{\mathbf{u}} = f(\mathbf{y}) = \arg\max_{\mathbf{u}} \mathrm{P}(\mathbf{u}\,|\,\mathbf{y}). \tag{5.3}$$

El criterio (5.3) se denomina *máximo a posteriori* (MAP) y resulta ser el óptimo en cualquier problema de decisión cuando la función objetivo que se desea minimizar es una probabilidad de error.

Pero, por la fórmula de probabilidad de Bayes,

$$\mathrm{P}(\mathbf{u}\,|\,\mathbf{y}) = \frac{\mathrm{P}(\mathbf{y}\,|\,\mathbf{u})\,\mathrm{P}(\mathbf{u})}{\mathrm{P}(\mathbf{y})}$$

de manera que

$$\arg\max_{\mathbf{u}} \mathrm{P}(\mathbf{u}\,|\,\mathbf{y}) = \arg\max_{\mathbf{u}} \frac{\mathrm{P}(\mathbf{y}\,|\,\mathbf{u})\,\mathrm{P}(\mathbf{u})}{\mathrm{P}(\mathbf{y})}.$$

En esta expresión $\mathrm{P}(\mathbf{y})$ es una constante, y puede prescindirse de su valor para calcular el vector $\mathbf{u}$ con el que se alcanza el máximo buscado. La regla óptima de decisión (MAP) queda, por tanto, como

$$\arg\max_{\mathbf{u}} \mathrm{P}(\mathbf{y}\,|\,\mathbf{u})\,\mathrm{P}(\mathbf{u}) \tag{5.4}$$

expresada ahora en función de las probabilidades de transición del canal $\mathrm{P}(\mathbf{y}\,|\,\mathbf{u})$ relativas a los vectores del código.

Si los mensajes son equiprobables, entonces $\mathrm{P}(\mathbf{u})$ es constante y la expresión (5.4) se simplifica a

$$\hat{\mathbf{u}} = f(\mathbf{y}) = \arg\max_{\mathbf{u}} \mathrm{P}(\mathbf{y}\,|\,\mathbf{u}).$$

Este criterio se conoce como de *máxima verosimilitud* o ML (*maximum likelihood*) y consiste en elegir la palabra del código que hace máxima la probabilidad de recibir el vector $\mathbf{y}$. El criterio ML depende únicamente de las probabilidades de transición en el canal e ignora la distribución de los mensajes.

En el caso especial de considerar un canal binario simétrico con probabilidad de equivocación $0 < p < \frac{1}{2}$, aún es posible simplificar más el criterio de decisión, pues por la independencia de los errores

$$\mathrm{P}(\mathbf{y}\,|\,\mathbf{u}) = \prod_{i=1}^{n} \mathrm{P}(y_i \mid u_i) = p^j (1-p)^{n-j}$$

[1] La notación $\arg\max g(x)$ significa calcular el valor de la variable x que hace máxima la función g.

cuando $\mathbf{u}$ e $\mathbf{y}$ difieren en j de sus n componentes. Pero la expresión

$$p^j(1-p)^{n-j}$$

es estrictamente decreciente para $j = 0, \ldots, n$ cuando $p < \frac{1}{2}$ (pues $p^j(1-p)^{n-j} \propto (p/(1-p))^j$, y $p/(1-p) < 1$). Y esto implica que, para canales binarios simétricos, la decodificación de máxima verosimilitud se reduce a seleccionar la palabra del código con menor número de símbolos distintos del vector recibido o, como también se denomina, a la regla de selección del vecino más próximo

$$\hat{\mathbf{u}} = f(\mathbf{y}) = \arg \min_{\mathbf{u}} |\{i : y_i \neq u_i\}| .$$

PARTE II

Conceptos básicos de control de errores

CAPÍTULO 6

Códigos lineales

En este capítulo se presentan la estructura de la familia de códigos lineales, sus algoritmos de codificación y decodificación, y las nociones básicas sobre sus propiedades de detección y corrección de errores. La discusión se restringe, tanto por sencillez como por ser los de más frecuente aplicación práctica, a los códigos lineales binarios. Pero no se pierde generalidad al proceder de esta forma, pues los resultados y propiedades de más alcance pueden extenderse sin gran esfuerzo a códigos no binarios.

6.1. Introducción

6.1.1. Códigos de bloques

Un código de control de errores es cualquier transformación invertible que, de acuerdo con unas reglas dadas, representa con redundancia (con más símbolos de los necesarios) las secuencias de símbolos de una fuente discreta dada que se supone que genera los símbolos equiprobablemente; la adición de símbolos redundantes introduce cierto grado de dependencia entre los símbolos del mensaje codificado, restringiéndose el universo de las secuencias válidas, de forma que habrá ocurrido ineludiblemente algún error si se observa a la salida del canal una secuencia cuya emisión por la fuente sea imposible. Si se añade suficiente redundancia a la representación del mensaje original, la secuencia recibida será la mayor parte de las veces más parecida a la secuencia emitida que a cualquier otra que se pudiera haber emitido.

El sistema encargado de la conversión de los símbolos de la fuente es el *codificador de canal*. Y el que, a la salida del canal discreto, realiza una

estimación del mensaje emitido es el *decodificador de canal*. En realidad, un decodificador no es sino un dispositivo de decisión o clasificación de las secuencias de símbolos a su entrada que, dada la naturaleza aleatoria de los errores, no siempre podrá inferir con exactitud qué mensaje se ha transmitido. Por tanto, una medida objetiva de la calidad del sistema (canal equivalente) compuesto por el codificador de canal, el propio canal y el decodificador es la probabilidad de error de decodificación: la pequeña probabilidad no nula con que el decodificador yerra al reconstruir el mensaje.

Hasta la publicación de los trabajos de Shannon era una creencia extendida que el incremento en la velocidad de transmisión en un sistema de comunicaciones digital provocaba también un aumento uniforme en la probabilidad de error. Shannon consiguió probar, sin embargo, que mientras la velocidad de transferencia de la información pretendida no superase un límite calculable (la capacidad del canal), era posible disminuir la probabilidad de error a un nivel arbitrariamente pequeño sin otro requisito que introducir cierta cantidad de (mínima) redundancia en la representación de la secuencia de datos original. Es decir, que cualquier canal se puede convertir en uno ideal (entendiendo aquí por ideal tan fiable como se quiera) añadiendo a los mensajes cierta proporción *acotada* de redundancia.[1] Por desgracia, en la demostración de este resultado se sigue una argumentación probabilística y no se describe ningún procedimiento constructivo de los buenos códigos de canal predichos.

Pero el resultado fundamental de existencia que establece el teorema de capacidad de canal de Shannon estimuló de inmediato el análisis formal y la búsqueda de tales transformaciones. En principio, este cambio de representación es arbitrario y se puede realizar de múltiples maneras. Atendiendo tan sólo al método, parece claro que la forma más simple de codificar es asignar a cada secuencia de una longitud prefijada de símbolos de la fuente una única secuencia del código también de longitud fija.[2] El nombre genérico de este proceso es el de codificación de bloques (cf. la definición 3.1, en la página 53).

Dada una fuente discreta sin memoria que emite símbolos (que supondremos equiprobables) pertenecientes a un alfabeto finito $\mathcal{F}$, un codificador de bloques asigna a cada secuencia $(u_1, u_2, \ldots, u_k) \in \mathcal{F}^k$ una única secuencia de símbolos $(v_1, v_2, \ldots, v_n) \in \mathcal{F}^n$ para cierto $n > k$ fijo. Diremos que $(v_1, v_2, \ldots, v_n)$ es la *palabra del código* que corresponde al mensaje $(u_1, u_2, \ldots, u_k)$. El cociente $R_c = \frac{k}{n} < 1$ se llama tasa del código, y es un importante parámetro (adimensional) de eficiencia puesto que mide la propor-

[1] De no ser cierta esta afirmación, carecería de sentido el concepto de capacidad de un canal; sólo se podría hablar con propiedad de la capacidad de un canal para una probabilidad de error dada.

[2] Forma que, por cierto, ya utilizó C. Shannon para deducir la existencia de códigos de canal óptimos.

ción de símbolos del mensaje en cada palabra del código; y $1-R_c = (n-k)/n$ mide la proporción de redundancia por cada símbolo de la secuencia codificada. La diferencia $n-k$ es la *redundancia* (en número de símbolos) del código, y n es su *longitud*. Representaremos normalmente un código con tales parámetros por $\mathcal{C}[n,k]$.

Aunque, en sentido estricto, un código es una aplicación entre $\mathcal{F}^k$ y $\mathcal{F}^n$, las propiedades de interés de un código dado dependen únicamente de la elección de las palabras del código, y no de la asignación particular de éstas a mensajes, que debe ser conocida pero puede ser arbitraria. Por esta razón, cuando se trata con códigos de control de errores se entiende que un código de bloques es un conjunto de n-tuplas, en número igual al de las k-tuplas que deben representarse.

El problema de los códigos de bloques genéricos es que, en ausencia de una estructura regular en el proceso de codificación:

a) es imposible sistematizar su estudio; y

b) la realización práctica de los dispositivos de codificación y decodificación es excesivamente costosa: exige mantener en ambos una tabla o diccionario con la correspondencia completa entre los mensajes y las secuencias del código, tabla cuyo tamaño aumenta exponencialmente con k.

Con la finalidad de soslayar estas dos limitaciones (la inexistencia de una estructura matemática definida y el coste, en términos de memoria y tiempo, de la (de)codificación) vamos a restringir el análisis de los códigos de bloques a aquéllos en que los procedimientos de codificación y decodificación consisten en operaciones aritméticas simples (sumas y multiplicaciones) con los elementos del conjunto $\mathcal{F}$. A tal efecto, es imprescindible contar con operaciones internas bien definidas en el conjunto de símbolos $\mathcal{F}$ con las que dotar al conjunto de palabras del código de una estructura algebraica sobre la que fundamentar el estudio matemático del código.

6.1.2. Aritmética binaria

Nuestro interés se limita en lo sucesivo a los códigos binarios, que son aquéllos cuyo alfabeto de codificación contiene dos únicos elementos, $\mathcal{F} = \{0,1\}$. Así pues, un codificador de bloques binario opera asignando a cada una de las 2^k posibles secuencias de k símbolos a su entrada una secuencia distinta de n símbolos binarios ($n > k$) seleccionada libremente entre las 2^n disponibles.

En el conjunto $\{0,1\}$, la operación + (suma) definida por la tabla siguiente

$+$	0	1
0	0	1
1	1	0

posee las siguientes propiedades:

a) es asociativa: $\forall a, b, c \in \{0,1\}, a + (b + c) = (a + b) + c$;

b) existe elemento neutro, el 0: $\forall a \in \{0,1\}, a + 0 = 0 + a = a$;

c) todo elemento posee simétrico (opuesto), pues $1 + 1 = 0$; y

d) es conmutativa: $\forall a, b \in \{0,1\}, a + b = b + a$.

Lo que significa que $(\{0,1\}, +)$ es un grupo abeliano o conmutativo.

Análogamente, para la operación $\cdot$ (producto) definida por la tabla

$\cdot$	0	1
0	0	0
1	0	1

se satisfacen las propiedades conmutativa y asociativa, 1 es el elemento neutro (o unidad) y también el simétrico (inverso) de sí mismo. Por tanto, $(\{1\}, \cdot)$ es también un grupo abeliano.

Además, el producto es distributivo respecto de la suma, pues para cualesquiera $a, b, c \in \{0,1\}, a \cdot (b + c) = (a \cdot b) + (a \cdot c)$. Por todo lo cual, si se consideran conjuntamente ambas operaciones, $(\{0,1\}, +, \cdot)$ es un cuerpo conmutativo con dos elementos, que simbolizaremos por GF(2).[3] Un cuerpo con un número finito de elementos se denomina cuerpo finito o cuerpo de Galois. Esta clase de objetos matemáticos desempeñan un papel fundamental en la construcción de códigos de control de errores y de códigos criptográficos, así como también en otras áreas del álgebra como la geometría

[3]Es evidente que GF(2) es el cuerpo con menor número de elementos que es posible construir, y que los símbolos "0" y "1" son arbitrarios; todo cuerpo binario consta del elemento neutro para la suma y del elemento identidad para la multiplicación. Ocurre, en general, que dos cuerpos finitos con el mismo número de elementos son isomorfos, y resulta entonces irrelevante la manera de simbolizar esos elementos. Si existe, por tanto, un cuerpo con q elementos es, salvo isomorfismos, único y se representa por GF(q). Aunque en este libro se tratará de los cuerpos finitos en un capítulo posterior, es procedente señalar aquí que no todo conjunto con un número finito de elementos puede ser dotado de estructura de cuerpo: se puede demostrar que sólo existen los cuerpos finitos GF(p^m) en donde p es un número primo y m es un número natural. El concepto algebraico de cuerpo formaliza y extiende la idea de un conjunto numérico en el que las operaciones de suma, resta, multiplicación y división entre sus elementos están bien definidas, salvo la división por cero.

finita, la combinatoria o la teoría de números. En el capítulo 9 se estudiará detenidamente cuál es su estructura y cómo representar sus elementos.

Las operaciones + y · son, respectivamente, las conocidas operaciones aritméticas de suma y multiplicación módulo-2 con números enteros, o también las operaciones lógicas O-EXCLUSIVO (XOR) e Y (AND) del álgebra de Boole. Al manejar expresiones con esta clase de operadores resulta útil recordar la propiedad de que la suma y la resta módulo-2 son la misma operación, debido a que 1 es su propio elemento opuesto.

Cuando se extiende la operación de suma módulo-2 (+) en GF(2) a los vectores de k elementos de $\{0,1\}$ sin más que definir la suma vectorial por

$$(u_1, u_2, \ldots, u_k) + (v_1, v_2, \ldots v_k) = (u_1 + v_1, u_2 + v_2, \ldots, u_k + v_k);$$

y cuando, partiendo de la multiplicación $(\cdot)$, se define la operación externa del producto entre un elemento a de $\{0,1\}$ y una secuencia $\mathbf{u}$ de $\{0,1\}^k$ como

$$a \cdot (u_1, u_2, \ldots, u_k) = (a \cdot u_1, a \cdot u_2, \ldots, a \cdot u_k)$$

entonces la terna $\mathcal{L}_k = (\{0,1\}^k, +, \cdot)$ es un espacio vectorial de dimensión k sobre GF(2). Es inmediato comprobar la axiomática de espacio vectorial:

a) $(\{0,1\}^k, +)$ es un grupo conmutativo, el producto directo de orden k del grupo abeliano $(\{0,1\}, +)$;

b) $\forall a \in \{0,1\},\ \mathbf{x}, \mathbf{y} \in \mathcal{L}_n,\ \ a \cdot (\mathbf{x} + \mathbf{y}) = a \cdot \mathbf{x} + a \cdot \mathbf{y}$;

c) $\forall a, b \in \{0,1\},\ \mathbf{x} \in \mathcal{L}_n,\ \ (a + b) \cdot \mathbf{x} = a \cdot \mathbf{x} + b \cdot \mathbf{x}$;

d) $\forall a, b \in \{0,1\},\ \mathbf{x} \in \mathcal{L}_n,\ \ (a \cdot b) \cdot \mathbf{x} = a \cdot (b \cdot \mathbf{x})$;

e) si 1 es el elemento unitario de GF(2), $\ 1 \cdot \mathbf{x} = \mathbf{x}, \quad \forall \mathbf{x} \in \mathcal{L}_n$.

Expresado de manera equivalente: el conjunto $\mathrm{GF}(2)^k$ de secuencias de k símbolos binarios junto con las operaciones de suma y producto arriba definidas es un espacio vectorial de dimensión k sobre el cuerpo binario GF(2). El número de vectores de $\mathcal{L}_k$ es 2^k.

Para simplificar la notación, se suprimirá el signo "·" en la operación producto de un escalar por un vector cuando no haya lugar a la confusión.

6.2. Códigos lineales

Definición 6.1 (Código lineal). *Un código de bloques binario $\mathcal{C}[n,k]$ es un código lineal si el conjunto de palabras del código es un subespacio vectorial de $\mathcal{L}_n$ de dimensión k.*

Mensaje	Palabra del código	Mensaje	Palabra del código
0000	0000000	1000	1000110
0001	0001101	1001	1001011
0010	0010111	1010	1010001
0011	0011010	1011	1011100
0100	0100011	1100	1100101
0101	0101110	1101	1101000
0110	0110100	1110	1110010
0111	0111001	1111	1111111

Tabla 6.1. $\mathcal{H}_3$, un código lineal binario $[7, 4]$.

Son consecuencia inmediata de la definición, para cualquier código lineal $\mathcal{C}[n, k]$, las siguientes propiedades:

1. $\mathbf{0}_n$ es una palabra del código, pues todo subespacio vectorial contiene al vector nulo.
2. Cualquier combinación lineal de palabras del código es otra palabra del código. En particular, $\mathcal{C}[n, k]$ es un subgrupo aditivo de $\mathcal{L}_n$.

De hecho, la segunda condición (que contiene a la primera) es condición necesaria y suficiente para que un código sea lineal.

El hecho de que el conjunto de secuencias de longitud finita con elementos pertenecientes a un cuerpo finito pueda ser dotado de estructura de espacio vectorial permite estudiar los códigos lineales aplicando los conceptos y las técnicas del álgebra lineal. Conviene recordar, sin embargo, que no se estudian porque sean los códigos más potentes conocidos (aunque, como se verá, muchos de los códigos óptimos descubiertos —óptimos de acuerdo con algún criterio matemático o aplicado bien definido— sí son lineales), sino por el doble motivo que se señaló en la introducción:

a) Por simplicidad, al no contar todavía con métodos de análisis generales para los códigos de bloques no lineales.

b) Porque la complejidad de implementar un código de bloques sin estructura matemática interna alguna es prohibitiva. En general, un dispositivo codificador/decodificador de un código de bloques binario no lineal necesitaría almacenar en su memoria un diccionario con las 2^k palabras del código, ocupando en total $n2^k$ bits.

Ejemplo 6.1. El código definido por la tabla 6.1 es un código lineal binario $[7, 4]$, que se identificará por $\mathcal{H}_3$. Es fácil comprobar que la suma de dos palabras del código da siempre otra palabra del código. ■

6.3. Matriz generadora

Todo subespacio lineal de dimensión k queda completamente caracterizado cuando se conocen los k vectores de una de sus bases. Así, si $\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}$ es una base del código $\mathcal{C}[n,k]$, toda palabra del código $\mathbf{x} \in \mathcal{C}[n,k]$ es combinación lineal de los vectores $\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}$:

$$\mathbf{x} = u_1\mathbf{g}_1 + \cdots + u_k\mathbf{g}_k$$

para ciertos escalares $u_i \in \{0,1\}$.

Con notación matricial podemos escribir la expresión anterior como

$$\mathbf{x} = \sum_{i=1}^{k} u_i\mathbf{g}_i = \mathbf{u} \cdot \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{k,1} & g_{k,2} & \cdots & g_{k,n} \end{pmatrix} = \mathbf{u} \cdot G$$

en donde el símbolo $\sum$ debe interpretarse como suma en módulo-2, $\mathbf{u}$ es un vector fila con los escalares u_i por elementos, y G es una matriz cuyas filas son los vectores $\mathbf{g}_i$. Una vez conocida G es inmediato calcular el conjunto de palabras del código: es el subespacio engendrado por sus vectores fila.

Definición 6.2 (Matriz generadora). *Una matriz*

$$G_{k\times n} = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \cdot\cdot \\ \mathbf{g}_k \end{pmatrix} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{k,1} & g_{k,2} & \cdots & g_{k,n} \end{pmatrix}$$

de k vectores linealmente independientes del código $\mathcal{C}[n,k]$ se denomina matriz generadora del código $\mathcal{C}[n,k]$.

Obsérvese que, tal como se ha definido, una matriz generadora sirve para calcular el conjunto de vectores del código, pero no para conocer el mensaje que cada uno representa. Para hacer explícita esta asignación es preciso suponer que G es la matriz asociada a la aplicación lineal de codificación en ciertas bases de los espacios de partida y de llegada.

Por consiguiente, la matriz generadora G de un código lineal no es única al no serlo las bases de las que procede. Un resultado básico del álgebra matricial enseña que, tras realizar una secuencia de operaciones elementales (el método de Gauss) en G:

- intercambiando filas o columnas,

- sumando una fila a otra,
- multiplicando una fila por un escalar no nulo,

siempre es posible encontrar una matriz G', transformada de G, que tenga la estructura

$$G'_{k\times n} = \begin{pmatrix} I_k & A_{k\times(n-k)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 & a_{1,1} & \dots & a_{1,n-k} \\ 0 & 1 & \dots & 0 & a_{2,1} & \dots & a_{2,n-k} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 & a_{k,1} & \dots & a_{k,n-k} \end{pmatrix}$$

donde I_k es la matriz identidad de k filas y k columnas, y $A_{k\times(n-k)}$ es una matriz de k filas y $n-k$ columnas con elementos de GF(2). Una matriz de esta forma se denomina *sistemática*, y un código caracterizado por una matriz sistemática se denomina código sistemático.

La adición a una fila de G de una combinación lineal de las otras filas convierte la base original de $\mathcal{C}[n,k]$ en un sistema equivalente de vectores, dejando invariado el conjunto de palabras del código. El intercambio de dos filas tampoco altera, como es lógico, el conjunto de vectores del código. Ambas operaciones equivalen tan sólo a una permutación en la correspondencia entre los mensajes y las palabras del código. Sin embargo, el intercambio de dos columnas se traduce en una permutación de dos coordenadas en todas las palabras del código.

Se deduce de aquí que G y G' no definen en general la misma función de codificación, aunque sí el mismo conjunto de vectores, salvo intercambio en las posiciones de los símbolos. Habida cuenta de que las secuencias de símbolos de entrada al codificador se suponen equiprobables y estadísticamente independientes, la asignación particular entre los mensajes y un conjunto dado de vectores del código carece de importancia, debido a que con tales hipótesis las propiedades del código relativas a la detección y corrección de errores dependen solamente de cuáles sean los símbolos de los vectores del código, y no del orden particular de sus elementos ni del mensaje al que representan (véase el apartado 6.7).

Por lo anterior, se dice que las matrices G y G' son *equivalentes* y, análogamente, que son equivalentes los códigos que representan.

Las matrices sistemáticas resultan mucho más convenientes desde el punto de vista operacional, porque de la ecuación

$$\mathbf{x} = \mathbf{u}\cdot G' = \mathbf{u}\cdot\begin{pmatrix} I_k & | & A_{k\times(n-k)} \end{pmatrix}$$

se deduce que los k primeros símbolos de la palabra del código $\mathbf{x}$ coinciden ahora con los del vector mensaje $\mathbf{u}$:

$$x_i = u_i, \quad i = 1,\dots,k.$$

Así, para obtener una palabra del código sólo es preciso calcular los $n-k$ símbolos de redundancia $x_{k+1},\ldots,x_n$ con las $n-k$ ecuaciones lineales

$$x_{k+j}=\sum_{i=1}^{k}u_i a_{i,j}\quad j=1,\ldots,n-k.$$

Ejemplo 6.2. Supóngase que la matriz generadora de cierto código lineal binario [4,3] es

$$G_1=\begin{pmatrix}1&0&1&0\\0&1&1&0\\1&0&1&1\end{pmatrix}.$$

Vea que la matriz no es sistemática, y no resulta posible obtener una matriz equivalente sistemática realizando solamente operaciones elementales con las filas.

Pero sumando a la tercera fila la primera resulta la matriz equivalente

$$G_2=\begin{pmatrix}1&0&1&0\\0&1&1&0\\0&0&0&1\end{pmatrix}$$

e intercambiando las columnas tercera y cuarta de G_2, el resultado

$$G_3=\begin{pmatrix}1&0&0&1\\0&1&0&1\\0&0&1&0\end{pmatrix}$$

es una matriz sistemática equivalente a G_1. En la tabla siguiente se puede comprobar cómo las palabras del código definidas por G_1 y G_3 no coinciden. Si se intercambian las coordenadas tercera y cuarta de los vectores en la última columna, se obtienen los mismos vectores del código que con G_1, aunque corresponden a mensajes diferentes.

Mensaje ($\mathbf{u}$)	($\mathbf{u}G_1$)	($\mathbf{u}G_3$)
000	0000	0000
001	1011	0010
010	0110	0101
011	1101	0111
100	1010	1001
101	0001	1011
110	1100	1100
111	0111	1110

■

Reuniendo los conceptos presentados hasta el momento, se ha probado entonces que:

TEOREMA 6.1. *Todo código lineal es equivalente a un código lineal sistemático.*

Luego, para cualquier matriz generadora G de un código lineal, existen una matriz no singular F y una matriz de permutación P tales que el producto $F \cdot G \cdot P$ da como resultado una matriz sistemática. Por ello, en todo lo que sigue no se pierde nada esencial por suponer, cuando así convenga, que los códigos objeto de estudio son sistemáticos y que las matrices generadoras son también sistemáticas.

6.4. Matriz de comprobación de paridad

Otra forma útil de caracterizar un código lineal $\mathcal{C}[n,k]$ es a través de cierto subespacio lineal unívocamente relacionado con él.

DEFINICIÓN 6.3 (CONJUNTO ORTOGONAL). *Se llama conjunto ortogonal de un código lineal al conjunto $\mathcal{C}^{\perp}$ formado por los vectores ortogonales a todas las palabras del código:*

$$\mathcal{C}^{\perp} = \big\{\mathbf{x} \in \mathcal{L}_n : \langle \mathbf{x}, \mathbf{y}\rangle = 0,\ \forall\, \mathbf{y} \in \mathcal{C}[n,k]\big\}.$$

El signo $\langle \mathbf{x}, \mathbf{y}\rangle$ indica el producto escalar o interno[4] de los vectores $\mathbf{x}$ e $\mathbf{y}$, definido por $\langle \mathbf{x}, \mathbf{y}\rangle = \sum_{i=1}^{n} x_i y_i$, donde suma y producto deben entenderse como operaciones en módulo-2. Con la terminología usual, se dirá que dos vectores $\mathbf{x}$ e $\mathbf{y}$ son ortogonales si $\langle \mathbf{x}, \mathbf{y}\rangle = 0$.

[4]En un espacio vectorial real V, un producto escalar es cualquier aplicación

$$\begin{aligned} \langle , \rangle : V \times V &\longrightarrow \mathbb{R} \\ (\mathbf{x}, \mathbf{y}) &\longrightarrow \langle \mathbf{x}, \mathbf{y}\rangle \end{aligned}$$

que satisface las propiedades:

1. $\langle \mathbf{x}, \mathbf{y} + \mathbf{z}\rangle = \langle \mathbf{x}, \mathbf{y}\rangle + \langle \mathbf{x}, \mathbf{z}\rangle$; $\langle \mathbf{x} + \mathbf{y}, \mathbf{z}\rangle = \langle \mathbf{x}, \mathbf{z}\rangle + \langle \mathbf{y}, \mathbf{z}\rangle$ (es bilineal).
2. $\langle \mathbf{x}, \mathbf{y}\rangle = \langle \mathbf{y}, \mathbf{x}\rangle$ (es simétrica).
3. $\langle \mathbf{x}, \mathbf{x}\rangle \geq 0$, $\langle \mathbf{x}, \mathbf{x}\rangle = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}$ (es definida positiva).

Como consecuencia inmediata de esta última se tienen que: a) dos vectores ortogonales no nulos son linealmente independientes; b) el producto escalar induce una norma en V, $\|\mathbf{x}\| = \langle \mathbf{x}, \mathbf{x}\rangle^2$.

En el espacio de vectores binarios $\mathcal{L}_n$, el producto escalar $\langle \mathbf{x}, \mathbf{y}\rangle = \sum x_i y_i$ define una forma bilineal simétrica no degenerada pero no definida positiva, de modo que ni la ortogonalidad implica independencia lineal ni existe una norma deducida del producto escalar.

No entraña dificultad comprobar que, por la linealidad del producto escalar, si $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{C}^\perp$ entonces $\mathbf{x}_1 + \mathbf{x}_2 \in \mathcal{C}^\perp$, y que $\mathbf{0}_n \in \mathcal{C}^\perp$; esto es, que $\mathcal{C}^\perp$ es un subespacio vectorial de $\mathcal{L}_n$, y por tanto un código lineal. Según se desprende de la definición, los vectores de $\mathcal{C}^\perp$ son las soluciones, $\mathbf{x}$, del sistema de ecuaciones lineal y homogéneo

$$\langle \mathbf{x}, \mathbf{g}_i \rangle = 0, \qquad i = 1, \ldots, k$$

que consta de k ecuaciones independientes y de n incógnitas, en donde $\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}$ es una base de $\mathcal{C}$. Por lo tanto, $\mathcal{C}^\perp$ tiene longitud n y dimensión

$$n - \text{rango}(\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}) = n - \text{rango}(G) = n - k.$$

Así pues, siempre es posible localizar $n - k$ vectores linealmente independientes y ortogonales a $\mathcal{C}[n, k]$.

Teorema 6.2 (Código dual). *El conjunto ortogonal $\mathcal{C}^\perp$ de un código lineal $\mathcal{C}[n, k]$ es un código lineal $[n, n-k]$, al que se denomina código dual de $\mathcal{C}[n, k]$.*

Es obvio que la relación de dualidad es simétrica: $(\mathcal{C}^\perp)^\perp = \mathcal{C}$. Pero conviene señalar que un código lineal $\mathcal{C}[n, k]$ y su dual no siempre son disjuntos (sin incluir al vector nulo), porque, en los espacios vectoriales construidos sobre cuerpos finitos, la ortogonalidad no implica independencia lineal. Por ejemplo, existen vectores $\mathbf{x} \in \mathcal{L}_n$ no nulos tales que $\langle \mathbf{x}, \mathbf{x} \rangle = 0$. Por esta razón pueden presentarse situaciones en las que $\mathcal{C} \subseteq \mathcal{C}^\perp$, es decir, que el código dual contenga (o sea igual) a $\mathcal{C}[n, k]$. Cuando un código coincida con su dual se dirá de él que es *autodual*; y cuando $\mathcal{C} \subset \mathcal{C}^\perp$, y $\mathcal{C} \neq \mathcal{C}^\perp$, es decir, cuando esté estrictamente contenido en su dual, se dirá que es *auto–ortogonal*. Adviértase que, para cualquier código autodual, la longitud n es par y la dimensión es $n/2$; que para cualquier código auto–ortogonal, $k/n < \frac{1}{2}$; y que si un código es autodual (auto–ortogonal), entonces todas las palabras del código son ortogonales entre sí, $\langle \mathbf{x}, \mathbf{y} \rangle = 0 \quad \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{C}$. Si el código es binario, la condición $\langle \mathbf{x}, \mathbf{x} \rangle = 0$ equivale a decir que todas las palabras del código no nulas tienen un número par de unos.

Ejemplo 6.3. El código $[8, 4]$ generado por

$$G = \left(\begin{array}{c|cccc} & 0 & 1 & 1 & 1 \\ I_4 & 1 & 0 & 1 & 1 \\ & 1 & 1 & 0 & 1 \\ & 1 & 1 & 1 & 0 \end{array} \right)$$

es autodual. ■

Si H es una matriz generadora del código dual, matriz que tendrá $n-k$ filas y n columnas, como, por construcción, las palabras del código dual son ortogonales a las de $\mathcal{C}[n,k]$, ocurre, en particular, que todos los vectores fila de G, una matriz generadora de $\mathcal{C}[n,k]$, son ortogonales a las filas de H (a las columnas de H^T). En consecuencia,

$$G \cdot H^T = [0]_{k \times n-k}.$$

Definición 6.4 (Matriz de comprobación de paridad). *Supuesto que $G_{k\times n}$ es matriz generadora de un código lineal $\mathcal{C}[n,k]$, se llama matriz de comprobación de paridad del código a toda matriz $H_{n-k\times n}$ de rango $n-k$ ortogonal a G; es decir, a cualquier matriz de rango máximo que verifique la identidad*

$$G \cdot H^T = [0]_{k \times n-k}.$$

Conviene mencionar que la matriz de comprobación de paridad define inequívocamente al código, y proporciona entonces la misma información acerca de él que la matriz generadora G; es decir, todo parámetro de interés deducible de G es también deducible de H, y a la inversa. En numerosas ocasiones resulta más fácil analizar las propiedades del código dual que las del código original, sobre todo cuando $n-k \ll k$.

La ortogonalidad entre un código lineal y su dual puede escribirse de manera concisa con la expresión

$$\mathbf{x} \in \mathcal{C}[n,k] \Leftrightarrow \mathbf{x} \cdot H^T = \mathbf{0}_{n-k}. \tag{6.1}$$

Estas $n-k$ identidades expresan el hecho conocido de que en la operación de codificación se ha introducido de forma controlada cierto grado de dependencia (lineal) entre los n símbolos de una palabra del código. Proporcionan, además, una condición necesaria y suficiente de pertenencia a un código lineal.

Si se consideran los elementos $\mathbf{y} = (y_1, \ldots, y_n)$ de un vector de longitud n como incógnitas, entonces la expresión

$$\mathbf{y}H^T = \mathbf{0}$$

o equivalentemente el sistema de ecuaciones

$$\sum_{i=1}^{n} y_i h_{ji} = 0, \quad j = 1, 2, \ldots, n-k$$

son las *ecuaciones de comprobación de paridad*.

En caso de manejar un código sistemático, una matriz de comprobación de paridad se obtiene de manera directa a partir de su matriz generadora sistemática G.

Teorema 6.3. *Si $G = [I_k \,|\, A]$ es una matriz generadora sistemática de $\mathcal{C}[n,k]$, entonces $H = [-A^T \,|\, I_{n-k}]$ es una matriz generadora para $\mathcal{C}^\perp$.*

Demostración. Sea $G = [I \,|\, A]$ la matriz generadora sistemática del código $\mathcal{C}[n,k]$. El código lineal generado por $H = [-A^T \,|\, I_{n-k}]$ tiene parámetros $[n, n-k]$. Como además

$$GH^T = \begin{pmatrix} I_k & A_{k\times n-k} \end{pmatrix} \begin{pmatrix} -A_{k\times n-k} \\ I_{n-k} \end{pmatrix} = -A + A = 0$$

resulta que los vectores fila de G son ortogonales a los de H y, en consecuencia, todas las palabras del código que H define son ortogonales a las palabras de $\mathcal{C}[n,k]$. Se infiere, pues, que H es matriz generadora del código dual $\mathcal{C}^\perp$. ▶

La estructura explícita de H deducida de una generadora sistemática es

$$H_{n-k\times n} = \begin{pmatrix} -a_{1,1} & \dots & -a_{k,1} & 1 & 0 & \dots & 0 \\ -a_{1,2} & \dots & -a_{k,2} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ -a_{1,n-k} & \dots & -a_{k,n-k} & 0 & 0 & \dots & 1 \end{pmatrix}.$$

Si se desarrolla para ella la expresión (6.1), las ecuaciones

$$\sum_{i=1}^{n} x_i h_{ji} = 0, \quad j = 1, \dots, n-k$$

significan, sencillamente, que para toda palabra del código, $\mathbf{x}$, ha de ser

$$x_{k+j} - \sum_{i=1}^{k} x_i a_{ij} = 0, \quad j = 1, \dots, n-k. \tag{6.2}$$

Pero estas ecuaciones de comprobación de paridad gozan de una interpretación lógica inmediata. Sea $\mathbf{y}$ una secuencia de longitud n recibida a la salida de un canal a través del cual se transmiten palabras de un código sistemático. El j-ésimo símbolo de redundancia calculado a partir de los k primeros dígitos del vector $\mathbf{y}$ vale $\sum_{i=1}^{k} y_i a_{ij}$; y el j-ésimo símbolo de redundancia recibido es y_{k+j}. La secuencia $\mathbf{y}$ pertenecerá al código solamente si ambos coinciden para $j = 1, \dots, n-k$; pero esto es precisamente lo que significan las ecuaciones de comprobación de paridad

$$y_{k+j} + \sum_{i=1}^{k} y_i a_{ij} = 0, \quad j = 1, \dots, n-k$$

cuando la suma módulo-2 se interpreta como una comparación binaria.

Ejemplo 6.4. Una matriz de comprobación de paridad de $\mathcal{H}_3$, el código lineal binario $[7, 4]$ presentado en el ejemplo 6.1, es

$$H = \begin{pmatrix} -A^T & I_{n-k} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Las ecuaciones de comprobación de paridad para el vector recibido $\mathbf{v}$ se escriben a partir de las filas de H y, en este caso, son

$$\begin{aligned} v_1 + v_3 + v_4 + v_5 &= 0 \\ v_1 + v_2 + v_3 + v_6 &= 0 \\ v_2 + v_3 + v_4 + v_7 &= 0 \end{aligned}$$

El código dual lo componen los vectores

0000000	1011100
1110010	0111001
0101110	1001011
1100101	0010111

Compárense las ecuaciones de comprobación de paridad con las que dan el valor de los dígitos de redundancia v_5, v_6 y v_7. Compruébese de paso, como ejercicio, que $G \cdot H^T = [0]$. ■

6.5. Decodificación por síndrome

Cuando se transmite un vector del código, $\mathbf{x}$, por un canal discreto con ruido, el vector recibido $\mathbf{y}$ no siempre coincide con el transmitido. La diferencia entre el vector que se transmite y el que se obtiene a la salida del canal es un vector $\mathbf{e}$ que tendrá una componente distinta de cero en las posiciones de aquellos símbolos en los que el canal haya introducido errores, lo que podemos representar con la expresión

$$\mathbf{x} + \mathbf{e} = \mathbf{y}.$$

Definición 6.5 (Vector de error). *Se denomina vector de error (o, simplemente, vector error), y se representa con el signo* $\mathbf{e}$*, a la diferencia entre el vector recibido* $\mathbf{y}$ *y el vector* $\mathbf{x}$ *que se ha transmitido:*

$$\mathbf{e} = \mathbf{y} - \mathbf{x}.$$

A la salida del canal, el decodificador, ignorando el valor de $\mathbf{e}$ y de $\mathbf{x}$, dispone tan sólo del vector recibido para inferir la palabra del código transmitida $\mathbf{x}$. ¿Cómo debe realizar tal inferencia? En primer lugar debe decidir la presencia de posibles errores de transmisión, determinando si $\mathbf{y}$ pertenece al código. Cuando el código carece de estructura interna, sólo se puede averiguar esto comparando la secuencia recibida con cada una de las palabras del código, y es evidente que este proceso de búsqueda sólo puede llevarse a cabo en un tiempo reducido si tal número de palabras del código no es demasiado grande. Sin embargo, cuando el código es lineal, existe un método muy simple de decidir cuándo es $\mathbf{y}$ un vector del código: basta con verificar si se cumplen las ecuaciones de comprobación de paridad (6.2),

$$\mathbf{y} \cdot H^T \stackrel{?}{=} \mathbf{0}.$$

El resultado de este cálculo, que constituye el procedimiento básico de decodificación de un código lineal, recibe la denominación de síndrome.

Definición 6.6 (Síndrome). *Sea H una matriz de comprobación de paridad de un código lineal $\mathcal{C}[n,k]$. Se llama síndrome de $\mathbf{x}$, para cualquier $\mathbf{x} \in \mathcal{L}_n$, al vector $\mathbf{s}(\mathbf{x}) = \mathbf{x} \cdot H^T$.*

Habitualmente, el contexto deja claro a qué vector se refiere el síndrome, por lo que $\mathbf{s}(\mathbf{x})$ se representará simplemente por $\mathbf{s}$ si con ello no se introduce confusión.

El síndrome $\mathbf{s}$ de un vector $\mathbf{x}$ tendrá, por consiguiente, $n-k$ elementos, entre los que habrá tantos ceros como ecuaciones de comprobación de paridad se satisfagan para $\mathbf{x}$.

Una consecuencia directa de haber modelado aditivamente los errores de transmisión es que el síndrome de un vector recibido $\mathbf{y}$

$$\mathbf{s} = \mathbf{y} \cdot H^T = (\mathbf{x}+\mathbf{e})H^T = \mathbf{x} \cdot H^T + \mathbf{e} \cdot H^T = \mathbf{0} + \mathbf{e} \cdot H^T = \mathbf{e} \cdot H^T \quad (6.3)$$

depende exclusivamente del vector de error y no de la palabra del código transmitida, $\mathbf{x}$, de modo que $\mathbf{s}$ no es más que una combinación lineal de ciertas columnas de la matriz H tomando como escalares las componentes de $\mathbf{e}$. Utilizando el concepto de síndrome, es posible enunciar de nuevo la condición necesaria y suficiente de pertenencia a un código lineal presentada en el apartado anterior.

Teorema 6.4. *Sea H una matriz de comprobación de paridad de un código lineal $\mathcal{C}[n,k]$; y sea $\mathbf{y}$ un vector de longitud n; $\mathbf{y}$ es una palabra del código $\mathcal{C}[n,k]$ si y sólo si su síndrome es nulo, $\mathbf{y} \cdot H^T = \mathbf{0}_{n-k}$.*

El teorema proporciona un método directo y eficiente para averiguar si una secuencia de símbolos pertenece o no al código: si el síndrome $\mathbf{s}$ del vector recibido $\mathbf{y}$ no es nulo, entonces $\mathbf{y}$ no es una palabra del código, y necesariamente se ha debido producir algún error durante la transmisión. En cambio, cuando $\mathbf{s}$ es nulo, se sabe que el vector recibido pertenece al código, si bien esto no significa siempre ausencia de errores de transmisión, ya que una secuencia de error puede haber transformado la palabra del código emitida en otra palabra del código. Ahora bien, cuando la secuencia de símbolos recibida pertenece al código, el decodificador debe suponer siempre que el canal ha transmitido sin errores, porque, en condiciones normales, se trata del suceso más probable. Se tiene entonces que son precisamente las secuencias de error que transforman una palabra del código en otra las únicas indetectables con este criterio de decodificación.

Por tanto, un síndrome no nulo es condición suficiente para detectar la presencia de errores en la transmisión. Pero, ¿en qué símbolos de la secuencia recibida? Responder a esta cuestión equivale a resolver la ecuación vectorial

$$\mathbf{s} = \mathbf{e} \cdot H^T \tag{6.4}$$

en la que el síndrome $\mathbf{s}$ y la matriz de comprobación de paridad H son conocidos y $\mathbf{e}$ es la incógnita. Según (6.3), esta ecuación dispone de 2^k soluciones diferentes:[5] los 2^k vectores $\mathbf{y} + \mathbf{x}_i$, con $\mathbf{x}_i \in \mathcal{C}$, cuyo síndrome es $\mathbf{s}$. Por consiguiente, el decodificador debe recurrir a alguna regla (bien arbitraria o bien deducida de la adopción de un criterio matemático definido) para seleccionar una solución particular. Nótese que la consecuencia práctica de elegir una solución $\mathbf{e}_0$ del sistema (6.4) es que implica *corregir* el error $\mathbf{e}_0$: el decodificador obtiene la verdadera palabra del código transmitida (el mensaje, en definitiva) si el vector error en el canal es en verdad $\mathbf{e}_0$.

Un criterio de decisión intuitivo y simple es aquél que consiste en elegir la secuencia de error más probable que produce el mismo síndrome que el de la secuencia recibida. En los canales sin memoria, tales como un canal binario simétrico, y suponiendo mensajes equiprobables, esto es equivalente a decidir en favor del vector error con menor número de elementos no nulos. El motivo de la equivalencia de ambas reglas es que la probabilidad de que se produzca un determinado patrón de error depende entonces nada más que del número de símbolos incorrectos, no de su posición (los errores afectan a cada símbolo con igual probabilidad, y son estadísticamente independientes del mensaje y entre sí), y disminuye a medida que aumenta la cantidad de símbolos erróneos (véase el apéndice 5.A).

La selección del vector de error más probable, conocida la secuencia de

[5] Se trata, por supuesto, de la variedad lineal o afín de soluciones $\mathbf{y} + \mathcal{C} = \{\mathbf{y} + \mathbf{x}_i : \mathbf{x}_i \in \mathcal{C}[n,k]\}$, siendo $\mathbf{y}$ una solución particular de la ecuación y $\mathcal{C}[n,k]$ (el conjunto de soluciones de la ecuación homogénea $\mathbf{e} \cdot H^T = 0$) la dirección de la variedad.

símbolos de salida del canal, se conoce como *regla de máxima verosimilitud* (o regla ML, por *Maximum Likelihood*); y se puede demostrar que, con las hipótesis que se han establecido, es la óptima cuando se pretende minimizar la probabilidad de error de decodificación. En general, la dependencia funcional entre esta probabilidad y el conjunto de palabras del código es complicada. Aun cuando no se tratarán otras situaciones en este libro, es oportuno advertir, no obstante, que el criterio de decodificación de máxima verosimilitud no es necesariamente óptimo cuando los mensajes no son igualmente probables.

Una vez estimado el vector de error, el decodificador finaliza entregando como estimación del vector del código transmitido la suma del vector recibido y el vector de error más probable de entre aquéllos con síndrome igual al de la secuencia recibida. Como el síndrome es independiente del vector del código transmitido, también lo es el vector de error que se elige con la regla de decodificación ML.

Según el criterio de máxima verosimilitud, el algoritmo de decodificación consta, entonces, de los siguientes pasos:

1. Calcular el síndrome $\mathbf{s}$ del vector recibido $\mathbf{y}$, $\mathbf{s} = \mathbf{y} \cdot H^T$.
2. Calcular el vector de error $\mathbf{e}_{\min}$ con menor número de elementos distintos de cero tal que $\mathbf{s} = \mathbf{e}_{\min} \cdot H^T$.
3. Estimar la palabra del código transmitida como $\mathbf{x}_{\text{est}} = \mathbf{y} - \mathbf{e}_{\min}$.

Para aplicar el algoritmo es suficiente con almacenar, en el extremo receptor, en una tabla indexada por el valor del síndrome, su correspondiente vector de error asociado con menor número de elementos no nulos, que se habrá calculado previamente. Esta tabla es la *tabla de decodificación por síndrome*; y una vez construida, todo lo que el receptor debe hacer es consultarla, tras obtener el síndrome del vector recibido, para averiguar una estimación del vector de error. Se necesitan $n2^{n-k}$ bits para almacenar la tabla de decodificación de un código lineal binario $[n, k]$.

6.6. Matriz típica

Sea $\mathcal{C}[n, k]$ un código lineal y considérese la clase de los conjuntos (*co-grupos*) de vectores de $\mathcal{L}_n$ de la forma

$$\mathbf{z} + \mathcal{C} = \left\{ \mathbf{z} + \mathbf{x} \ : \ \mathbf{x} \in \mathcal{C}[n, k] \right\}$$

en donde, para $\mathbf{z} \neq \mathbf{0}$, $\mathbf{z}$ es un vector de $\mathcal{L}_n$ que no pertenece a $\mathcal{C}$; y para $\mathbf{z} = \mathbf{0}$, el conjunto $\mathbf{0} + \mathcal{C}$ es simplemente el código $\mathcal{C}$.

Dados dos vectores arbitrarios, $\mathbf{z}_1$ y $\mathbf{z}_2$, si los conjuntos

$$\mathbf{z}_1 + \mathcal{C} = \left\{\mathbf{z}_1 + \mathbf{x} \ : \ \mathbf{x} \in \mathcal{C}[n,k]\right\}$$
$$\mathbf{z}_2 + \mathcal{C} = \left\{\mathbf{z}_2 + \mathbf{x} \ : \ \mathbf{x} \in \mathcal{C}[n,k]\right\}$$

tuviesen en común algún vector $\mathbf{u}$, entonces existirían dos vectores del código, $\mathbf{w}_1$ y $\mathbf{w}_2$, tales que

$$\mathbf{u} = \mathbf{z}_1 + \mathbf{w}_1 = \mathbf{z}_2 + \mathbf{w}_2$$

y, por tanto, $\mathbf{z}_1 - \mathbf{z}_2 = \mathbf{w}_2 - \mathbf{w}_1 = \mathbf{w}$ sería una palabra del código. Pero en este supuesto se podría escribir que $\mathbf{z}_2 = \mathbf{z}_1 - \mathbf{w} \in \mathbf{z}_1 + \mathcal{C}$ y, en consecuencia, que $\mathbf{z}_1 + \mathcal{C} = \mathbf{z_2} + \mathcal{C}$. Se presentan así dos únicas alternativas: los cogrupos $\mathbf{z}_1 + \mathcal{C}$ y $\mathbf{z}_2 + \mathcal{C}$ o bien son disjuntos o bien coinciden, y lo último ocurre si y solamente si $\mathbf{z}_1 - \mathbf{z}_2 \in \mathcal{C}$, es decir, si la diferencia entre dos representantes es un vector del código.

De acuerdo con esto, es claro que la unión de todos los posibles cogrupos es el propio espacio de vectores $\mathcal{L}_n$:

$$\mathcal{L}_n = \cup_j \left(\mathbf{z}_j + \mathcal{C}\right) \cup \mathcal{C}, \quad \mathbf{z}_j \in \mathcal{L}_n, \mathbf{z}_j \notin \mathcal{C}.$$

Y puesto que existen 2^n vectores binarios de n elementos, y cada cogrupo contiene 2^k vectores, se deduce que deben existir un total de 2^{n-k} cogrupos diferentes que constituyen una partición de $\mathcal{L}_n$ con respecto a $\mathcal{C}$

$$\mathcal{L}_n = \cup_j \left(\mathbf{r}_j + \mathcal{C}\right), \quad (\mathbf{r}_i + \mathcal{C}) \cap (\mathbf{r}_j + \mathcal{C}) = \emptyset, \quad i,j = 1, \ldots, 2^{n-k}, \quad i \neq j$$

partición que se representa como $\mathcal{L}_n/\mathcal{C}$. Los vectores $\mathbf{r}_j$, $1 \leq j \leq 2^{n-k}$, son los *representantes* de las clases de equivalencia de la partición.

Construyamos ahora una tabla cuya primera fila esté formada por todas las palabras del código $\mathcal{C}$, comenzando por el vector nulo. En la primera columna de la fila siguiente incluyamos un representante no nulo cualquiera de la partición $\mathcal{L}_n/\mathcal{C}$, y completemos la fila sumando cada una de las palabras del código escritas en la primera fila a este vector y anotando el resultado en la columna correspondiente a cada palabra del código. Repitamos el mismo procedimiento, eligiendo para cada fila un representante distinto, hasta completar 2^{n-k} filas en la tabla, de manera que hayamos escrito cada cogrupo en una fila

$$\begin{array}{ccccc}
\mathbf{0}_n = \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \ldots & \mathbf{c}_{2^k} \\
\mathbf{r}_2 & \mathbf{c}_2 + \mathbf{r}_2 & \mathbf{c}_3 + \mathbf{r}_2 & \ldots & \mathbf{c}_{2^k} + \mathbf{r}_2 \\
\ldots & \ldots & \ldots & \ldots & \ldots \\
\mathbf{r}_{2^{n-k}} & \mathbf{c}_2 + \mathbf{r}_{2^{n-k}} & \mathbf{c}_3 + \mathbf{r}_{2^{n-k}} & \ldots & \mathbf{c}_{2^k} + \mathbf{r}_{2^{n-k}}
\end{array}$$

Esta matriz de vectores constituye la *matriz típica* del código. En ella, por construcción, la diferencia entre dos vectores de una misma fila es una

palabra del código, y la diferencia entre dos vectores de una misma columna es un vector independiente de las palabras del código. La suma de dos vectores cualesquiera de una misma fila es una palabra del código sólo si el alfabeto de codificación es binario.

Pues bien, partiendo de que el conjunto de vectores binarios de longitud n junto con la operación de suma forman un grupo aditivo del que $\mathcal{C}[n,k]$ es un subgrupo (es decir, sin más que recurrir a las propiedades de la suma de vectores binarios), es posible establecer una propiedad fundamental de la matriz típica, que aclara la relación directa entre los síndromes y los errores en el canal.

Teorema 6.5. *Todos los vectores de una misma fila de la matriz típica tienen el mismo síndrome, y el síndrome asociado a cada fila es distinto.*

Demostración. Cualquier vector de la fila m es la suma de un representante $\mathbf{r}_m$ y una palabra del código. En estas condiciones, su síndrome depende solamente de $\mathbf{r}_m$. Además, si dos filas, i y j, tuviesen el mismo síndrome, $\mathbf{r}_i H^T = \mathbf{r}_j H^T$. Luego $(\mathbf{r}_i - \mathbf{r}_j)H^T = \mathbf{0}$, y el vector $\mathbf{r}_i - \mathbf{r}_j$ sería una palabra del código, $\mathbf{x}_k$. Pero esto es imposible, porque, de ser cierto, el vector $\mathbf{r}_i = \mathbf{r}_j + \mathbf{x}_k$ debería figurar ya en la fila j-ésima, y no podría haber sido elegido como representante para encabezar la fila i $(i > j)$. ▶

Expresado de manera menos formal, este teorema constata que existe una relación uno a uno entre los 2^{n-k} síndromes y las filas de la matriz típica, es decir, cada cogrupo está asociado a un único síndrome. Ahora resulta claro que la matriz típica tabula todas las soluciones de la ecuación $\mathbf{s} = \mathbf{y} \cdot H^T$ para cada posible valor de $\mathbf{s}$, y que cada una de las filas contiene todos los vectores que producen un determinado vector síndrome. Así pues, dado un síndrome $\mathbf{s}$, su fila en la matriz típica contiene todos los vectores de error que pueden haberse producido durante la transmisión de una palabra del código por el canal.

La importancia de la matriz típica para el proceso de decodificación de códigos lineales reside, entonces, en el hecho de que representa una partición del conjunto de vectores de $\mathcal{L}_n$ (los posibles vectores recibidos) según el síndrome que generan. Más concretamente, cada fila $\mathbf{r}_i + \mathcal{C}$ de la matriz típica es una clase de equivalencia de la relación binaria $\mathcal{R}$ entre vectores definida por el predicado[6] $\mathbf{x}\,\mathcal{R}\,\mathbf{y} \Leftrightarrow \mathbf{x}\cdot H^T = \mathbf{y}\cdot H^T$, clase a la que pertenecen todos los vectores con síndrome igual a $\mathbf{s}_i = \mathbf{r}_i \cdot H^T$ o, lo que es lo mismo, todas las soluciones $\mathbf{y}$ de la ecuación $\mathbf{s}_i = \mathbf{y} \cdot H^T$.

[6] O también $\mathbf{x}\,\mathcal{R}\,\mathbf{y} \Leftrightarrow \mathbf{x} - \mathbf{y} \in \mathcal{C}[n,k]$. Las filas de la matriz típica de un código lineal $\mathcal{C}$ son las clases adjuntas del grupo cociente $\mathcal{L}_n/\mathcal{C}$.

Palabras del código			
00000	01011	10101	11110
00001	01010	10100	11111
00010	01001	10111	11100
00100	01111	10001	11010
01000	00011	11101	10110
10000	11011	00101	01110
11000	10011	01101	00110
10010	11001	00111	01100

Tabla 6.2. Matriz típica para el código $[5, 2]$ del ejemplo 6.5.

Síndrome	Vector error	Síndrome	Vector error
000	00000	011	01000
001	00001	101	10000
010	00010	110	11000
100	00100	111	10010

Tabla 6.3. Tabla de decodificación por síndrome para el código lineal binario $[5, 2]$ del ejemplo 6.5.

Según lo visto, a la hora de decodificar bastaría con localizar en la matriz típica el vector recibido y estimar como vector de error el que figura en la primera columna de la misma fila. Sin embargo, éste es, simplemente, un vector con igual síndrome que el vector recibido y, en general, no tiene por qué resultar el más conveniente para hacer mínima la probabilidad de error. Pero si se tiene en cuenta el criterio de máxima verosimilitud, y se supone que el canal carece de memoria, la probabilidad de error del decodificador se hace mínima al seleccionar para la primera columna de cada fila de la matriz típica aquel vector de la fila con menos elementos no nulos. O bien, lo que es igual, al situar en la primera posición de cada fila de la matriz típica el vector con menos elementos distintos de cero que no figure aún en ninguna de las filas anteriores. En caso de haber más de uno, la elección entre ellos es indiferente, debido a que implican la corrección del mismo número de símbolos erróneos.

El decodificador puede almacenar solamente los 2^{n-k} vectores de la primera columna junto con su síndrome, y estimar como vector error aquél cuyo síndrome coincida justamente con el del vector recibido.

La tabla que refleja la asociación uno a uno entre síndromes y vectores de error corregibles es la tabla de decodificación por síndrome. El espacio de memoria necesario para almacenarla es, para códigos binarios, de $n2^{n-k}$ bits, el correspondiente a n funciones lógicas de $n - k$ variables.

Ejemplo 6.5. Una matriz típica del código binario lineal $[5, 2]$ de matriz generadora

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

se da en la tabla 6.2, y la tabla 6.3 es la de decodificación por síndrome. En ésta, los vectores de la primera columna son el vector nulo, todos los de 1 bit y dos vectores de 2 bits, los únicos errores corregibles. ■

Ejemplo 6.6. Consideremos de nuevo el código con la matriz de comprobación de paridad dada en el ejemplo 6.4

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

y supongamos que el vector transmitido es $\mathbf{x} = 1111111$ y el recibido es $\mathbf{y} = 1101111$.

Tras observar $\mathbf{y}$, el decodificador calcula el síndrome: $\mathbf{s} = \mathbf{y} \cdot H^T = 111$.

Los vectores de error $\mathbf{e} = (e_1, \ldots, e_7)$ que cumplan

$$\begin{aligned} e_1 + e_3 + e_4 + e_5 &= 1 \\ e_1 + e_2 + e_3 + e_6 &= 1 \\ e_2 + e_3 + e_4 + e_7 &= 1 \end{aligned}$$

poseen todos el síndrome 111. De entre ellos, $\mathbf{e} = 0010000$ es el de menor cantidad de bits distintos de cero; y si el canal es binario simétrico, se trata del error más probable, lo que conduce a estimar como vector transmitido $\hat{\mathbf{x}} = \mathbf{y} + 0010000 = 1111111$. En este caso el receptor ha tomado la decisión correcta.

Supongamos ahora que la palabra del código transmitida hubiese sido $\mathbf{x} = 1000110$ y la secuencia recibida el vector $\mathbf{y} = 0000111$, de suerte que el canal hubiese alterado los bits primero y último de $\mathbf{x}$; o, de manera equivalente, $\mathbf{e} = 1000001$.

En estas condiciones el síndrome de $\mathbf{y}$ vale $\mathbf{s} = 111$, y el vector de error más probable con este síndrome es, como se sabe, $\mathbf{e}_{\min} = 0010000$. El decodificador estima, por tanto, como palabra del código transmitida

$$\mathbf{y} + \mathbf{e}_{\min} = 0010111$$

y comete un error.

La tabla de decodificación por síndrome (tabla 6.4) tiene, para este código, una estructura especialmente sencilla, por ser las columnas de la matriz

$\mathbf{s}$	$\mathbf{e}_{\min}$	$\mathbf{s}$	$\mathbf{e}_{\min}$
000	0000000	100	0000100
001	0000001	101	0001000
010	0000010	110	1000000
011	0100000	111	0010000

TABLA 6.4. Tabla de síndromes para el código del ejemplo 6.6.

H todos los vectores binarios no nulos de tres elementos. En consecuencia, para cualquier síndrome $\mathbf{s}$, el vector error de menor número de elementos no nulos correspondiente a $\mathbf{s}$ es el vector con un 1 en la misma posición que $\mathbf{s}$ ocupa entre las columnas de H y los restantes elementos nulos. El código puede corregir todos los errores de un bit. ■

6.7. Detección y corrección de errores

La cantidad de errores que un código lineal puede corregir aplicando la técnica de decodificación por síndrome depende únicamente del número de dígitos de redundancia, y es independiente del conjunto de palabras del código. Resulta muy sencillo constatarlo.

TEOREMA 6.6. *Un código lineal binario $\mathcal{C}[n,k]$ es capaz de corregir 2^{n-k} vectores de error.*

DEMOSTRACIÓN. La matriz típica permite corregir los 2^{n-k} patrones de error que aparecen en su primera columna, incluido el vector cero, y estima incorrectamente todos los demás vectores de error. ▶

Es obvio también que todos los códigos de bloques (no necesariamente lineales) de longitud n y 2^k palabras detectan el mismo número de errores.

TEOREMA 6.7. *Un código de bloques binario $\mathcal{C}[n,k]$ es capaz de detectar $2^n - 2^k$ vectores de error.*

DEMOSTRACIÓN. De los 2^n posibles vectores binarios recibidos, sólo 2^k son palabras del código. Los $2^n - 2^k$ restantes son siempre detectables, bien por tener síndrome no nulo, si el código es lineal, bien por no pertenecer a $\mathcal{C}[n,k]$, en general, si el código no es lineal. ▶

Los vectores de error indetectables son aquéllos y sólo aquéllos que transforman una palabra del código en otra. Si el objetivo es minimizar la probabilidad de que un error pase inadvertido, entonces es de desear que estos vectores de error tengan la máxima cantidad posible de elementos no nulos, de modo que sea necesario alterar un número elevado de símbolos de una palabra del código para obtener otra palabra del código. Y en el caso particular de que el código sea lineal, los únicos vectores de error indetectables son justamente los que coinciden con alguna palabra del código no nula.

Parece así claro que el subconjunto de errores que es posible detectar y/o corregir debe depender de alguna noción de diferencia o distancia entre las palabras del código. Las dos definiciones siguientes establecen una métrica de distancia.

Definición 6.7 (Peso Hamming). *Se denomina peso (Hamming) de un vector* $\mathbf{x}$*, y se representa por* $p_H(\mathbf{x})$*, al número de elementos no nulos de* $\mathbf{x}$*.*

Definición 6.8 (Distancia Hamming). *La distancia (Hamming) entre dos vectores* $\mathbf{x}$ *e* $\mathbf{y}$ *es el número de símbolos en que difieren*

$$d_H(\mathbf{x},\mathbf{y}) = p_H(\mathbf{x}-\mathbf{y}).$$

Observación: la distancia Hamming es una métrica bien definida en el espacio $\mathcal{L}_n$. Es obvio que $d_H(\mathbf{x},\mathbf{y}) \geq 0$ y que $d_H(\mathbf{x},\mathbf{y}) = d_H(\mathbf{y},\mathbf{x})$, $d_H(\mathbf{x},\mathbf{y}) = 0 \Leftrightarrow \mathbf{x} = \mathbf{y}$, y no presenta especial dificultad comprobar la desigualdad triangular $d_H(\mathbf{x},\mathbf{z}) \leq d_H(\mathbf{x},\mathbf{y}) + d_H(\mathbf{y},\mathbf{z})$. Además, se da la circunstancia de que la distancia Hamming es una métrica invariante por traslación: $d_H(\mathbf{x}+\mathbf{z},\mathbf{y}+\mathbf{z}) = d_H(\mathbf{x},\mathbf{y})$.

Las propiedades de detección estarán entonces dominadas, de alguna forma, por el número mínimo de símbolos que es preciso modificar en una palabra del código para conseguir otra palabra del código. Esta cantidad es el peso Hamming del vector de error con peso Hamming mínimo que convierte una palabra del código en otra, y se la conoce como distancia del código.

Definición 6.9 (Distancia de un código). *Se define la distancia de un código* $\mathcal{C}$ *como la mínima de las distancias entre todos los pares de palabras del código:*

$$d_{\mathcal{C}} = \min\big\{d_H(\mathbf{x},\mathbf{y})\big\}; \quad \forall \mathbf{x},\mathbf{y} \in \mathcal{C}, \quad \mathbf{x} \neq \mathbf{y}.$$

Según esto, para calcular la distancia de un código es suficiente formar

todos los posibles pares de palabras del código diferentes, obtener la distancia Hamming que las separa, y conservar el mínimo de todas las distancias. El procedimiento es trivial, pero impracticable si el número de palabras del código es elevado. Por fortuna, la tarea es algo más sencilla si se tiene en cuenta la linealidad.

TEOREMA 6.8. *La distancia de un código lineal es el peso Hamming de cualquier palabra del código no nula de peso mínimo:*

$$d_{\mathcal{C}} = \min_{\mathbf{x}\in\mathcal{C},\mathbf{x}\neq\mathbf{0}} \{p_H(\mathbf{x})\}.$$

DEMOSTRACIÓN. En un código lineal, la diferencia entre dos palabras del código cualesquiera es otra palabra del código; así que, al formar todos los pares distintos de palabras del código, su diferencia recorrerá todo el conjunto de palabras del código no nulas. ▶

Luego para calcular la distancia de un código lineal basta con conocer las palabras del código y averiguar el peso Hamming mínimo de todas las palabras no nulas.

La distancia es un invariante básico de un código, y adquiere especial importancia a la hora de evaluar la capacidad de detección y corrección de errores. Por eso, cuando la distancia de un código $[n, k]$ sea conocida, se escribirá $[n, k, d_{\mathcal{C}}]$.

TEOREMA 6.9. *Con un código de bloques de distancia $d_{\mathcal{C}}$ se pueden corregir todos los vectores de error* $\mathbf{e}$ *tales que*

$$p_H(\mathbf{e}) \leq \left\lfloor \frac{d_{\mathcal{C}} - 1}{2} \right\rfloor$$

o se pueden detectar todos los vectores de error $\mathbf{e}$ *tales que*

$$p_H(\mathbf{e}) \leq d_{\mathcal{C}} - 1.$$

DEMOSTRACIÓN. Toda vez que se dispone del concepto de distancia Hamming, la regla de decodificación de máxima verosimilitud admite, para canales discretos y sin memoria, una elegante interpretación geométrica: el cálculo del vector error de peso Hamming mínimo equivale a estimar como palabra del código transmitida la más cercana, en distancia Hamming, al vector recibido. Formalmente, la decisión del decodificador será

$$\mathbf{x}_{\text{est}} = \arg\min_{\mathbf{x}\in\mathcal{C}} d_H(\mathbf{x}, \mathbf{y}).$$

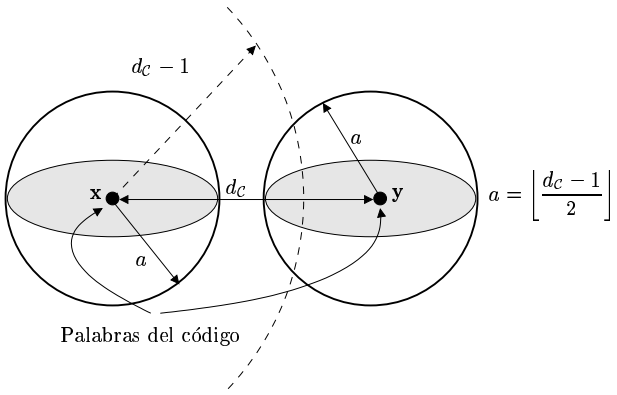


Figura 6.1. Relación geométrica entre la distancia de un código y la capacidad de control de errores.

Sea $\mathbf{x}$ la palabra del código transmitida y $\mathbf{e}$ un vector de error con $p_H(\mathbf{e}) \leq \lfloor \frac{d_{\mathcal{C}}-1}{2} \rfloor$. Se cometería un error de decodificación si existiese otra palabra del código $\mathbf{z} \neq \mathbf{x}$ más cercana que $\mathbf{x}$ al vector recibido $\mathbf{y}$. Pero entonces, aplicando la desigualdad triangular (la distancia Hamming es una métrica), se tendría que

$$d_H(\mathbf{x}, \mathbf{z}) \leq d_H(\mathbf{x}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{z}) \leq \left\lfloor \frac{d_{\mathcal{C}} - 1}{2} \right\rfloor + \left\lfloor \frac{d_{\mathcal{C}} - 1}{2} \right\rfloor \leq d_{\mathcal{C}} - 1.$$

De modo que habría dos palabras del código a distancia menor que $d_{\mathcal{C}}$, lo que por hipótesis es imposible y obliga a rechazar la existencia de otra palabra del código más cercana al vector recibido que $\mathbf{x}$.

Por otra parte, si el peso de $\mathbf{e}$ es menor o igual que $d_{\mathcal{C}} - 1$, resulta imposible transformar una palabra del código en otra, y por tanto el error $\mathbf{e}$ es detectable. ▶

Conviene hacer hincapié en dos observaciones acerca del teorema:

a) El teorema no dice que los únicos errores $\mathbf{e}$ detectables sean los de $p_H(\mathbf{e}) \leq d_{\mathcal{C}} - 1$, ni los únicos corregibles aquéllos de $p_H(\mathbf{e}) \leq \lfloor \frac{d_{\mathcal{C}}-1}{2} \rfloor$; estas expresiones son nada más que condiciones suficientes, no necesarias.

b) El teorema se puede aplicar a cualquier clase de códigos de bloques, ya que en ningún punto de la demostración se hace uso de la hipótesis de linealidad.

Pero en el caso de un código lineal existe, a través del concepto de síndrome, una relación adicional muy útil entre la distancia y la estructura de la matriz comprobadora de paridad.

TEOREMA 6.10. *Sea H la matriz de comprobación de paridad de un código lineal de distancia $d_{\mathcal{C}}$. Cualesquiera $s-1$ columnas de H son linealmente independientes si y solamente si $d_{\mathcal{C}} \geq s$.*

DEMOSTRACIÓN. El síndrome es una combinación lineal de los vectores columna de la matriz H. Luego si el código es de distancia $d_{\mathcal{C}}$, cualquier vector error de peso menor que $d_{\mathcal{C}}$ es detectable y su síndrome $\mathbf{s} = \mathbf{e} \cdot H^T$ es no nulo. Es decir, cualquier conjunto de menos de $d_{\mathcal{C}}$ columnas de H es linealmente independiente. ▶

COROLARIO 6.11. *Se verifica que $d_{\mathcal{C}} = s$ si y solamente si*

a) cualquier conjunto de $s-1$ columnas de H es linealmente independiente, y

b) existe un conjunto de s columnas de H linealmente dependientes.

DEMOSTRACIÓN. Es claro que

$$d_{\mathcal{C}} = s \Leftrightarrow d_{\mathcal{C}} \geq s, d_{\mathcal{C}} < s+1.$$

Pero si $d_{\mathcal{C}} \geq s$, cualquier subconjunto de $s-1$ columnas de H es, por el teorema anterior, linealmente independiente; y, según el mismo argumento, si $d_{\mathcal{C}} < s+1$, no todos los grupos de s columnas de H pueden ser linealmente independientes, porque entonces se cumpliría $d_{\mathcal{C}} \geq s+1$. ▶

La aplicación práctica del corolario es obvia. La distancia mínima de un código lineal se puede obtener también calculando el número mínimo de columnas de la matriz comprobadora de paridad que son linealmente dependientes. En un código binario esto equivale a descubrir s columnas cuya suma sea el vector nulo, algo que puede hacerse por inspección si el número de símbolos de redundancia y la longitud del código no son demasiado grandes. El ejemplo siguiente ilustra el procedimiento a seguir.

EJEMPLO 6.7. La matriz de comprobación de paridad del código introducido en el ejemplo 6.1 es

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

Ninguna de sus columnas es el vector cero, así que la distancia del código debe ser mayor que 1. Pero tampoco hay dos columnas iguales, lo que significa que la suma de dos de ellas nunca puede dar como resultado el vector nulo. Entonces, la distancia del código es al menos 3. En cambio sí hay varios grupos de tres columnas que suman cero; por ejemplo, la primera, la quinta y la sexta. Llegamos así, finalmente, a la conclusión de que el código posee distancia 3, y corroboramos este cálculo viendo que en la tabla 6.1 todas las palabras del código distintas de cero tienen no menos de tres unos. En virtud del teorema 6.9, todos los errores de un bit se pueden corregir y todos los de dos bits se pueden detectar. Como existen en total 7 vectores de error de peso 1, y hay 8 síndromes distintos, se concluye que sólo los errores simples son corregibles. ■

Otra consecuencia directa del teorema 6.9 es que, para códigos lineales, todos los vectores no nulos de peso menor o igual que $\lfloor \frac{d_{\mathcal{C}}-1}{2} \rfloor$ deben aparecer en filas distintas de la matriz típica. Lo cual equivale a afirmar que tienen síndromes distintos. El teorema siguiente formaliza esta propiedad, aunque realmente es una versión alternativa del propio teorema 6.9 teniendo en cuenta la linealidad.

Teorema 6.12. *Sea $\mathcal{C}[n,k]$ un código lineal con matriz de comprobación de paridad H. Si*

$$0 < p_H(\mathbf{x}), p_H(\mathbf{y}) \leq \left\lfloor \frac{d_{\mathcal{C}}-1}{2} \right\rfloor \quad \mathbf{x},\mathbf{y} \in \mathcal{L}_n \quad \mathbf{x},\mathbf{y} \neq \mathbf{0}$$

entonces $\mathbf{x}H^T \neq \mathbf{y}H^T$.

Demostración. Supongamos que la tesis no fuese cierta y que existiese un vector $\mathbf{y}$ en la misma fila de $\mathbf{x}$ en la matriz típica tal que

$$p_H(\mathbf{y}) \leq p_H(\mathbf{x}).$$

Pero $\mathbf{x}-\mathbf{y} \in \mathcal{C}[n,k]$ y además

$$p_H(\mathbf{x}-\mathbf{y}) \leq p_H(\mathbf{x}) + p_H(\mathbf{y}) \leq d_{\mathcal{C}} - 1$$

lo que quiere decir que existiría una palabra del código distinta de cero, $\mathbf{x}-\mathbf{y}$, de peso menor que $d_{\mathcal{C}}$. Y esto es imposible en un código lineal de distancia $d_{\mathcal{C}}$. ▶

Cuando lo que se pretende es caracterizar la máxima capacidad de corrección de un código, se ha de analizar otro invariante básico suyo, el *radio de cobertura*.

Definición 6.10 (Radio de cobertura). *El radio de cobertura de $\mathcal{C}$ se define como*

$$\rho(\mathcal{C}) = \max_{\mathbf{x}\in\mathcal{L}_n} \min_{\mathbf{v}\in\mathcal{C}} d(\mathbf{x},\mathbf{v}) = \max_{\mathbf{x}\in\mathcal{L}_n} d(\mathbf{x},\mathcal{C}).$$

Así pues, $\rho(\mathcal{C})$ es el menor entero r tal que las esferas de radio r con centro en los vectores código de $\mathcal{C}$ son un recubrimiento de $\mathcal{L}_n$. En lenguaje más geométrico: ningún vector de $\mathcal{L}_n$ está a distancia mayor que $\rho(\mathcal{C})$ de alguna palabra del código $\mathcal{C}$. Es obvio que

$$\rho(\mathcal{C}) \geq \left\lfloor \frac{d_{\mathcal{C}} - 1}{2} \right\rfloor.$$

En códigos lineales, el radio de cobertura puede hallarse a través de la matriz típica o de una matriz de comprobación de paridad.

Teorema 6.13.

a) Se llama peso de un cogrupo al mínimo peso Hamming de los vectores del cogrupo. $\rho(\mathcal{C})$ es el peso del cogrupo de mayor peso.

b) $\rho(\mathcal{C})$ es el menor entero r tal que cualquier síndrome es combinación lineal de r o menos columnas de H, una matriz de comprobación de paridad de $\mathcal{C}$.

Demostración.

a) Una consecuencia inmediata de la definición.

b) Del apartado anterior y del teorema 6.5. ▶

6.8. Probabilidad de error

La caracterización del conjunto de errores detectables y corregibles con un código exige el conocimiento completo de las palabras del código, por lo que el problema se torna demasiado laborioso en los casos de interés. Es así que, para describir la capacidad de control de errores de un código lineal, normalmente se recurre a resumir todas sus propiedades de detección y corrección en dos parámetros sintéticos: la probabilidad de no detectar un error y la probabilidad de cometer un error de decodificación. Por supuesto, ambas probabilidades dependerán del modelo de canal que se considere. Aquí se supondrá el caso de un canal binario simétrico sin memoria con una probabilidad de error de bit $p < \frac{1}{2}$.

La probabilidad de que se produzca un error de transmisión no detectable por el receptor es, con un código lineal $\mathcal{C}[n,k]$, la probabilidad de que el vector error coincida con alguna de las palabras del código no nulas. Si los coeficientes A_i, $i=1,\ldots,n$, representan el número de palabras de $\mathcal{C}[n,k]$ de peso Hamming i, entonces dicha probabilidad admite la expresión

$$\mathrm{P_{ed}} = \sum_{i=1}^{n} A_i p^i (1-p)^{n-i} \tag{6.5}$$

al haber A_i vectores de error distintos indetectables (tantos como palabras del código) de peso Hamming i, sucediendo cada uno con probabilidad $p^i(1-p)^{n-i}$. La distribución de pesos $\{A_1, A_2, \ldots, A_n\}$, y no las palabras del código individuales, fijan así el valor de la probabilidad de error residual.

En cuanto a la posibilidad de decodificar incorrectamente, se sabe que sucede sólo cuando el vector de error no es corregible, es decir, cuando no es ninguno de los que figuran en la tabla de decodificación por síndrome. Designando por α_i, para $i=0,\ldots,n$, al número de vectores de la tabla de decodificación por síndrome con peso Hamming i, se tiene entonces que

$$\mathrm{P(error\ de\ decodificación)} = \mathrm{P_{ec}} = 1 - \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}. \tag{6.6}$$

Dado que un error no detectado implica una decodificación incorrecta, siempre es $\mathrm{P_{ed}} \leq \mathrm{P_{ec}}$.

Si la distancia del código es $d_{\mathcal{C}} = 2t+1$ o $d_{\mathcal{C}} = 2t+2$, todos los vectores de error de peso menor o igual a t se pueden corregir (teorema 6.9), y entonces

$$\alpha_i = \binom{n}{i}, \quad i = 0,\ldots,t.$$

Pero, por lo general, es muy difícil calcular α_i para $i > t$, y de hecho se desconoce su valor para la mayor parte de las familias de códigos.

Un código capaz de corregir todos los errores de peso Hamming menor o igual que t, y sólo éstos (es decir, tal que $\alpha_i = \binom{n}{i}$, $i=0,\ldots,t$, y $\alpha_i = 0$ para $i > t$), se llama *código perfecto*, en virtud de sus especiales características geométricas: las esferas de radio t centradas en una palabra del código son todas disjuntas y cubren todo el espacio $\mathcal{L}_n$ de vectores binarios. Es decir, en los códigos perfectos se cumple que $\rho(\mathcal{C}) = t$.

Y si es capaz de corregir todos los vectores de error de peso Hamming menor que t, algunos de peso $t+1$ y ninguno de peso mayor que $t+1$ (esto es, $\alpha_i = \binom{n}{i}$, $i=0,\ldots,t$, $0 < \alpha_{t+1} < \binom{n}{t+1}$, y $\alpha_i = 0$ para $i > t+1$), entonces se trata de un código *quasi–perfecto*. En términos geométricos, en

un código quasi–perfecto se cumple que $\rho(\mathcal{C}) = t+1$, de modo que las esferas centradas en una palabra del código con radio $t+1$ se solapan, pero cubren todo el espacio de vectores binarios.

Aplicando, por ejemplo, la fórmula (6.5) al código $\mathcal{H}_3$ (página 144), se obtiene

$$\mathrm{P}_{\mathrm{ed}} = 7p^3(1-p)^4 + 7p^4(1-p)^3 + p^7.$$

Y para particularizar la expresión (6.6), recordemos del ejemplo 6.6 que la tabla de síndromes contiene, para este código, el vector nulo, $\alpha_0 = 1$, y todos los vectores de error de 1 bit, $\alpha_1 = 7$. Por lo tanto

$$\mathrm{P}_{\mathrm{ec}} = 1 - (1-p)^7 - 7p(1-p)^6.$$

Cuando n es grande resulta muy laborioso calcular los valores de A_i, y se recurre a veces a la simplificación consistente en aproximar P_{ed} por el primer o, a lo sumo, los dos primeros términos, que suelen ser los dominantes,

$$\mathrm{P}_{\mathrm{ed}} \approx A_j p^j (1-p)^{n-j} \quad j = \mín\{1 \le i \le n : A_i \neq 0\} = d_{\mathcal{C}}.$$

Esta aproximación está justificada cuando $p \ll 1$.

En cambio, resulta mucho menos complicado estimar la probabilidad promedio de un error no detectado para el conjunto de todos los posibles códigos lineales sistemáticos $[n, k]$. Un código sistemático puede definirse mediante una matriz generadora sistemática. Cada uno de los $k \times (n-k)$ coeficientes de la submatriz A puede valer 0 o 1, y cada combinación da lugar a un código diferente. En total hay, pues, $2^{k(n-k)}$ códigos lineales sistemáticos $[n, k]$. Si se elige uno, $\mathcal{C}_i$, de manera aleatoria con probabilidad $\mathrm{P}(\mathcal{C}_i) = 2^{-k(n-k)}$, y $\{A_{ji}, j = 0, \ldots, n\}$ es su distribución de pesos, la probabilidad de error indetectado sobre un BSC con este código particular valdrá

$$\mathrm{P}(\text{error no detectado} \,|\, \mathcal{C}_i) = \sum_{j=1}^{n} A_{ji} p^j (1-p)^{n-j} \tag{6.7}$$

y la probabilidad de error promedio será

$$\mathrm{P}(\text{error no detectado}) = \sum_{i=1}^{2^{k(n-k)}} \mathrm{P}(\mathcal{C}_i)\, \mathrm{P}(\text{error no detectado} \,|\, \mathcal{C}_i).$$

Introduciendo (6.7) en la expresión anterior, y cambiando el orden de las sumas,

$$\mathrm{P}(\text{error no detectado}) = 2^{-k(n-k)} \sum_{j=1}^{n} p^j (1-p)^{n-j} \sum_{i=1}^{2^{k(n-k)}} A_{ji}. \tag{6.8}$$

Vectores del código	Tabla de síndromes					
	$\mathbf{s}$	$\mathbf{e}_{\text{mín}}$	$p_H(\mathbf{e}_{\text{mín}})$	$\mathbf{s}$	$\mathbf{e}_{\text{mín}}$	$p_H(\mathbf{e}_{\text{mín}})$
0000000	0000	0000000	0	1000	0001000	1
0100111	0001	0000001	1	1001	0001001	2
0011101	0010	0000010	1	1010	0001010	2
0111010	0011	0000011	2	1011	0001011	3
1001110	0100	0000100	1	1100	0001100	2
1101001	0101	0000101	2	1101	0010000	1
1010011	0110	0000110	2	1110	1000000	1
1110100	0111	0100000	1	1111	1000001	2

Tabla 6.5. Código y tabla de síndromes del ejemplo 6.8.

Un vector binario o bien no pertenece a ningún código sistemático (si sus k primeros símbolos son nulos y alguno de los $n-k$ restantes no lo es) o bien está contenido en $2^{(k-1)(n-k)}$ de ellos, porque en cada una de las $n-k$ últimas columnas de la matriz generadora es posible elegir $k-1$ coeficientes libremente. Por haber $\binom{n}{j}$ vectores binarios de peso j, se cumple que

$$\sum_{i=1}^{2^{k(n-k)}} A_{ji} \leq \binom{n}{j} 2^{(k-1)(n-k)}.$$

Incorporando esta expresión en la probabilidad de error promedio (6.8), obtenemos finalmente

$$\begin{aligned} \text{P(error no detectado)} &\leq 2^{-(n-k)} \sum_{j=1}^{n} \binom{n}{j} p^j (1-p)^{n-j} \\ &= 2^{-(n-k)} (1 - (1-p)^n) \\ &\leq 2^{-(n-k)}. \end{aligned}$$

En otras palabras, existen códigos lineales sistemáticos $[n, k]$ en los que la probabilidad de que un error pase inadvertido decrece exponencialmente con el número de bits de redundancia. Aun con valores moderados de $n-k$, este número se hace rápidamente muy pequeño.

Ejemplo 6.8. La matriz generadora de un cierto código lineal binario es

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

y su matriz comprobadora de paridad

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

Es sencillo ver que ninguna combinación de tres columnas de H puede dar el vector 0000, pero que existen combinaciones de cuatro columnas cuya suma sí es el vector nulo; la distancia del código es, por lo tanto, igual a 4. A partir de la matriz comprobadora de paridad podemos construir la tabla de síndromes 6.5. Nótese que, en todas las entradas de la tabla en las que $p_H(\mathbf{e}_{\text{mín}}) > 1$, existe al menos otra solución con el mismo peso Hamming; la que hemos listado es, de entre todas, arbitraria.

En la tabla típica se ve que todos los errores simples son corregibles, como lo son también 7 errores dobles y un error triple. No son corregibles los restantes 14 errores dobles, 34 errores triples, 35 errores cuádruples, 21 errores quíntuples, 7 séxtuples y el vector de error de 7 bits. Sobre un canal binario simétrico, la probabilidad de un error no detectado es la probabilidad de que el vector error coincida con alguna de las palabras del código, que también se indican en la tabla 6.5

$$\mathrm{P}_{\text{error no detectado}} = 1 - \mathrm{P}_{\text{error detectado}} = 7p^4(1-p)^3. \qquad \blacksquare$$

6.9. Códigos Hamming

Descubiertos entre 1947 y 1948 por R. W. Hamming y M. J. E. Golay, se trata de un conjunto de códigos lineales óptimos para corregir errores que afecten a un solo símbolo. Aunque nos interesan principalmente los códigos Hamming binarios, vamos a definirlos y a deducir sus propiedades suponiendo que se utiliza un alfabeto de codificación general. La construcción de códigos Hamming es muy sencilla.

Definición 6.11 (Código Hamming). *Un código Hamming es un código lineal 1-perfecto.*

Los parámetros $[n, k]$ de un código Hamming binario son, por tanto, las soluciones de la ecuación $n = 2^{n-k} - 1$. Existen, entonces, códigos Hamming binarios $[2^m - 1, 2^m - m - 1]$ para $m = 2, 3, \ldots$, y $\mathcal{H}_m$ será en adelante el de m símbolos de redundancia.

Ejemplo 6.9. De acuerdo con la definición anterior, y a la vista de su matriz de comprobación de paridad

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

el código descrito en los ejemplos 6.1, 6.4 y 6.6 es el código Hamming $\mathcal{H}_3$, de parámetros $[7, 4]$, sistemático. ■

Por definición, los códigos Hamming tienen distancia 3, y cualquier vector binario de longitud $2^m - 1$ o bien pertenece a $\mathcal{H}_m$ o bien dista una unidad de algún vector de $\mathcal{H}_m$. Además, según el teorema 6.10, para que estas condiciones se cumplan, dos columnas cualesquiera de H deben ser linealmente independientes (distintas, cuando el código es binario) y no nulas. En consecuencia, una matriz de comprobación de paridad de $\mathcal{H}_m$ tendrá por columnas todos los vectores binarios no nulos de longitud m. Conocida la estructura de H es fácil concluir que los códigos Hamming (los binarios $\mathcal{H}_m$ en particular) son códigos lineales correctores únicamente de errores simples que, salvo equivalencia, son únicos.

Además de los Hamming, se ha encontrado que el único código lineal binario perfecto no trivial es el Golay[23,12] (véase el ejemplo 7.4 en la página 211), y se ha demostrado que cualquier código binario perfecto no lineal tiene los mismos parámetros que un código Golay o que un código Hamming [41]. Los códigos perfectos son raros.

La distancia de un subcódigo nunca es inferior a la del código que lo contiene. Así, la de $\mathcal{H}_m$ se puede incrementar si se eliminan l columnas cualesquiera de H, resultando una matriz H', de dimensiones $m \times 2^m - l - 1$, comprobadora de paridad de un código lineal con parámetros $n = 2^m - l - 1$, $k = 2^m - m - l - 1$, $n - k = m$ y $d \geq 3$. Que la distancia del código acortado definido por H' es mayor o igual que 3 se deduce de ser H' una submatriz de H. Eligiendo adecuadamente qué columnas borrar, se pueden obtener códigos de distancia mínima igual a 4. Por ejemplo, si se prescinde de todas las columnas con un número par de unos, la suma de tres de las restantes no puede ser el vector nulo; es siempre un vector de peso impar, y existe por ello otra columna en H' idéntica a esa suma, lo cual implica que $d = 4$.

En lugar de acortar el código reduciendo símbolos de información y de redundancia, es posible a veces incrementar la distancia de un código lineal binario añadiendo un bit de paridad par, proceso que se conoce como *extensión* del código. Puesto que los códigos Hamming poseen distancia 3, es claro que la distancia del código Hamming extendido $[2^m, 2^m - m - 1]$, $\widehat{\mathcal{H}}_m$, será 4. El código dual de un Hamming extendido pertenece a la clase de los

códigos *Reed–Muller de primer orden*, cuya principal ventaja es que existe para ellos un algoritmo de decodificación muy simple [41].

Examinaremos aún en mayor profundidad la estructura de los códigos Hamming en el apartado 7.7.

6.10. Identidad de MacWilliams*

Hemos visto en el apartado 6.7 que la distribución de pesos de las palabras del código (el vector $(A_0, A_1, \ldots, A_n)$ con A_i el número de palabras del código de peso i) determina la probabilidad de detectar un error de transmisión.

Cuando un código lineal no posee ninguna propiedad definitoria especial o si, de existir alguna, no nos es conocida, entonces para calcular el vector $(A_0, \ldots, A_n)$ no se dispone de otro método que tabular todas las palabras del código con sus pesos. Obviamente, este es un procedimiento inviable para códigos de dimensión grande, por lo que, en general, la distribución de pesos de un código dado no es conocida a priori.

Sin embargo, ciertas clases de códigos exhiben propiedades estructurales más restrictivas que la simple linealidad (por ejemplo, los códigos Hamming o los códigos BCH), y, en ocasiones, esta información sobre la manera en que están construidos permite determinar su distribución de pesos sin necesidad de obtener todos los vectores del código.

Veremos a continuación que existe una relación analítica sorprendentemente sencilla entre la distribución de pesos de un código lineal y la distribución de pesos de su código dual. Esta relación, que además es una transformación lineal, es muy útil en la práctica cuando uno de los dos códigos es de un tamaño reducido, de manera que para él sí tenga sentido evaluar $(A_0, A_1, \ldots, A_n)$ por simple enumeración. De aquí se podrá entonces deducir la distribución de pesos del otro código, lo que normalmente sería imposible por el método directo. Aun en el caso de que la distribución de pesos se haya obtenido por otras vías, esta relación simplifica muchas veces el estudio de las propiedades del dual del código original.

Para hacer más sencillo el estudio de esa relación, se define el siguiente polinomio.[7]

[7] Aunque la deducción de la identidad de MacWilliams se efectúa, en este libro, a partir de los polinomios enumeradores homogéneos, el resultado final se puede obtener igualmente aplicando otras técnicas matemáticas [53].

Definición 6.12 (Polinomio enumerador). *Sea $\mathcal{C}[n,k]$ un código lineal con distribución de pesos $\{A_i\}$. Se llama polinomio enumerador de pesos de $\mathcal{C}$ a*

$$W_{\mathcal{C}}(x,y) = \sum_{i=0}^{n} A_i x^{n-i} y^i.$$

El polinomio enumerador es un polinomio homogéneo (de orden n) en dos variables, cuya utilidad es combinar todos los valores de los coeficientes A_i en una expresión compacta. Aunque reduciéndolo a una variable (fijando $x = 1$) no se pierde ninguna información esencial, existen buenas razones para conservar la notación en dos variables. Puesto que A_i es el número de palabras del código de peso Hamming i, $W_{\mathcal{C}}(x,y)$ se podrá escribir como

$$W_{\mathcal{C}}(x,y) = \sum_{\mathbf{x}\in\mathcal{C}} x^{n-p_H(\mathbf{x})} y^{p_H(\mathbf{x})} = \sum_{\mathbf{x}\in\mathcal{C}} P_{\mathbf{x}}(x,y)$$

si se conviene en definir $P_{\mathbf{x}}(x,y) = x^{n-p_H(\mathbf{x})} y^{p_H(\mathbf{x})}$.

Ahora, para cualquier $\mathbf{y} \in \mathcal{L}_n$, sea

$$g_n(\mathbf{y}) = \sum_{\mathbf{x}\in\mathcal{L}_n} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} P_{\mathbf{x}}(x,y) \tag{6.9}$$

una transformación[8] de los polinomios $P_{\mathbf{x}}(x,y)$, en donde $\langle \mathbf{x},\mathbf{y}\rangle$ representa el producto escalar de los vectores $\mathbf{x}$ e $\mathbf{y}$. Se cumple entonces que:

Lema 6.14. *Para cualquier código binario $\mathcal{C}[n,k]$*

$$\sum_{\mathbf{x}\in\mathcal{C}^{\perp}} P_{\mathbf{x}}(x,y) = \frac{1}{|\mathcal{C}|} \sum_{\mathbf{y}\in\mathcal{C}} g_n(\mathbf{y}) = \frac{1}{|\mathcal{C}|} \sum_{\mathbf{y}\in\mathcal{C}} \sum_{\mathbf{x}\in\mathcal{L}_n} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} P_{\mathbf{x}}(x,y).$$

[8] La ecuación (6.9) define la *transformada de Hadamard* de los polinomios $P_{\mathbf{x}}(x,y)$, $\mathbf{x} \in \mathcal{L}_n$. El vector $\mathbf{y}$ es un parámetro de tal transformación. En términos más generales, un homomorfismo

$$\begin{aligned} \chi : G &\longrightarrow \{z : \ |z| = 1\} \\ \chi(a+b) &\longrightarrow \chi(a)\chi(b) \end{aligned}$$

entre un grupo abeliano (aquí con notación aditiva) G y el conjunto de los números complejos de módulo 1 es un *carácter*; por consiguiente, la transformada de Hadamard

$$\begin{aligned} \chi_{\mathbf{x}} : \ G &\longrightarrow \{-1,1\} \\ \mathbf{y} &\longrightarrow \chi_{\mathbf{x}}(\mathbf{y}) = (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} \end{aligned}$$

es un carácter algebraico.

DEMOSTRACIÓN. Según (6.9),

$$\sum_{\mathbf{x}\in\mathcal{C}} g_n(\mathbf{x}) = \sum_{\mathbf{x}\in\mathcal{C}} \sum_{\mathbf{y}\in\mathcal{L}_n} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} P_{\mathbf{y}}(x,y) = \sum_{\mathbf{y}\in\mathcal{L}_n} P_{\mathbf{y}}(x,y) \sum_{\mathbf{x}\in\mathcal{C}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle}.$$

Si el vector $\mathbf{y} \in \mathcal{C}^\perp$, entonces $\langle \mathbf{x},\mathbf{y}\rangle = 0\ \forall \mathbf{x} \in \mathcal{C}$, y por tanto

$$\sum_{\mathbf{x}\in\mathcal{C}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} = \sum_{\mathbf{x}\in\mathcal{C}} (-1)^0 = |\mathcal{C}|.$$

Si el vector $\mathbf{y} \notin \mathcal{C}^\perp$, consideremos los conjuntos

$$\mathcal{C}_0(\mathbf{y}) = \{\mathbf{x} \in \mathcal{C} : \langle \mathbf{x},\mathbf{y}\rangle = 0\}$$
$$\mathcal{C}_1(\mathbf{y}) = \{\mathbf{x} \in \mathcal{C} : \langle \mathbf{x},\mathbf{y}\rangle = 1\}$$

de palabras del código ortogonales ($\mathcal{C}_0$) y no ortogonales ($\mathcal{C}_1$) a $\mathbf{y}$. Cuando el conjunto $\mathcal{C}_1(\mathbf{y})$ no es vacío, entonces contiene el mismo número de elementos que $\mathcal{C}_0(\mathbf{y})$; en efecto, si $\exists \mathbf{z} \in \mathcal{C}_1(\mathbf{y})$ entonces $\mathcal{C}_1(\mathbf{y}) = \{\mathbf{z} + \mathbf{u}, \mathbf{u} \in \mathcal{C}_0(\mathbf{y})\}$. Puesto que $\mathcal{C}_1(\mathbf{y}) \neq \emptyset$ si $\mathbf{y} \notin \mathcal{C}^\perp$,

$$\sum_{\mathbf{x}\in\mathcal{C}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} = \sum_{\mathbf{x}\in\mathcal{C}_0(\mathbf{y})} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} + \sum_{\mathbf{x}\in\mathcal{C}_1(\mathbf{y})} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} = 0.$$

Por lo tanto

$$\begin{aligned}\sum_{\mathbf{x}\in\mathcal{C}} g_n(\mathbf{x}) &= \sum_{\mathbf{x}\notin\mathcal{C}^\perp} P_{\mathbf{x}}(x,y) \sum_{\mathbf{y}\in\mathcal{C}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} + \sum_{\mathbf{x}\in\mathcal{C}^\perp} P_{\mathbf{x}}(x,y) \sum_{\mathbf{y}\in\mathcal{C}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} \\ &= |\mathcal{C}| \sum_{\mathbf{x}\in\mathcal{C}^\perp} P_{\mathbf{x}}(x,y).\end{aligned}$$

▶

Revisando todos los pasos de esta demostración, no es difícil ver que el lema se cumple, en general, para cualquier función $Q(\mathbf{x})$ definida sobre los vectores de $\mathcal{L}_n$, sin necesidad de que ésta coincida con el polinomio enumerador $P_{\mathbf{x}}(x,y)$.

Entonces, si fuera posible hallar una expresión explícita para el polinomio transformado $g_n(\mathbf{x})$, este lema permitiría deducir la relación que se busca. Tal expresión existe y se calcula en el lema siguiente.

LEMA 6.15. *Para todo* $\mathbf{x} \in \mathcal{L}_n$

$$g_n(\mathbf{x}) = (x+y)^{n-p_H(\mathbf{x})}(x-y)^{p_H(\mathbf{x})}.$$

Demostración. Se procederá por inducción en n. Para $n = 1$ tenemos que los únicos vectores de $\mathcal{L}_1 = \mathrm{GF}(2)$ son los elementos 0 y 1, y

$$\begin{aligned} g_n(\mathbf{x}) &= (-1)^{\langle \mathbf{x},0\rangle} P_0(x,y) + (-1)^{\langle \mathbf{x},1\rangle} P_1(x,y) \\ &= \begin{cases} (x+y), & \text{si } \mathbf{x} = 0 \\ (x-y), & \text{si } \mathbf{x} = 1 \end{cases} \\ &= (x+y)^{1-p_H(\mathbf{x})}(x-y)^{p_H(\mathbf{x})}. \end{aligned}$$

Supongamos que la identidad se verifica para $n = k$, y consideremos el caso $n = k+1$. Representemos por

$$\begin{aligned} \mathbf{x} &= (x_1, \ldots, x_{k+1}) \\ \mathbf{y} &= (y_1, \ldots, y_{k+1}) \end{aligned}$$

dos vectores de $k+1$ elementos, y por $\mathbf{u}$ y $\mathbf{v}$ dos vectores con las k primeras componentes de $\mathbf{x}$ e $\mathbf{y}$, respectivamente,

$$\begin{aligned} \mathbf{u} &= (x_1, \ldots, x_k) \\ \mathbf{v} &= (y_1, \ldots, y_k) \end{aligned}$$

Con esta notación, $g_{k+1}(\mathbf{x})$ se podrá escribir como

$$\begin{aligned} g_{k+1}(\mathbf{x}) &= \sum_{\substack{\mathbf{y}\in\mathcal{L}_{k+1} \\ y_{k+1}=0}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} P_{\mathbf{y}}(x,y) + \sum_{\substack{\mathbf{y}\in\mathcal{L}_{k+1} \\ y_{k+1}=1}} (-1)^{\langle \mathbf{x},\mathbf{y}\rangle} P_{\mathbf{y}}(x,y) \\ &= \sum_{\mathbf{v}\in\mathcal{L}_k} (-1)^{\langle \mathbf{u},\mathbf{v}\rangle} x^{k+1-p_H(\mathbf{v})} y^{p_H(\mathbf{v})} \\ &+ \sum_{\mathbf{v}\in\mathcal{L}_k} (-1)^{\langle \mathbf{u},\mathbf{v}\rangle + x_{k+1}} x^{k-p_H(\mathbf{v})} y^{p_H(\mathbf{v})+1} \\ &= x g_k(\mathbf{u}) + y(-1)^{x_{k+1}} g_k(\mathbf{u}) = g_k(\mathbf{u})(x + y(-1)^{x_{k+1}}). \end{aligned}$$

Aplicando ahora la hipótesis de inducción, obtenemos

$$g_{k+1}(\mathbf{x}) = (x+y)^{k-p_H(\mathbf{u})}(x-y)^{p_H(\mathbf{u})}(x + y(-1)^{x_{k+1}}).$$

Y considerando por separado los casos $x_{k+1} = 0$ y $x_{k+1} = 1$, se deduce finalmente

$$g_{k+1}(\mathbf{x}) = (x+y)^{k+1-p_H(\mathbf{x})}(x-y)^{p_H(\mathbf{x})}. \qquad \blacktriangleright$$

Teorema 6.16 (Identidad de MacWilliams). *Sea $\mathcal{C}[n,k]$ un código lineal binario, y sea $\mathcal{C}^\perp[n, n-k]$ su código dual. Se cumple entonces que*

$$W_{\mathcal{C}^\perp}(x,y) = \frac{1}{2^k} W_{\mathcal{C}}(x+y, x-y).$$

DEMOSTRACIÓN. Sea $(A_0, \ldots, A_n)$ la distribución de pesos de $\mathcal{C}[n,k]$.

$$\begin{aligned}\sum_{\mathbf{x}\in\mathcal{C}^\perp} P(\mathbf{x}) &= \frac{1}{|\mathcal{C}|}\sum_{\mathbf{x}\in\mathcal{C}} g_n(\mathbf{x}) \\ &= \frac{1}{|\mathcal{C}|}\sum_{\mathbf{x}\in\mathcal{C}} (x+y)^{n-p_H(\mathbf{x})}(x-y)^{p_H(\mathbf{x})} \\ &= \frac{1}{|\mathcal{C}|}\sum_{i=0}^{n} A_i(x+y)^{n-i}(x-y)^i = \frac{1}{|\mathcal{C}|}W_{\mathcal{C}}(x+y,x-y)\end{aligned}$$

donde la primera igualdad se sigue del lema 6.14 y la segunda del lema 6.15. ▶

La identidad de MacWilliams resulta más fácil de utilizar cuando el polinomio enumerador se reduce a una sola variable fijando $x = 1$. Pues así $W_{\mathcal{C}}(x,y)$ se puede expresar como

$$\begin{aligned}B(x) = W_{\mathcal{C}^\perp}(1,x) &= \sum_{i=0}^{n} B_i x^i = (\text{teorema (6.16)})\ 2^{-k}W_{\mathcal{C}}(1+x,1-x) \\ &= 2^{-k}\sum_{i=0}^{n} A_i(1+x)^{n-i}(1-x)^i = 2^{-k}(1+x)^n \sum_{i=0}^{n} A_i\left(\frac{1-x}{1+x}\right)^i \\ &= 2^{-k}(1+x)^n A\left(\frac{1-x}{1+x}\right)\end{aligned}$$

siendo $A(x) = W_{\mathcal{C}}(1,x) = \sum_{i=0}^{n} A_i x^i$ el polinomio enumerador en una variable de $\mathcal{C}$. Si se intercambian en las expresiones anteriores los papeles de $\mathcal{C}$ y $\mathcal{C}^\perp$, y se repite el desarrollo, la fórmula

$$A(x) = 2^{-(n-k)}(1+x)^n B\left(\frac{1-x}{1+x}\right)$$

indica cómo calcular la distribución de pesos de $\mathcal{C}$ una vez conocida la del código dual. Desarrollando término a término ambos miembros de estas dos igualdades se llega a dos sistemas de ecuaciones lineales:

$$\sum_{j=0}^{n}\binom{j}{\nu}A_j = 2^{k-\nu}\sum_{j=0}^{\nu}(-1)^j\binom{n-j}{n-\nu}B_j, \quad 0\le\nu\le n$$

$$\sum_{j=0}^{n-\nu}\binom{n-j}{\nu}A_j = 2^{k-\nu}\sum_{j=0}^{\nu}\binom{n-j}{n-\nu}B_j, \quad 0\le\nu\le n$$

que las distribuciones de pesos de un código y su dual deben satisfacer.

Cualquiera de estos sistemas se puede resolver para obtener explícitamente los coeficientes B_i en función de los A_i, o viceversa. Para poder expresar la solución de manera compacta, definamos antes los *polinomios de Krawtchouk*

$$K_{k,n}(x) = \sum_{j=0}^{k} (-1)^j \binom{x}{j} \binom{n-x}{k-j}, \quad \text{para } 0 \leq k \leq n.$$

$K_{k,n}(x)$ es un polinomio de grado k en la variable x, comenzando la sucesión con

$$\begin{aligned} K_{0,n}(x) &= 1 \\ K_{1,n}(x) &= n - 2x \\ K_{2,n}(x) &= 2x^2 - 2nx + \binom{n}{2} \\ K_{3,n}(x) &= -\frac{4}{3}x^3 + 2nx^2 - \left(n^2 + n + \frac{2}{3}\right) + \binom{n}{3} \end{aligned}$$

Los polinomios satisfacen numerosas propiedades, entre ellas algunas interesantes de simetría:

$$K_{k,n}(x) = (-1)^k K_{k,n}(n-x)$$

y de ortogonalidad:

$$\sum_{i=0}^{n} \binom{n}{i} K_{k,n}(i) K_{l,n}(i) = \delta_{kl} \binom{n}{k} 2^n$$

en donde δ_{kl} es el símbolo de Kronecker. Pues bien, dados $A_0, A_1, \ldots, A_n$, la distribución de pesos de un cierto código lineal, el número de palabras del código de peso j en su código dual es

$$B_j = \frac{1}{|\mathcal{C}|} \sum_{i=0}^{n} A_i K_{j,n}(i) \qquad 0 \leq j \leq n. \tag{6.10}$$

Ejemplo 6.10. El conjunto de palabras del código Hamming [7,4,3] aparece en la tabla 6.6. Hay una palabra del código de peso 0, 7 de peso 3, otras 7 de peso 4 y una de peso 7. El polinomio enumerador del código Hamming es entonces

$$W_{\mathcal{H}_3}(x,y) = x^7 + 7x^4y^3 + 7x^3y^4 + y^7$$

que exhibe la peculiaridad de ser simétrico, $W_{\mathcal{H}_3}(x,y) = W_{\mathcal{H}_3}(y,x)$, reflejando la simetría de la distribución de pesos. La identidad de MacWilliams permite afirmar que el código dual tiene un polinomio enumerador

$$W_{\mathcal{H}_3^\perp}(x,y) = x^7 + 7x^3y^4$$

Palabra del código	Peso	Palabra del código	Peso
0000000	0	1000110	3
0001101	3	1001011	4
0010111	4	1010001	3
0011010	3	1011100	4
0100011	3	1100101	4
0101110	4	1101000	3
0110100	3	1110010	4
0111001	4	1111111	7

Tabla 6.6. El código Hamming $\mathcal{H}_3$.

$K_{j,7}(i)$	j							
	1	7	21	35	35	21	7	1
	1	5	9	5	−5	−9	−5	−1
	1	3	1	−5	−5	1	3	1
i	1	1	−3	−3	3	3	−1	−1
	1	−1	−3	3	3	−3	−1	1
	1	−3	1	5	−5	−1	3	−1
	1	−5	9	−5	−5	9	−5	1
	1	−7	21	−35	35	−21	7	−1

Tabla 6.7. Valores de los polinomios de Krawtchouk.

de donde se deduce que este código dual consta de una palabra de peso 0 y 7 de peso 4. La tabla 6.7 lista los coeficientes $K_{j,7}(i)$ con los que verificar la expresión (6.10). El código dual de un Hamming se estudia con más detalle en el apartado 7.7.

Si se extiende $\mathcal{H}_3$, se obtiene un código $[8,4,4]$ autodual, $\widehat{\mathcal{H}}_3$, compuesto por el vector nulo, 14 vectores de peso 4 y un vector de peso 8. Por tanto,

$$W_{\widehat{\mathcal{H}}_3}(x,y) = x^8 + 14x^4y^4 + y^8.$$

Puede comprobarse sin dificultad que $W_{\widehat{\mathcal{H}}_3}(x,y) = W_{\widehat{\mathcal{H}}_3^{\perp}}(x,y)$ y, aun más, que $W_{\widehat{\mathcal{H}}_3}(x,y)$ es el único polinomio enumerador posible de un código $[8,4,4]$ autodual. ■

Ejemplo 6.11. Más en general, la distribución de pesos del dual de un código Hamming binario, $\mathcal{H}_m^{\perp}$, se conoce y está dada por $B_0 = 1$, $B_{2^{m-1}} = 2^m - 1$ y $B_i = 0$ para cualquier otro índice i. Haciendo uso de las ecuaciones (6.10), se puede determinar la distribución de pesos de $\mathcal{H}_m$:

$$A_j = 2^{-m}\left(K_{j,n}(0) + (2^m - 1)K_{j,n}(2^{k-1})\right), \quad 0 \le j \le n$$

para $n = 2^m - 1$. ■

Notas bibliográficas

Muchas de las construcciones matemáticas que luego se propondrían como códigos de control de errores se conocían desde tiempo antes en otros contextos. Por ejemplo, C. E. Shannon ya menciona el trabajo de R. W. Hamming, y los códigos epónimos, en [59]; Fisher descubrió los códigos símplex binarios en 1942, mientras estudiaba el problema de cómo diseñar experimentos estadísticos factoriales; los códigos Reed–Muller eran conocidos en 1952 como matrices ortogonales; y los códigos Golay aparecieron como objetos geométricos con anterioridad al descubrimiento de sus propiedades algebraicas.

La teoría general de los códigos lineales, que unificaba en cierto modo entes matemáticos tan dispares, fue tomando cuerpo durante el decenio de los cincuenta, con contribuciones singulares de D. E. Slepian [62] y R. W. Hamming. El artículo de Slepian [63] resume bien todo este desarrollo. Berlekamp [4] y el mismo Slepian [64] reunieron el estado del arte de la disciplina a mediados de los setenta. Los tratados de MacWilliams y Sloane [41], de finales de los setenta, y de Pless y Huffman [53] son obras enciclopédicas de obligada referencia, tanto por la extensión y rigor con que se presentan los temas como por la exhaustividad de las citas bibliográficas que los acompañan. De hecho, el capítulo primero de [53] es una completa exposición de la teoría de códigos lineales; [53, cap. 3] trata enteramente los códigos autoduales y [53, cap. 8] discute una serie de resultados acerca del radio de cobertura. La identidad de MacWilliams admite varias versiones no lineales que se describen ampliamente en [41, cap. 5], así como una generalización que da lugar a los llamados momentos de Pless [52, 54]. Las cotas de distancia para códigos de bloques tienen una larga historia, y se tratan en profundidad tanto en [41, cap. 17] como en [69, cap. 5].

Otras familias de códigos no incluidas en el presente texto son las de los códigos QR, Reed–Muller, Goppa y de geometría algebraica, que sí se cubren en [41] y [53]. Los códigos no lineales de Hadamard, Kerdock, Preparata, Delsarte y Delsarte–Goethals también se discuten en estas dos obras.

Si al lector le interesa conocer las aplicaciones de los códigos de control de errores, el artículo de Costello *et al.* [14] contiene una relación extensa de casos prácticos. Calderbank [11] traza un recorrido histórico, siguiendo el desarrollo de la teoría de la codificación desde los orígenes hasta nuestros días. Ambos trabajos forman parte del volumen conmemorativo [70].

6.A. Cotas de distancia

Llegados a este punto, debe resultar claro que la posibilidad de construir un buen código lineal parte de la respuesta al siguiente problema inverso: dados n, $k < n$ y $d > 0$, ¿existe algún código lineal $[n, k]$ de distancia mínima (al menos) d? O bien, conocidos $d > 0$ y $R_c < 1$, ¿es posible construir algún código lineal de distancia d con tasa R_c? Ambas son versiones simplificadas del problema básico general de la existencia de códigos de bloques, que se puede formular de la manera siguiente: dados un alfabeto finito $\mathcal{F}$ y un número natural $d > 0$, ¿cuál es el número máximo de secuencias del código en $\mathcal{F}^n$ con distancia Hamming al menos d entre dos cualesquiera de ellas? Generalizar el problema en esta forma es conveniente y necesario, puesto que, en ocasiones, para poder encontrar el máximo número de palabras del código a una determinada distancia se deben construir códigos no lineales. Así, por ejemplo, el único código binario de longitud 16, distancia 6, con 256 palabras es no lineal (el código Nordstrom–Robinson).

No obstante la dificultad de responder genéricamente a la cuestión, sí es posible llegar a deducir límites superiores e inferiores a la distancia d de un código haciendo solamente consideraciones de índole geométrica o combinatoria sobre el espacio de vectores binarios. En este apartado se discuten una serie de teoremas que determinan el valor de algunas de estas cotas.

Comenzaremos por presentar varias cotas superiores, a cual más precisa, para la distancia máxima que puede alcanzar un código de bloques de longitud n y M palabras, código que se representará como (n, M).

Teorema 6.17 (Singleton). *Para el número M de palabras de un código binario de longitud n y distancia d se cumple siempre que*

$$M \leq 2^{n-d+1}.$$

En particular, si $\mathcal{C}$ es un código lineal $\mathcal{C}[n, k]$, entonces su distancia satisface la inecuación

$$d \leq n - k + 1.$$

Demostración. Suponga que existen M vectores binarios de longitud n con distancia d o mayor entre dos cualesquiera de ellos. Si se borran $d - 1$ símbolos de todas las secuencias del código, se tiene un código de bloques de longitud $n - d + 1$, distancia al menos 1 y M palabras, es decir, las M palabras del código son distintas entre sí. Por lo tanto, $M \leq 2^{n-d+1}$.

Si el código es lineal, $M = 2^k$. También basta con observar que la distancia es una unidad mayor que r, el número máximo tal que todo grupo de r columnas de la matriz comprobadora de paridad son linealmente independientes; y este número no es mayor que el rango de H, que es $n - k$. ▶

Un código $[n, k]$ con distancia $d = n - k + 1$ se llama *código separable de máxima distancia*. Excepto los códigos $[n, n-1]$ de paridad par y los duales de éstos, los códigos $[n, 1]$ de repetición (que son aquéllos cuyas palabras del código se construyen repitiendo un número n —generalmente impar, ¿por qué?— de veces los símbolos del mensaje), no hay códigos binarios separables de máxima distancia; en cambio, sí existen códigos no binarios de distancia óptima $d = n - k + 1$, como por ejemplo los de Reed–Solomon. En la práctica, en la mayoría de los códigos, el límite superior de Singleton suele estar bastante alejado de su distancia real.

Por otra parte, según establece el teorema 6.9, si un código ha de corregir t errores, su distancia mínima tendrá que cumplir $d \geq 2t+1$. Pero, aplicando la cota de Singleton

$$2t + 1 \leq d \leq n - k + 1 \Rightarrow n - k \geq 2t,$$

se concluye inmediatamente que el número de bits de redundancia deberá ser al menos el doble que la capacidad de corrección.

A menudo interesa determinar si toda una colección de códigos con idéntica estructura interna es eficiente. Puesto que las propiedades de control de los errores dependen directamente de la distancia, y como, según el teorema de Shannon, los buenos códigos tienen longitud grande, una manera intuitiva de medir la bondad de una familia infinita de códigos es estudiar la convergencia, si existe, de la distancia relativa $\frac{d}{n}$ cuando $n \to \infty$. En otras palabras, para códigos de una misma clase, se trata de determinar si, a medida que su longitud aumenta, lo hace también, y en qué proporción, su capacidad de detección y corrección de errores. Pero si la noción de convergencia ha de tener algún sentido, se han de comparar códigos con la misma cantidad de redundancia por cada símbolo del mensaje, es decir con el mismo valor de R_c. La siguiente definición expone de manera más precisa estos conceptos.

Definición 6.13 (Códigos no degenerados). *Una familia de códigos $\mathcal{C}$ se denomina no degenerada si contiene una secuencia infinita de códigos $\mathcal{C}_1(n_1, M_1)$, $\mathcal{C}_2(n_2, M_2), \ldots$, tal que existen*

$$\lim_{i\to\infty} \frac{\log M_i}{n_i} \quad y \quad \lim_{i\to\infty} \frac{d_{\mathcal{C}_i}}{n_i}$$

y son distintos de cero.

Ejemplo 6.12. Para la clase de códigos Hamming se comprueba con facilidad que

$$\lim_{n\to\infty} \frac{k}{n} = 1 \qquad \lim_{n\to\infty} \frac{d_{\mathcal{C}}}{n} = 0$$

y, por ello, que no constituyen una familia no degenerada. ■

El supuesto más habitual en la práctica es examinar familias de códigos $\mathcal{C} = \{\mathcal{C}_i(n_i, M_i), i \geq 1\}$ para las cuales $(\log M_i)/n_i = R_c$, $\forall i \geq 1$. Y aunque el cálculo explícito de $\lim_{i\to\infty} d_{\mathcal{C}_i}/n_i$ no es posible en la mayor parte de los casos, podemos emplear las cotas de distancia que iremos presentando para calcular también cotas límite de distancia que nos ayudarán a comprender las propiedades de los códigos de longitud grande.

De acuerdo con esto, si se dividen ambos miembros de la desigualdad de Singleton entre n, y se toma el límite cuando $n \to \infty$, resulta la siguiente fórmula asintótica.

Teorema 6.18. *En cualquier código de bloques* $(n, 2^k)$

$$R \lesssim 1 - \frac{d}{n}.$$

(La notación $f(n) \lesssim g(n)$ significa que $f(n) \leq g(n)(1+o(1))$, con $o(1) \to 0$, cuando $n \to \infty$). Así que, en el caso más favorable, la convergencia de las distancias de una familia no degenerada de códigos es lineal en n y no supera el valor $1 - R$, la proporción de redundancia del mensaje codificado.

El siguiente resultado elemental establece un límite superior al número de errores que se pueden corregir con un código de bloques binario; o, dicho de otro modo, da un límite inferior a la redundancia que se necesita para que exista un código con distancia $2t + 1$.

Teorema 6.19 (Hamming). *Si* $\mathcal{C}(n, 2^k)$ *es un código binario corrector de* t *errores, entonces se verifica que*

$$n - k \geq \log_2 \sum_{i=0}^{t} \binom{n}{i}.$$

Demostración. Las 2^k esferas con centro en una palabra del código y radio t son disjuntas entre sí y contienen, cada una, $\sum_{i=0}^{t} \binom{n}{i}$ vectores. Pero el número total de vectores del espacio es 2^n. Por tanto, como el número de vectores contenidos en alguna esfera con centro en una palabra del código y radio t evidentemente no puede ser mayor que el número de vectores binarios de longitud n, se podrá escribir

$$2^n \geq 2^k \sum_{i=0}^{t} \binom{n}{i}.$$

Tomando logaritmos en ambos miembros se tiene la desigualdad del enunciado.

Una vez más, en la demostración no ha sido necesario apelar a la condición de linealidad, por lo que la desigualdad de Hamming es válida para cualquier código de bloques. ▶

Sólo los códigos perfectos, por definición, satisfacen la cota de Hamming con igualdad, de manera que, cuando son lineales, cada columna $j = 1, \ldots, 2^k$ de su matriz típica está formada por todos los vectores pertenecientes a la esfera con centro en la palabra del código C_j que encabeza la columna y radio t.

Para la deducción de la cota asintótica de Hamming necesitaremos el siguiente lema, que da una estimación precisa de la suma de coeficientes binomiales.

LEMA 6.20.

$$2^{nH_2(p)}O(n^{-1/2}) \leq \sum_{i=0}^{np} \binom{n}{i} \leq 2^{nH_2(p)} \quad \forall\, 0 \leq p < \frac{1}{2}.$$

Por tanto

$$\lim_{n\to\infty} \frac{1}{n} \log_2 \sum_{i=0}^{np} \binom{n}{i} = H_2(p) \quad \forall\, 0 \leq p < \frac{1}{2}.$$

DEMOSTRACIÓN. Probemos en primer lugar la cota superior. Primero,

$$\sum_{i=0}^{np} \binom{n}{i} = \sum_{i=n\lambda}^{n} \binom{n}{i}$$

con $\lambda = 1 - p$. Para cualquier número positivo r

$$2^{rn\lambda} \sum_{i=n\lambda}^{n} \binom{n}{i} \leq \sum_{i=n\lambda}^{n} 2^{ri} \binom{n}{i} \leq \sum_{i=0}^{n} \binom{n}{i} 2^{ri} = (1 + 2^r)^n$$

y por tanto

$$\sum_{i=n\lambda}^{n} \binom{n}{i} \leq \left(2^{-r\lambda} + 2^{r\lambda}\right)^n.$$

Eligiendo $r = \log_2(\lambda/(1-\lambda))$ y operando se tiene que

$$\sum_{i=n\lambda}^{n} \binom{n}{i} \leq 2^{nH_2(\lambda)} \left(1 - \lambda + \lambda\right)^n = 2^{nH_2(\lambda)} = 2^{nH_2(p)}.$$

La cota inferior se sigue de la desigualdad

$$\sum_{i=np}^{n} \binom{n}{i} \geq \binom{n}{np}.$$

Aplicando la fórmula de Stirling

$$\sqrt{2\pi n} n^n e^{-n} < n! < \sqrt{2\pi n} n^n e^{-n+\frac{1}{12n}}$$

tenemos

$$\begin{aligned}\binom{n}{np} &= \frac{n!}{(np)!(n(1-p))!} \\ &\geq \frac{1}{\sqrt{2\pi np(1-p)}} \frac{1}{p^{np}(1-p)^{n(1-p)}} e^{-\frac{1}{12np}-\frac{1}{12n(1-p)}} \\ &= \frac{1}{\sqrt{2\pi np(1-p)}} 2^{nH_2(p)} e^{-1/(12np(1-p))} = 2^{nH_2(p)} O(n^{-1/2}).\end{aligned}$$

Tomando el límite $n \to \infty$ de las cotas superior e inferior

$$\lim_{n\to\infty} \frac{1}{n} \log_2 \sum_{i=np}^{m} \binom{n}{i} = H_2(p). \qquad \blacktriangleright$$

Tomando límites en ambos miembros de la desigualdad de Hamming y aplicando este lema, obtenemos directamente el siguiente resultado.

Teorema 6.21 (Cota asintótica de Hamming).

$$R \lesssim 1 - H_2\left(\frac{d}{2n}\right).$$

Teorema 6.22 (Plotkin). *La distancia mínima de un código binario de longitud n y M palabras cumple*

$$d \leq \frac{1}{2} \frac{nM}{M-1}.$$

En códigos lineales

$$d \leq \frac{n2^{k-1}}{2^k - 1}.$$

Demostración. Calculemos

$$\sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}} d(\mathbf{x},\mathbf{y}),$$

la suma de las distancias entre todos los posibles pares ordenados de palabras del código, de dos formas distintas.

Escribiendo las palabras del código $\mathcal{C}$ como las filas de una matriz $M \times n$, y si c_i, $1 \leq i \leq n$, es el número de unos en la columna i de esa matriz, se tiene

$$\sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}} d(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} 2c_i(M - c_i) \leq n\frac{M^2}{2}.$$

Por otra parte, hay $M(M-1)$ pares ordenados diferentes de palabras del código con distancia mínima d entre ellas. Luego

$$\sum_{\mathbf{x},\mathbf{y}\in\mathcal{C}} d(\mathbf{x},\mathbf{y}) \geq M(M-1)d.$$

Por consiguiente,

$$d \leq \frac{1}{2}\frac{nM}{M-1}.$$

En los códigos lineales basta fijar $M = 2^k$. Observe que en la inecuación de Plotkin se pueden suponer conocidos dos parámetros cualesquiera y tomar el otro como incógnita. Así,

$$M \leq \frac{2d}{2d-n}$$

da un límite al número máximo de secuencias de un código de bloques de longitud n y distancia d, si $n < 2d$. ▶

La cota de Plotkin viene a decir simplemente que la distancia de un código lineal binario no es mayor que el número medio de símbolos no nulos en una palabra del código.

Una familia de códigos lineales que satisface con igualdad la inecuación de Plotkin es la familia de códigos símplex $[2^m - 1, m]$ con $m \geq 2$, cuyas palabras no nulas tienen todas peso 2^{m-1} y son ortogonales entre sí. De hecho, se aprecia de inmediato en la demostración que, de alcanzar la igualdad estricta, un código debe constar sólo de secuencias de igual peso Hamming. En el apartado 7.7 se analizarán las propiedades de los códigos símplex con más detenimiento. Asimismo, en virtud del denominado teorema de Levenshtein [41] (que es una versión más general del teorema de Plotkin), se sabe que también para algunos códigos no lineales, como los códigos Hadamard, la desigualdad se convierte en una igualdad.

Si en la inecuación de Plotkin consideramos que $R_c = \frac{k}{n}$ se mantiene constante cuando $n \to \infty$, entonces

$$\lim_{n\to\infty} \frac{d}{n} \leq \lim_{k\to\infty} \frac{2^{k-1}}{2^k - 1} = \frac{1}{2}$$

por lo que, en general, podemos suponer que para cualquier familia de códigos no degenerada, $\frac{d}{n} < \frac{1}{2}$.

El resultado siguiente muestra cómo es posible mejorar la cota de Singleton cuando se consideran códigos lineales.

Teorema 6.23 (Griesmer). *Sea $N(k,d)$ la longitud mínima de cualquier código lineal binario de dimensión k y distancia d. Entonces*

$$N(k,d) \geq \sum_{i=0}^{k-1} \left\lceil \frac{d}{2^i} \right\rceil .$$

Demostración. Sea $\mathcal{C}$ un código lineal binario $(N(k,d),k)$ con distancia d. Su matriz generadora se podrá escribir en la forma

$$G = \begin{pmatrix} 0 & 0 & \dots & 0 & 1 & 1 & \dots & 1 \\ & & G_1 & & & & G_2 & \end{pmatrix}$$

con G_1 una matriz $(k-1)\times(N(k,d)-d)$ de rango $k-1$: si el rango de G_1 no fuese $k-1$, se podría hacer nula su primera fila y G no generaría un código con distancia d. Por tanto, G_1 genera un código lineal $(N(k,d)-d, k-1)$ con distancia d'.

Pero si $(u,v) \in \mathcal{C}$ (u un vector binario de longitud $N(k,d)-d$ y peso d'), como $(u, v+1) \in \mathcal{C}$, resulta

$$\begin{aligned} d' + p_H(v) &\geq d \\ d' + d - p_H(v) &\geq d \end{aligned}$$

y, sumando ambas miembro a miembro, $2d' \geq d$ o bien $d' \geq \lceil d/2 \rceil$.

Ocurre, entonces, que $N(k,d) \geq d + N(k-1, \lceil d/2 \rceil)$, y por inducción, que

$$N(k,d) \geq \sum_{i=0}^{k-1} \left\lceil \frac{d}{2^i} \right\rceil . \qquad \blacktriangleright$$

Una familia de códigos que alcanza la cota de Griesmer es la familia de los códigos duales de los códigos Hamming.

El próximo resultado generaliza el argumento utilizado en la deducción de la cota de Plotkin, aplicándolo a una región adecuadamente elegida de $\mathrm{GF}(2)^n$.

Teorema 6.24 (Elias). *Sean $\mathcal{C}(n,M,d)$ un código de bloques binario, $\mathbf{x}$ un vector binario de longitud n, y K el número medio de palabras del código contenidas en cualquier bola B de centro $\mathbf{x} \in \mathcal{L}_n$ y radio $t > 0$, $B = \{\mathbf{y} \in$*

$\mathcal{L}_n : d_H(\mathbf{x}, \mathbf{y}) \leq t\}$. *Entonces, la distancia mínima del código satisface la desigualdad*

$$d \leq 2t \left(1 - \frac{t}{n}\right) \frac{K}{K-1}.$$

DEMOSTRACIÓN. La prueba de esta cota superior es un tanto laboriosa, así que se expondrá por partes:

1. Fijemos $t > 0$. Para cualquier secuencia binaria $\mathbf{x}_i$ de longitud n, sea K_i el número de palabras de $\mathcal{C}$ contenidas en la bola de centro $\mathbf{x}_i$ y radio t. La palabra del código $\mathbf{c}_j$ pertenece a $\sum_{j=0}^{t} \binom{n}{j}$ bolas de radio menor o igual que t, todas aquéllas cuyo centro es justamente un vector situado a distancia Hamming menor o igual que t de $\mathbf{c}_j$. Por tanto, $\sum_{i=1}^{2^n} K_i = M \sum_{j=0}^{t} \binom{n}{j}$.

2. Si se define $K = 2^{-n} \sum_{i=1}^{2^n} K_i$, entonces $M \sum_{j=0}^{t} \binom{n}{j} = 2^n K$. Luego existe al menos una bola B_K de radio t y centro $\mathbf{x}$ que contiene $K = M 2^{-n} \sum_{j=0}^{t} \binom{n}{j}$ o más palabras del código, siendo K el número medio de palabras del código en una bola de radio t.

3. Supongamos ahora que A es una matriz cuyas filas son los vectores $\mathbf{c}_j - \mathbf{x}$ para todo $\mathbf{c}_j \in B_K \cap \mathcal{C}, j = 1, \ldots, K$, que α_i es el número de unos en la columna $i = 1, \ldots, n$ de A y que β_i es el número de ceros en la columna i. La suma del peso Hamming de los $K \geq 1$ vectores fila de A puede escribirse también como $\sum_{i=1}^{n} \alpha_i = Kn - \Delta$, en donde $\Delta \geq K(n-t)$ es el número total de ceros en A ya que $p_H(\mathbf{c}_j - \mathbf{x}) \leq t$. Sea
$$\overline{\beta} = \frac{\sum_{j=1}^{n} \beta_j}{n}$$
la media aritmética de la secuencia β_j. Como
$$\sum_{j=1}^{n} (\beta_j - \overline{\beta})^2 = \sum_{j=1}^{n} \beta_j^2 - n\overline{\beta}^2 \geq 0$$
por ser positivos todos los sumandos de la primera suma, resulta
$$\sum_{j=1}^{n} \beta_j^2 \geq n\overline{\beta}^2 \geq n^{-1}(Kn - \Delta)^2, \qquad \alpha_j^2 = (K - \beta_j)^2.$$

4. Fijemos una columna j y una fila i de A. Si el elemento $a_{i,j}$ vale 1, entonces a la hora de sumar las distancias de todos los posibles pares ordenados de las filas de A hemos de contabilizar $K - \alpha_j$ unidades debidas a los ceros de la columna j. Del mismo modo, si $a_{i,j}$ vale 0, se

han de contabilizar $K-\beta_j$ unidades contribuidas por los unos de la columna j. Por lo tanto, la columna j aporta $\alpha_j(K-\alpha_j)+\beta_j(K-\beta_j)$ a la suma total, que valdrá $\sum_{j=1}^{n}\alpha_j(K-\alpha_j)+\beta_j(K-\beta_j)$. Pero la suma de las distancias entre todos los posibles pares *ordenados* de A es la suma d_T de las $K(K-1)$ distancias entre las filas de A. Así que

$$\begin{aligned} d_T &= \sum_{j=1}^{n}\alpha_j(K-\alpha_j)+\beta_j(K-\beta_j) \\ &= \sum_{j=1}^{n}(\alpha_j+\beta_j)K - \sum_{j=1}^{n}\left(\alpha_j^2+\beta_j^2\right) \\ &\leq nK^2 - \sum_{j=1}^{n}(K^2-2K\Delta+2\beta_j^2) \\ &\leq nK^2 - (nK^2-2K\Delta+2n^{-1}\Delta^2). \end{aligned}$$

Utilizando que $\Delta \geq K(n-t)$, y tomando $t \leq n/2$, se tiene que $\Delta \geq K(n-t) \geq nK/2$. Introduciendo esta desigualdad, la suma de distancias satisface

$$d_T \leq K^2 t\left(2-\frac{2t}{n}\right).$$

5. Y si $d_{\min}$ es la distancia mínima entre las filas de A, es también la distancia mínima entre las palabras del código que pertenecen a B_K, de manera que $K(K-1)d_{\min} \leq d_T$. La distancia mínima de $\mathcal{C}$ es $d \leq d_{\min}$. Finalmente,

$$d \leq d_{\min} \leq 2t\left(1-\frac{t}{n}\right)\frac{K}{K-1}$$

lo que concluye la demostración del enunciado. ▶

Una extensión de este argumento lleva a la siguiente cota superior para el tamaño de un código de longitud n y distancia d, conocida como cota de Elias.

Teorema 6.25 (Elias). *Sea $\mathcal{C}$ un código de bloques binario (n, M, d) y t un número natural positivo tal que $2t^2-2nt+nd>0$. Se cumple que*

$$M \leq \frac{dn}{2t^2-2nt+nd}\,\frac{2^n}{\sum_{i=0}^{t}\binom{n}{i}}.$$

Demostración. Fijemos cualquier $t>0$. De acuerdo con la demostración del teorema 6.24, $\mathcal{C}$ contiene un subcódigo con $K \geq M\sum_{i=0}^{t}\binom{n}{i}/2^n$

palabras. Por lo tanto,

$$M\frac{\sum_{i=0}^{t}\binom{n}{i}}{2^n} \leq K \leq \frac{dn}{2t^2-2nt+nd}$$

si el denominador es una cantidad positiva. ▶

Hasta el descubrimiento de la cota de McEliece–Rodemich–Rumsey–Welch [45], la cota asintótica de Elias era la mejor cota superior conocida para los parámetros de un código. La formulación precisa de esta última es la siguiente.

TEOREMA 6.26 (COTA ASINTÓTICA DE ELIAS). *Existen códigos de bloques* (n, M, d) *tales que*

$$R \lesssim 1 - H_2\left(\frac{1}{2} - \frac{1}{2}\sqrt{1-\frac{2d}{n}}\right)$$

siendo $R = \log_2 M/n$.

DEMOSTRACIÓN. Fijemos

$$\gamma < \frac{1}{2}\left(1-\sqrt{1-\frac{2d}{n}}\right)$$

y tomemos $t = \lfloor \gamma n \rfloor$, de tal modo que $2t^2 - 2nt + nd > 0$. Por el teorema anterior

$$\frac{\log_2 M}{n} \leq \frac{1}{n}\left(\log_2 \frac{nd}{2t^2-2nt+nd} + \log_2 \frac{2^n}{\sum_{i=0}^{t}\binom{n}{i}}\right).$$

Tomando en ambos miembros el límite cuando $n \to \infty$, tenemos

$$R \leq 1 - H_2(\gamma),$$

ya que

$$\log_2 \frac{nd}{2t^2-2nt+nd} \to \log_2 \frac{\lambda}{2\gamma^2-2\gamma+\lambda} \quad \text{cuando } n \to \infty$$

para cierto $\lambda < 1/2$. Dado que $R \leq 1 - H_2(\gamma)$ para cualquier $\gamma < \frac{1}{2}(1 - \sqrt{1-\frac{2d}{n}})$, se cumple que

$$R \lesssim 1 - H_2\left(\frac{1}{2}\left(1-\sqrt{1-\frac{2d}{n}}\right)\right). \qquad ▶$$

Disponer de todo este conjunto de cotas superiores no resuelve el problema básico que se enunció al comienzo de este apartado: ¿existen códigos con distancia mínima mayor que una dada? He aquí una condición suficiente para poder afirmarlo.

Teorema 6.27 (Gilbert–Varshamov). *Existe un código lineal* $[n, k]$ *binario de distancia al menos* d *si*

$$\sum_{i=0}^{d-2} \binom{n-1}{i} < 2^{n-k}.$$

Demostración. Probaremos la desigualdad mostrando que, si se satisface la condición del enunciado, entonces existe una matriz de comprobación de paridad de dimensiones $(n-k) \times n$ con la propiedad de que ningún subconjunto de $d-1$ o menos columnas suyas es linealmente dependiente. Por el teorema 6.10, esto bastará para establecer que la distancia del código que define es por lo menos d.

Supongamos que para algún $j > n-k$ se tiene una matriz $(n-k) \times j$ tal que no hay $d-1$ columnas suyas linealmente dependientes. El número total de combinaciones lineales de $d-2$ o menos vectores que es posible formar con las j columnas es de

$$\binom{j}{1} + \binom{j}{2} + \cdots + \binom{j}{d_{\mathcal{C}}-2}.$$

Si este número es menor que $2^{n-k}-1$, entonces se puede elegir un vector no nulo distinto de cualquiera de estas combinaciones lineales, y añadirlo a las columnas de la matriz manteniendo la propiedad de que ningún subconjunto con menos de d columnas de la nueva matriz $(n-k) \times (j+1)$ sea linealmente dependiente. Y esto se podrá seguir haciendo mientras que

$$\sum_{i=1}^{d-2} \binom{n-1}{i} < 2^{n-k} - 1$$

o bien, como $\binom{n-1}{0} = 1$, mientras que

$$\sum_{i=0}^{d-2} \binom{n-1}{i} < 2^{n-k}$$

que es el resultado que se pretendía demostrar.

La matriz que se acaba de construir es la de un código de longitud n, con dimensión *al menos* k y distancia *al menos* d. ▶

El de Gilbert–Varshamov es un teorema de existencia que proporciona una cota *inferior* a la distancia de un buen código lineal. En la proposición siguiente se indica cómo obtenerla.

TEOREMA 6.28 (COTA ASINTÓTICA DE GILBERT–VARSHAMOV). *Dado* $0 \leq \lambda \leq 1/2$*, existe para cualquier* n *un código binario lineal* $[n, k, d]$ *tal que*

$$\frac{d}{n} \geq \lambda, \ y \ R \geq 1 - H_2\left(\frac{d}{n}\right) \quad \left(R = \frac{k}{n}\right).$$

DEMOSTRACIÓN. Basta con hacer uso de la desigualdad del lema 6.21

$$\sum_{i=0}^{\lambda n} \binom{n}{i} \leq 2^{nH_2(\lambda)}, \quad \lambda \in [0, 1/2]$$

que, interpretada a la luz del teorema 6.27, significa que existe un código lineal binario de distancia mayor o igual que $n\lambda$ y con no más de $nH_2(\lambda)$ símbolos de redundancia, esto es

$$n - k \leq nH_2(\lambda) \leq nH_2\left(\frac{d}{n}\right)$$

o bien

$$R \geq 1 - H_2\left(\frac{d}{n}\right).$$

Vea que cualquier familia de códigos que satisfaga la cota asintótica de Gilbert–Varshamov es no degenerada. ▶

Se conocen varias familias (alternantes, Goppa, doble–circulantes y autoduales) que contienen algunos códigos que alcanzan la cota con igualdad; pero, en general, excepto en los casos triviales $R = 0$ o $R = 1$, se ignora cuáles son estos códigos. Para dar una idea de la dificultad de encontrar por medios sistemáticos códigos eficientes, baste decir que no se obtuvo una construcción explícita de códigos con prestaciones por encima de la cota de Gilbert–Varshamov hasta 1982, y esto sólo para alfabetos de codificación con no menos de 49 elementos. Este método constructivo utiliza conceptos avanzados de geometría algebraica, y se ha convertido en un activo campo de investigación [53].

Las siguientes son condiciones suficientes para poder afirmar que en una familia de códigos se cumple la desigualdad de Gilbert–Varshamov.

Teorema 6.29. *Sea $\mathcal{C}$ una familia de códigos de bloques binarios $(n, 2^k)$ con las propiedades*

a) $\frac{k}{n} \geq R$

b) cualquier vector binario no nulo de longitud n pertenece al mismo número de códigos de $\mathcal{C}$.

En estas condiciones, existen códigos de $\mathcal{C}$ que satisfacen asintóticamente la cota de Gilbert–Varshamov,

$$1 - R = H_2\left(\lim_{n\to\infty} \frac{d}{n}\right).$$

Demostración. Sea N_T el número de códigos de longitud n de $\mathcal{C}$ y N_c el número de códigos que contienen a un vector no nulo cualquiera. Entonces

$$(2^n - 1)N_c = (2^k - 1)N_T.$$

El número de vectores binarios no nulos de longitud n y peso Hamming menor que d es

$$\sum_{i=1}^{d-1} \binom{n}{i}$$

por lo que el número total de códigos con distancia mínima menor que d es como máximo

$$N_c \sum_{i=1}^{d-1} \binom{n}{i}.$$

Si este número es menor que el número total de códigos N_T, entonces se cumple

$$\sum_{i=1}^{d-1} \binom{n}{i} < \frac{2^n - 1}{2^k - 1} < 2^{n-k}$$

y $\mathcal{C}$ contiene un código binario de longitud n y distancia mayor o igual que d. Pero

$$\lim_{n\to\infty} \frac{1}{n} \log_2 \frac{2^n - 1}{2^k - 1} = 1 - R$$

y

$$\lim_{n\to\infty} \frac{1}{n} \log_2 \sum_{i=1}^{d} \binom{n}{i} = H_2\left(\lim_{n\to\infty} \frac{d}{n}\right),$$

lo que prueba que existen códigos de $\mathcal{C}$ que asintóticamente satisfacen

$$1 - R \leq H_2\left(\lim_{n\to\infty} \frac{d}{n}\right). \qquad \blacktriangleright$$

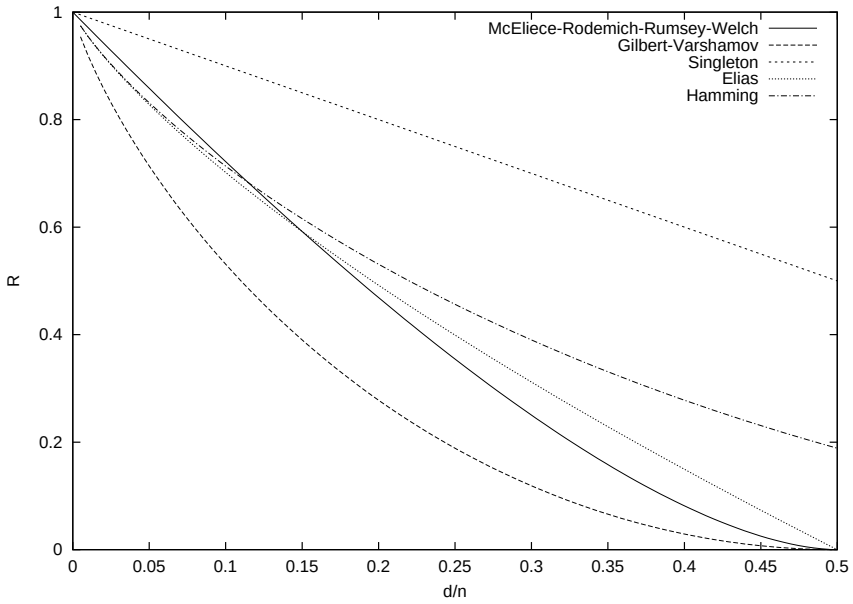


FIGURA 6.2. Cotas asintóticas para los mejores códigos binarios.

Gracias a estas condiciones, se puede dar otra demostración sencilla de la existencia de códigos lineales que satisfacen la inecuación de Gilbert–Varshamov.

TEOREMA 6.30. *Dado* $0 < R < 1$*, existen códigos binarios lineales* $[n, k]$ *con distancia* d *que verifican*

$$R = \frac{k}{n}, \quad 1 - R \leq H_2\left(\frac{d}{n}\right).$$

DEMOSTRACIÓN. Consideremos el conjunto $\mathcal{C}_i$ de todos los códigos lineales binarios de longitud i y dimensión $\lceil Ri \rceil$. Es obvio que

$$\frac{\lceil Ri \rceil}{i} \geq R, \quad \forall i \geq 1$$

de modo que la primera condición del teorema 6.29 es verdadera.

Para comprobar que la segunda también lo es, y por tanto la veracidad del enunciado, tomemos un vector binario no nulo de longitud i. Este vector pertenece a

$$\frac{(2^i - 2)(2^i - 4) \cdots (2^i - 2^{k-2})}{(2^k - 1)(2^k - 2) \cdots (2^k - 2^{k-1})}$$

subespacios lineales distintos de dimensión k, pues los $k-1$ restantes vectores de una base del subespacio se pueden escoger de $(2^i-2)(2^i-4)\cdots(2^i-2^{k-2})$ maneras diferentes, y hay exactamente $(2^k-1)(2^k-2)\cdots(2^k-2^{k-1})$ conjuntos de k vectores linealmente independientes en cada código. ▶

La mejor estimación, por exceso, de la distancia de un código de bloques es la cota superior de McEliece–Rodemich–Rumsey–Welch, que no se va a formular en este texto aunque se puede consultar, por ejemplo, en [41]. Junto con la inferior de Gilbert–Varshamov y las de Singleton, Hamming y Elias, se ha representado en la figura 6.2. Todos los códigos de bloques se sitúan sobre o por debajo de la gráfica de la cota de McEliece, mientras que sólo los mejores códigos lineales se sitúan por encima de la curva de Gilbert–Varshamov.

CAPÍTULO 7

Códigos cíclicos

Los códigos cíclicos son un subconjunto de los códigos lineales que, por gozar de una estructura matemática más restrictiva, presentan una significativa ventaja: sus algoritmos de codificación y, en especial, de decodificación admiten una realización más simple. La de los cíclicos es por ello una de las familias de códigos más útiles; contiene buenos grupos de códigos (como los Golay, BCH, Reed–Solomon, de residuo cuadrático, ...), que se emplean solos o como componentes en esquemas de codificación iterativos (códigos concatenados).

En este capítulo se definen y caracterizan los códigos cíclicos enfatizando su carácter lineal. Se describen asimismo los circuitos generadores y comprobadores de redundancia; se identifican luego las propiedades de control de errores; y, para finalizar, se analizan los fundamentos de los algoritmos de decodificación.

7.1. Definición

El código lineal $\mathcal{C}$ es cíclico si es invariante para cualquier rotación de sus coordenadas.

Definición 7.1 (Código cíclico). *Un código lineal $\mathcal{C}[n,k]$ sobre un alfabeto finito $\mathcal{S}$ es un código cíclico si cualquier desplazamiento cíclico de una palabra del código es también una palabra del código (propiedad cíclica):*

$$(v_{n-1}, v_{n-2}, \ldots, v_0) \in \mathcal{C} \quad \Rightarrow \quad (v_{n-2}, \ldots, v_0, v_{n-1}) \in \mathcal{C}, \quad v_i \in S \quad \forall i.$$

Debido a la condición de linealidad, se supone implícitamente que sobre

los elementos de $\mathcal{S}$ hay definidas dos operaciones con las cuales se dota a $\mathcal{S}$ de estructura de cuerpo finito. En particular, para códigos binarios, $\mathcal{S} = \mathrm{GF}(2)$.

Un vector $\mathbf{v} = (v_{n-1}, \ldots, v_0)$ de un código cíclico binario admite una representación unívoca en forma de polinomio en la variable x con coeficientes en GF(2):

$$v(x) = v_{n-1}x^{n-1} + v_{n-2}x^{n-2} + \cdots + v_0$$

en la que el subíndice de cada símbolo se ha hecho coincidir, por simple convenio notacional, con el exponente de su término polinómico correspondiente.

Es el caso que para cualesquiera escalares $\alpha, \beta \in \mathrm{GF}(2)$ y cualesquiera vectores $\mathbf{v}_1, \mathbf{v}_2$ de un código lineal, la representación polinómica del vector del código $\alpha\mathbf{v}_1 + \beta\mathbf{v}_2$ es el polinomio $\alpha v_1(x) + \beta v_2(x)$; por lo tanto, el conjunto $F_2[x]$ de polinomios con coeficientes binarios es un espacio vectorial isomorfo (tiene idéntica estructura algebraica) al espacio $\mathcal{L}_n$ de vectores binarios, de donde se infiere que resulta indistinto utilizar la notación vectorial o la notación polinómica para operar con las palabras del código.

Sin embargo, la representación polinómica de los vectores no se ha introducido arbitrariamente, sino porque resulta extremadamente útil para la manipulación algebraica de los desplazamientos cíclicos.

TEOREMA 7.1. *El polinomio correspondiente a un desplazamiento cíclico de la palabra del código $v(x)$ es el resto de la división de $xv(x)$ entre $x^n - 1$.*

DEMOSTRACIÓN. Operando directamente se tiene que

$$\begin{aligned} xv(x) &= v_{n-1}x^n + v_{n-2}x^{n-1} + \cdots + v_0x \\ &= v_{n-1}(x^n - 1) + v_{n-2}x^{n-1} + \cdots + v_0x + v_{n-1}. \end{aligned}$$

El resto de dividir entre $x^n - 1$ vale, en efecto,

$$xv(x) \mod (x^n - 1) = v_{n-2}x^{n-1} + \cdots + v_0x + v_{n-1}. \qquad \blacktriangleright$$

Análogamente, podemos escribir para cualquier $i > 1$ y cualquier polinomio del código $v(x)$,

$$\begin{aligned} x^iv(x) &= v_{n-1}x^{n+i-1} + \cdots + v_0x^i \\ &= (v_{n-1}x^{i-1} + v_{n-2}x^{i-2} + \cdots + v_{n-i+1}x + v_{n-i})(x^n - 1) \\ &+ v_{n-i-1}x^{n-1} + \cdots + v_0x^i + v_{n-1}x^{i-1} + \cdots + v_{n-i}. \end{aligned}$$

Y como los coeficientes del polinomio resto

$$v_{n-i-1}x^{n-1} + v_{n-i-2}x^{n-2} + \cdots + v_0x^i + v_{n-1}x^{i-1} + \cdots + v_{n-i}$$

son los de $\mathbf{v}$ desplazados cíclicamente i posiciones, podemos afirmar en general que:

Teorema 7.2. *El desplazamiento cíclico de i posiciones de $v(x)$ es*

$$v^{(i)}(x) = x^i v(x) \mod x^n - 1$$

para $i = 1, \ldots, n-1$.

Aunque la operación de desplazamiento cíclico de un vector se ha presentado como una rotación cíclica en un solo sentido (hacia la izquierda), es obvio que una rotación de i posiciones en uno de los sentidos es equivalente a una rotación de $n - i$ posiciones en sentido contrario. En este capítulo, por coherencia, un desplazamiento cíclico de i posiciones será siempre la operación simbólica definida en el teorema 7.2.

La siguiente definición es innecesaria en el caso de los códigos cíclicos binarios, pero se incluye aquí por generalidad.

Definición 7.2 (Polinomio mónico). *Se denomina polinomio mónico a aquél cuyo coeficiente del término de mayor grado es 1.*

El motivo de hacerlo así sólo se aclarará tras exponer el teorema 7.4. De momento, mostraremos que, debido a la linealidad y a la propiedad cíclica, algunos polinomios del código satisfacen ciertas restricciones algebraicas.

Teorema 7.3. *En un código cíclico, el polinomio mónico de menor grado del código es único y su término independiente es no nulo.*

Demostración. Sean $g_1(x)$ y $g_2(x)$ dos polinomios mónicos de un código cíclico $\mathcal{C}$, distintos y ambos de grado m mínimo. Por la linealidad de $\mathcal{C}$, $g_1(x) - g_2(x)$ representa una palabra del código y es de grado $m - 1$ como máximo; luego $g_1(x) - g_2(x) \neq 0$ es de grado menor que m, contra la hipótesis.

Por otra parte, si fuese $g_0 = 0$, entonces $g(x) = x(x^{m-1} + g_{m-1}x^{m-2} + \cdots + g_1)$; y existiría, por la propiedad cíclica, un polinomio mónico del código de grado menor que m, $x^{m-1} + g_{m-1}x^{m-2} + \cdots + g_1$, lo que está en contradicción con la premisa de que m es mínimo. ▶

La característica fundamental de $g(x)$ es que a partir de él se pueden obtener todas las demás palabras del código.[1] El polinomio mónico de menor grado de un código cíclico se llama *polinomio generador*, y se suele representar por $g(x)$.

TEOREMA 7.4 (CARACTERIZACIÓN DE UN CÓDIGO CÍCLICO).

a) Un código lineal $\mathcal{C}[n,k]$ es cíclico si y solamente si todas las palabras del código son múltiplos de $g(x)$, el polinomio mónico de menor grado del código.

b) El polinomio generador $g(x)$ divide a x^n-1 y es de grado $n-k$.

DEMOSTRACIÓN. Sea $m = \text{grado}(g(x))$. Para cualquier $a(x) \in F_2[x]$ de grado menor que $n-m$, el producto

$$a(x)g(x) = \sum_{i=0}^{n-m-1} a_i x^i g(x) = \sum_{i=0}^{n-m-1} a_i \big(x^i g(x)\big)$$

es una combinación lineal de las palabras del código $x^i g(x)$; y por tanto, en virtud de la condición de linealidad, cualquier múltiplo de $g(x)$ con grado inferior a n es también una palabra del código.

Por otra parte, cualquier polinomio del código $v(x)$ se puede escribir, de manera única (merced al algoritmo de división euclídeo), como $v(x) = a(x)g(x) + r(x)$, siendo $a(x)$ el polinomio cociente de la división de $v(x)$ entre $g(x)$, y $r(x)$ el polinomio resto. El resto es o bien el polinomio nulo o bien un polinomio de grado menor que el de $g(x)$. Si $r(x)$ no fuese nulo, entonces $r(x) = v(x) - a(x)g(x)$ sería una palabra del código de grado menor que $g(x)$, lo que es imposible por definición. En consecuencia, $r(x) = 0$ y todo polinomio del código es divisible por $g(x)$, es decir, $v(x) = a(x)g(x)$.

Puesto que $v(x) = a(x)g(x)$, es fácil deducir de aquí que, para engendrar las 2^k palabras de un código binario $[n,k]$, es preciso que el grado de $g(x)$ sea $n-k$.

Por último, si se divide $x^k g(x)$ entre x^n-1,

$$x^k g(x) = x^n - 1 + r(x)$$

el resto $r(x) = g^{(k)}(x) = x^k g(x) \mod x^n - 1$ es la palabra del código que resulta del desplazamiento cíclico de k posiciones de $g(x)$ y, por todo lo anterior, contiene a $g(x)$ como factor. Por lo tanto, $x^n - 1 = x^k g(x) - r(x) = x^k g(x) - b(x)g(x) = h(x)g(x)$, siendo $h(x)$ un polinomio mónico de grado k con término independiente no nulo. Así pues, el polinomio generador es un divisor de $x^n - 1$. ▶

[1] Pero $g(x)$ no es el *único* polinomio del código con esta propiedad.

Este teorema nos capacita para resolver dos cuestiones básicas:

- Cómo identificar un código cíclico, y cómo calcular sus vectores. Todos los polinomios del código de un subespacio cíclico $[n, k]$ son múltiplos del polinomio generador, y éste es el polinomio mónico de grado mínimo del código. Obsérvese que esto no quiere decir que sea el de menor peso Hamming, aun cuando a veces coinciden.
- Cómo construir códigos cíclicos. Cualquier polinomio mónico de grado $n-k$ con término independiente distinto de cero y divisor de x^n-1 genera un código cíclico $[n, k]$. En otras palabras, existe una relación uno a uno entre los códigos cíclicos de longitud n y los divisores mónicos de $x^n - 1$.[2]

La propiedad algebraica que liga a todos los polinomios de un código cíclico puede enunciarse, además, de esta otra manera equivalente: si $v(x)$ es un polinomio del código y α es una raíz de $g(x)$, entonces α es también raíz de $v(x)$, por ser $v(\alpha) = b(\alpha)g(\alpha) = 0b(\alpha) = 0$. Ocurre, en consecuencia, que los polinomios no nulos de un código cíclico son aquéllos y sólo aquéllos que admiten como raíces a todas las del polinomio generador con la misma o mayor multiplicidad, raíces que, conocida la longitud n, definen el código. Pese a que permite desarrollar un cuerpo de teoría más general, esta caracterización no resulta esencial en el presente capítulo, y no insistiremos por el momento más en ella. Habrá ocasión de estudiar más adelante, en los capítulos 10 y 11, dos importantes familias de códigos cíclicos definidas precisamente por los ceros de su polinomio generador.

En este punto, una cuestión natural es la de averiguar si, dados cualesquiera valores de n y k, existe algún código cíclico $[n, k]$. El resultado que se acaba de presentar garantiza la existencia cuando $x^n - 1$ contiene un factor de grado $n - k$. El problema de la búsqueda de códigos cíclicos binarios se transforma así en otro equivalente, aunque del mismo grado de dificultad: la factorización de $x^n - 1$ en polinomios mónicos irreducibles binarios. Para que la descomposición resulte fácil de describir, es condición básica que $\mathrm{mcd}(q, n) = 1$, siendo q el cardinal del alfabeto de codificación. Por tanto, en los códigos binarios ($q = 2$), supondremos siempre que n es impar.

Ejemplo 7.1.

a) El polinomio $x^n - 1$ admite siempre estos dos divisores

$$x^n - 1 = (x - 1)(1 + x + \cdots + x^{n-1}).$$

[2]Para no incurrir en problemas formales, definiremos el polinomio generador del código $\mathcal{C} = \{\mathbf{0}\}$ como $x^n - 1$.

Mensaje	Vector del código	Polinomio del código
0000	0000000	$0 = 0g(x)$
0001	0001011	$x^3 + x + 1 = g(x)$
0010	0010110	$x^4 + x^2 + x = xg(x)$
0011	0011101	$x^4 + x^3 + x^2 + 1 = (x + 1)g(x)$
0100	0101100	$x^5 + x^3 + x^2 = x^2g(x)$
0101	0100111	$x^5 + x^2 + x + 1 = (x^2 + 1)g(x)$
0110	0111010	$x^5 + x^4 + x^3 + x = (x^2 + x)g(x)$
0111	0110001	$x^5 + x^4 + 1 = (x^2 + x + 1)g(x)$
1000	1011000	$x^6 + x^4 + x^3 = x^3g(x)$
1001	1010011	$x^6 + x^4 + x + 1 = (x^3 + 1)g(x)$
1010	1001110	$x^6 + x^3 + x^2 + x = (x^3 + x)g(x)$
1011	1000101	$x^6 + x^2 + 1 = (x^3 + x + 1)g(x)$
1100	1110100	$x^6 + x^5 + x^4 + x^2 = (x^3 + x^2)g(x)$
1101	1111111	$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 =$ $= (x^3 + x^2 + 1)g(x)$
1110	1100010	$x^6 + x^5 + x = (x^3 + x^2 + x)g(x)$
1111	1101001	$x^6 + x^5 + x^3 + 1 = (x^3 + x^2 + x + 1)g(x)$

Tabla 7.1. Palabras y polinomios código del código cíclico binario [7,4] generado por $g(x) = x^3 + x + 1$.

dimensión	polinomio generador
1	$1 + x + x^2 + \cdots + x^6$
3	$1 + x^2 + x^3 + x^4$
3	$1 + x + x^2 + x^4$
4	$1 + x + x^3$
4	$1 + x^2 + x^3$
6	$1 + x$
7	1

Tabla 7.2. Códigos cíclicos binarios de longitud 7.

Peso	0	7	8	11	12	15	16	23
Palabras del código	1	253	506	1288	1288	506	253	1

Tabla 7.3. Distribución de pesos del código Golay$(23, 12)$.

El polinomio $1+x+\cdots+x^{n-1}$ genera el código binario de repetición de longitud n, que es obviamente cíclico. El factor $x-1$ genera el código cíclico compuesto por los vectores de peso par de $\mathcal{L}_n$. Estos dos códigos son duales.

b) En la tabla 7.1 se han incluido las palabras de un código cíclico binario $[7,4]$ con polinomio generador $g(x)=x^3+x+1$, así como sus respectivas representaciones polinómicas. Verifique que se trata realmente un código cíclico.

c) La tabla 7.2 contiene todos los códigos cíclicos binarios de longitud 7. Los dos códigos de dimensión 4 son códigos Hamming $[7,4,3]$.

d) El código Golay$(23,12)$ es un código cíclico binario perfecto de distancia 7 generado por

$$g_1(x)=x^{11}+x^9+x^7+x^6+x^5+x+1$$

o bien por

$$g_2(x)=x^{11}+x^{10}+x^6+x^5+x^4+x^2+1=x^{11}g_1(x^{-1}).$$

Las palabras del código generadas por $g_2(x)$ son las mismas que las engendradas por $g_1(x)$, pero escribiendo los símbolos en orden inverso, de manera que ambos códigos son equivalentes (por permutación). La tabla 7.3 muestra la distribución simétrica de los pesos de las palabras del código. ■

7.2. Descripción matricial

Todo código cíclico, por ser lineal, admite la caracterización vista en el capítulo anterior en forma de subespacio lineal y de su matriz asociada. Examinemos qué forma particular adoptan, en el caso de los códigos cíclicos, las matrices generadora y comprobadora de paridad, y cómo pueden obtenerse a partir del conocimiento del polinomio generador.

Teniendo en cuenta que $x^i g(x)$, para $i=0,\ldots,k-1$, son k palabras del código linealmente independientes, y considerando que el polinomio generador es de la forma $g(x)=x^{n-k}+g_{n-k-1}x^{n-k-1}+\cdots+g_1x+g_0$, con $g_0\neq 0$, una matriz generadora del código cíclico $[n,k]$ engendrado por $g(x)$ es

$$G_{k\times n}=\begin{pmatrix} 1 & g_{n-k-1} & \cdots & g_0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & g_{n-k-1} & \cdots & g_0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 1 & g_{n-k-1} & \cdots & g_0 \end{pmatrix}$$

que, evidentemente, no es sistemática. Las filas de esta matriz son los coeficientes del polinomio generador y sus $k-1$ primeros desplazamientos, $x^i g(x)$, $i = 0, \ldots, k-1$, considerados como polinomios de grado $n-1$.

La matriz comprobadora de paridad del código puede deducirse partiendo de la relación $g(x)h(x) = 0 \mod x^n - 1$, que implica

$$v(x) \in \mathcal{C}(n,k) \Leftrightarrow v(x)h(x) = 0 \mod x^n - 1 \tag{7.1}$$

ya que $v(x)h(x) = a(x)g(x)h(x) = 0 \mod x^n - 1$.

La ecuación (7.1) significa que $v(x)h(x)$ es un polinomio cuyos coeficientes de grado $k \leq i < n$ son nulos solamente si $v(x)$ es una palabra del código. El desarrollo algebraico del producto entre $v(x) = \sum_{i=0}^{n-1} v_i x^i$ y $h(x) = \sum_{i=0}^{k} h_i x^i$ vale

$$v(x)h(x) = \sum_{i=0}^{n+k-1} c_i x^i = \sum_{i=0}^{n+k-1} x^i \sum_{j=0}^{i} h_j v_{i-j},$$

de manera que $v(x)$ es un polinomio del código sólo cuando

$$\sum_{j=0}^{i} h_j v_{i-j} = 0, \quad \forall i = k, \ldots, n-1. \tag{7.2}$$

El sistema de ecuaciones (7.2) significa que el vector $(v_{n-1}, \ldots, v_1, v_0)$ es ortogonal a los desplazamientos cíclicos de $(0, 0, \ldots, h_0, h_1, \ldots, h_k)$. Escritas en forma matricial, estas $n-k$ ecuaciones se convierten en

$$\mathbf{v} \cdot H^T = \mathbf{0}_{n-k}$$

con

$$H_{n-k \times n} = \begin{pmatrix} h_0 & h_1 & \ldots & h_{k-1} & 1 & 0 & \ldots & 0 \\ 0 & h_0 & h_1 & \ldots & h_{k-1} & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & \ldots & h_0 & h_1 & \ldots & h_{k-1} & 1 \end{pmatrix}. \tag{7.3}$$

Resulta interesante observar que la última fila de H es el polinomio $h_r(x) = h_0 x^k + h_1 x^{k-1} + \cdots + h_{k-1} x + h_k = x^k h(x^{-1})$, es decir, es el polinomio recíproco de $h(x)$; y que las filas de G son ortogonales a las de H. H es, naturalmente, una matriz de comprobación de paridad de $\mathcal{C}[n,k]$ (generadora de $\mathcal{C}^{\perp}[n, n-k]$), por lo que el polinomio recíproco $h_r(x)$ es el polinomio generador del código cíclico dual del $\mathcal{C}[n,k]$.

En general, los polinomios $g(x)$ y $h(x)$ no son ortogonales. Ahora bien, como $h(x)$ es un polinomio mónico de grado k divisor de $x^n - 1$, es a su

$$\begin{array}{ccccc}
\mathcal{C}=\langle g(x)\rangle & \longrightarrow & \begin{array}{c}\mathcal{C}+\mathcal{D}=\mathcal{L}_n\\ \mathcal{C}\cap\mathcal{D}=\{\mathbf{0}\}\end{array} & \longleftarrow & \mathcal{D}=\langle h(x)\rangle \\
\mathcal{C}\pi_{-1}\downarrow\uparrow\mathcal{D}^{\perp}\pi_{-1} & & & & \mathcal{C}^{\perp}\pi_{-1}\uparrow\downarrow\mathcal{D}\pi_{-1} \\
\mathcal{D}^{\perp}=\langle\dfrac{x^{n-k}g(x^{-1})}{g_0}\rangle & \longrightarrow & \begin{array}{c}\mathcal{C}^{\perp}+\mathcal{D}^{\perp}=\mathcal{L}_n\\ \mathcal{C}^{\perp}\cap\mathcal{D}^{\perp}=\{\mathbf{0}\}\end{array} & \longleftarrow & \mathcal{C}^{\perp}=\langle\dfrac{x^{k}h(x^{-1})}{h_0}\rangle
\end{array}$$

Tabla 7.4. Relaciones entre los códigos cíclicos engendrados por $g(x)$, $h(x)$ y por sus polinomios recíprocos cuando $g(x)h(x)=x^n-1$. $\mathcal{C}\pi_{-1}$ es el código que resulta al aplicar a todas las palabras de $\mathcal{C}$ la permutación de coordenadas $i\mapsto -i \mod n$, es decir, al escribir las palabras de $\mathcal{C}$ en sentido contrario.

vez generador de un código cíclico $[n,n-k]$, cuya matriz generadora tendrá por filas los coeficientes de $h(x)$ y sus sucesivos desplazamientos. El código engendrado por $h(x)$ es equivalente al código dual del engendrado por $g(x)$. En realidad, está formado por todas las palabras de éste último código, escritos sus elementos en orden inverso, lo que no es más que una permutación de los símbolos en todos sus vectores.[3]

Ejemplo 7.2. Tomando de nuevo el polinomio generador $g(x)=x^3+x+1$, una matriz generadora del código $[7,4]$ dado por él es

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

que tiene por filas a los polinomios código $x^ig(x), i=0,1,2,3$. Es fácil comprobar que la factorización de x^7-1 en polinomios binarios irreducibles da

$$x^7-1=(x+1)(x^3+x^2+1)(x^3+x+1)$$

[3]La relación entre ambos es más fuerte. Sea $\mathcal{C}'$ el conjunto formado por todos los vectores que no pertenecen al código cíclico $\mathcal{C}$. $\mathcal{C}'$ es un código cíclico llamado *código complementario*, que por construcción satisface

$$\begin{aligned}\mathcal{C}+\mathcal{C}'&=\mathcal{L}_n\\ \mathcal{C}\cap\mathcal{C}'&=\{\mathbf{0}\}.\end{aligned}$$

Pues bien, $\mathcal{C}'$ es el código generado por $h(x)$.

por lo que el polinomio $h(x)$ vale $x^4 + x^2 + x + 1$, y el polinomio recíproco es $x^4 h(x^{-1}) = x^4 + x^3 + x^2 + 1$. Entonces, es posible construir una matriz comprobadora de paridad escribiendo en sus filas los coeficientes desplazados de $x^4 h(x^{-1})$:

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

Por supuesto, otra alternativa es, como se sabe, obtener una matriz generadora del código en forma sistemática, $G' = [I_k \; A_{k\times(n-k)}]$, y escribir a partir de ella una matriz ortogonal $H' = [-A^T \; I_{n-k}]$. Las palabras del código dual $[7, 3]$ son las combinaciones lineales de las filas de H. Se indican en la tabla siguiente.

Código dual	
0000000	1010011
1101001	0100111
1001110	0011101
0111010	1110100

¿De qué código es H matriz comprobadora de paridad? ■

7.3. Códigos cíclicos sistemáticos

El método de multiplicar el mensaje por el polinomio generador para construir los polinomios código no produce, en general, un código sistemático. Pero es posible hallar una técnica de codificación sistemática (y una matriz generadora sistemática) con el procedimiento siguiente.

Supóngase que la matriz sistemática

$$G = \begin{pmatrix} I_k & A_{k\times(n-k)} \end{pmatrix} = \left(\begin{array}{c|ccc} & a_{1,1} & \dots & a_{1,n-k} \\ & a_{2,1} & \dots & a_{2,n-k} \\ I_k & \dots & \dots & \dots \\ & a_{k,1} & \dots & a_{k,n-k} \end{array} \right)$$

es generadora de un código cíclico. Sus filas son, entonces, los coeficientes de k polinomios (palabras del código) linealmente independientes. Pero los elementos de la fila $i = 1, \dots, k$ representan al polinomio

$$f_i(x) = x^{n-i} + \sum_{j=1}^{n-k} a_{i,j} x^{n-k-j} = x^{n-i} + r_i(x) \tag{7.4}$$

siendo $r_i(x)$ de grado máximo $n-k-1$. Por ser $f_i(x)$ una palabra del código,

$$f_i(x) = 0 \mod g(x).$$

Introduciendo (7.4) en esta expresión, se tiene

$$f_i(x) = x^{n-i} + r_i(x) = 0 \mod g(x)$$

de donde resulta que el polinomio $-r_i(x)$ es el resto de la división de x^{n-i} entre el polinomio generador $g(x)$.

Para codificar el mensaje $m(x) = m_{k-1}x^{k-1} + \cdots + m_1x + m_0$, basta con considerar que, aplicando la propiedad de linealidad, la suma de los polinomios del código

$$\begin{array}{l} m_{k-1}[x^{n-1} + (x^{n-1} \mod g(x))] \\ m_{k-2}[x^{n-2} + (x^{n-2} \mod g(x))] \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ m_0[x^{n-k} + (x^{n-k} \mod g(x))] \end{array}$$

es un polinomio del código cuyos k primeros coeficientes son justamente los símbolos $(m_{k-1}, \ldots, m_0)$ del mensaje. Pero esta suma vale

$$m_{k-1}x^{n-1} + \cdots + m_0x^{n-k} + [(m_{k-1}x^{n-1} + \cdots + m_0x^{n-k}) \mod g(x)]$$

lo que significa que la redundancia es

$$m_{k-1}x^{n-1} + \cdots + m_1x^{n-k+1} + m_0x^{n-k} = x^{n-k}m(x) \mod g(x).$$

En resumen, para calcular, en un código cíclico sistemático, los dígitos de redundancia de un bloque cualquiera de k bits, representado por el polinomio $m(x)$ de grado máximo $k-1$, se deben efectuar las siguientes operaciones:

1. Calcular $x^{n-k}m(x)$, o, de manera más gráfica, desplazar los símbolos del mensaje $n-k$ posiciones hacia la izquierda.

2. Obtener el resto de la división del mensaje desplazado entre el polinomio generador, $r_m(x) = x^{n-k}m(x) \mod g(x)$.

3. Sumar el resto $-r_m(x)$ a $x^{n-k}m(x)$, o lo que es igual, concatenar los coeficientes de $-r_m(x)$ tras los de $m(x)$: el resultado es la palabra del código a transmitir.

Ejemplo 7.3. El código del ejemplo 7.1-(b) no es sistemático. Según se acaba de explicar, dividiendo x^i, para $i = 3, 4, 5, 6$, entre $g(x) = x^3 + x + 1$,

y sumando el resto a x^i, se calculan los coeficientes de las filas de la matriz generadora sistemática. Así pues, como

$$\begin{aligned} x^6 &= x^2 + 1 \mod g(x) \\ x^5 &= x^2 + x + 1 \mod g(x) \\ x^4 &= x^2 + x \mod g(x) \\ x^3 &= x + 1 \mod g(x) \end{aligned}$$

esa matriz vale

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

Supongamos ahora que se desea calcular, para el código cíclico sistemático, el polinomio del código correspondiente al mensaje 1101, o lo que es lo mismo, de $m(x) = x^3 + x^2 + 1$. Los pasos del algoritmo son:

1. Multiplicar $m(x)$ por x^3. El resultado es $x^3 m(x) = x^6 + x^5 + x^3$.
2. Calcular el resto de la división de $x^3 m(x)$ entre $g(x)$. Como
$$x^6 + x^5 + x^3 = (x^3 + x^2 + x + 1)g(x) + 1$$
los dígitos del resto son 001.
3. Sumar el resto a $x^3 m(x)$. El vector del código resultante es 1101001.

Por supuesto,

$$(1101001) = (1101) \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

pero se mostrará en el próximo apartado que el circuito de cálculo del resto de la división polinómica es más simple que el codificador de un código lineal no cíclico. ■

7.4. Circuitos codificadores

El cálculo de los símbolos de redundancia de un código cíclico sistemático se realiza en la práctica con un registro de desplazamiento realimentado y con sumadores con aritmética finita. Existen dos formas de generar la redundancia:

a) Con un circuito lineal multiplicativo basado en las ecuaciones de comprobación de paridad $\mathbf{v} \cdot H^T = \mathbf{0}_{n-k}$.

b) Mediante un circuito lineal divisor que lleva a cabo el cálculo $R(x) = x^{n-k}m(x) \mod g(x)$.

Se expone a continuación el funcionamiento de los circuitos con cada método.

7.4.1. Circuitos multiplicadores

El procedimiento más directo para obtener las palabras de un código cíclico $\mathcal{C}[n,k]$ es la multiplicación del polinomio mensaje por el polinomio generador. En la figura 7.1 (página 210) se representa la estructura de un filtro lineal de respuesta impulsional finita que lleva a cabo la multiplicación de dos polinomios cualesquiera, $m(x)$ y $g(x) = g_m x^m + \cdots + g_1 x + g_0$. Aplicado al caso de un código cíclico binario, el funcionamiento es como se pasa a describir.

Inicialmente, las $n-k$ celdas de memoria (las cajas rectangulares, cada una de las cuales permite almacenar un símbolo) están a cero. Los k símbolos del polinomio mensaje se introducen en el circuito en secuencia, comenzando por el del término de mayor exponente. A continuación, el bit contenido en cada posición de memoria se multiplica (módulo-2) por el coeficiente correspondiente de $g(x)$, y el resultado de todos estos productos se suma (módulo-2) para obtener un bit del polinomio producto. Al final de cada etapa, el contenido global del registro de memoria se desplaza una posición hacia la derecha. Se requieren en total n etapas para producir la palabra del código, y durante las $n-k$ últimas es preciso introducir ceros a la entrada del codificador. El inconveniente más severo de esta clase de codificadores es que no son sistemáticos.

Consideremos ahora las $n-k$ ecuaciones lineales

$$\mathbf{v} \cdot H^T = \mathbf{0}_{n-k}$$

y recordemos la forma de la matriz H, introducida en el apartado 7.2. A partir de la estructura particular de H se ve que los símbolos de redundancia $v_0, v_1, \ldots, v_{n-k-1}$, en un código sistemático, se pueden calcular de manera recurrente multiplicando los símbolos $(v_{n-1}, \ldots, v_{n-k})$ del mensaje por los sucesivos desplazamientos de los coeficientes del polinomio recíproco $h_r(x)$. Esta operación produce como resultado los coeficientes de orden $k \leq i < n$ del polinomio resultante de multiplicar el polinomio del código, $v(x)$, por el polinomio comprobador de paridad $h(x)$.

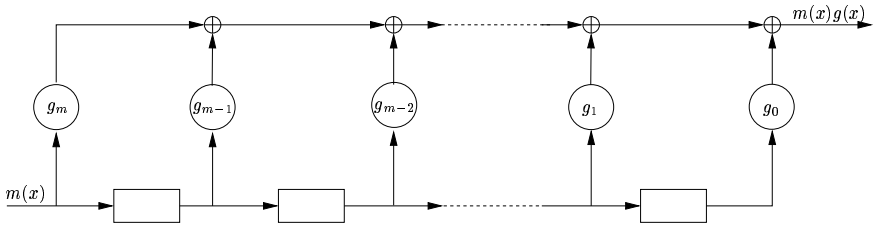


FIGURA 7.1. Circuito genérico para multiplicar los polinomios $m(x)$ y $g(x)$.

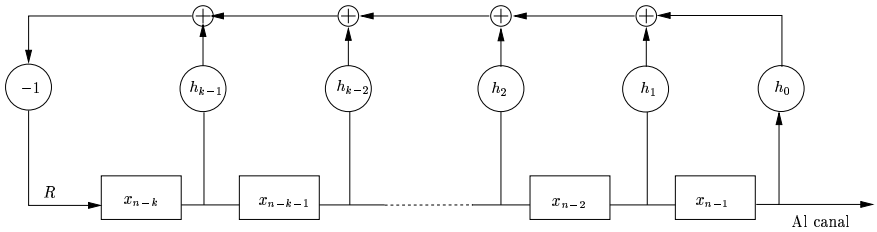


FIGURA 7.2. Circuito codificador basado en el polinomio de comprobación de paridad $h(x)$.

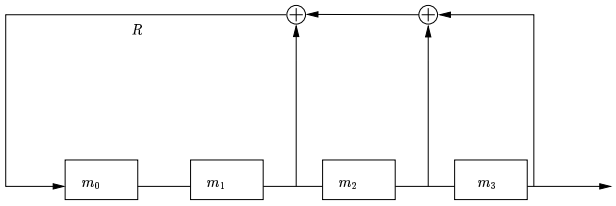


FIGURA 7.3. Codificador multiplicativo del código cíclico binario de polinomio generador $g(x) = x^3 + x + 1$.

El producto

$$v(x)h(x) = \sum_{i=0}^{n+k-1} c_i x^i = \sum_{i=0}^{n+k-1} x^i \sum_{j=0}^{k} h_j v_{i-j} = \sum_{i=0}^{n+k-1} x^i \sum_{j=0}^{n-1} h_{i-j} v_j$$

implica, para todo polinomio del código, que

$$\sum_{j=0}^{k} h_j v_{i-j} = 0, \quad \forall i = n-1, \ldots, k$$

y al ser $h_k = 1$

$$v_{i-k} = -\sum_{j=0}^{k-1} h_j v_{i-j}, \quad \forall i = n-1, \ldots, k \tag{7.5}$$

que son las ecuaciones recurrentes anunciadas para calcular los dígitos de redundancia en función de los símbolos del mensaje y los símbolos de redundancia obtenidos en las iteraciones previas.

En la figura 7.2 se ha dibujado el esquema de un circuito lineal genérico para llevar a cabo el cálculo de la recurrencia (7.5). Cada una de las cajas representa una celda de memoria capaz de almacenar un bit (un biestable), cada círculo es un multiplicador binario (una puerta lógica AND), y todos los sumadores emplean aritmética módulo-2 (puertas lógicas XOR).

Los k bits del mensaje se almacenan inicialmente en las celdas del registro de desplazamiento, con el primer símbolo, m_{k-1}, en la posición x_{n-1}. En cada etapa $j = 1, \ldots, n-k$ se realizan las multiplicaciones y sumas indicadas, con lo que el j-ésimo coeficiente de redundancia, v_{n-k-j}, está disponible en el punto R. A continuación, el contenido de cada celda se desplaza una posición a la derecha, al tiempo que v_{n-k-j} entra en la primera celda de la memoria. En cada iteración sucesiva se calcula un nuevo bit de redundancia, temporalmente almacenado en la primera celda, cuyo valor depende de los bits del mensaje y de todos los bits de redundancia generados antes que él. Después de k iteraciones, la posición x_{n-1} contiene el primer bit de redundancia, que se extrae del circuito por la rama de salida a la vez que se realimenta para continuar calculando los dígitos de redundancia restantes. Al cabo de $n-k$ desplazamientos más, todos los bits de redundancia habrán sido calculados y transmitidos por el circuito.

El circuito de multiplicación realimentado es sencillamente un filtro lineal recurrente (o autorregresivo) con aritmética binaria. No se debe olvidar que, en codificadores binarios, $h_0 = 1$ y $-1 = 1$.

Ejemplo 7.4. El codificador multiplicativo correspondiente a $g(x) = x^3 + x + 1$ se ha dibujado en la figura 7.3. El proceso de cálculo de la redundancia

del mensaje 1010 se resume en la tabla adjunta, donde figuran el contenido del registro de desplazamiento y el valor del bit de realimentación en cada etapa.

Iteración	$R = m_1 + m_2 + m_3$	m_0	m_1	m_2	m_3
1		0	1	0	1
2	0	0	0	1	0
3	1	1	0	0	1
4	1	**1**	**1**	**0**	0

Al término de la etapa 4, los biestables $m_0 m_1 m_2$ guardan el valor de los bits de redundancia del mensaje. ■

7.4.2. Circuitos divisores

Los bits de redundancia de un código sistemático también pueden ser generados realizando explícitamente la división del polinomio $x^{n-k}m(x)$ entre el generador $g(x)$ mediante un registro de desplazamiento realimentado de $n-k$ posiciones como el de la figura 7.4.

El registro tiene sus $n-k$ celdas de memoria inicialmente a 0. Los coeficientes del polinomio desplazado $x^{n-k}m(x)$ se introducen en el circuito por la izquierda, en orden de mayor a menor exponente, de manera que los primeros $n-k$ desplazamientos sencillamente sirven para almacenar los $n-k$ primeros símbolos de $x^{n-k}m(x)$ en el registro, porque el coeficiente de la rama de realimentación vale siempre cero.

En cada una de las siguientes k iteraciones, el símbolo de realimentación es el coeficiente $j = k, k-1, \ldots, 1$ del polinomio cociente, que se multiplica por $g(x)$, y el resultado se sustrae (recuérdese que la suma y la resta en módulo-2 son la misma operación) del resto parcial almacenado en el registro. El resultado se desplaza una posición a la derecha para dar lugar a la entrada de un nuevo bit, y se continúa con la siguiente iteración. Después de haber introducido los n dígitos en el circuito, el registro de desplazamiento contiene los bits de redundancia, que se extraen mediante el conmutador de la línea de realimentación realizando $n-k$ desplazamientos más. En códigos binarios, $1 = -1$ y $g_0 = 1$.

Formalmente, el circuito ejecuta el siguiente par de ecuaciones recurrentes: sea $Q^{(r)}(x)$ el cociente parcial, en la iteración r, de la división polinómica de $v(x)$ entre $g(x)$, con $v(x)$ y $g(x)$ arbitrarios; y sea $R^{(r)}(x)$ el resto parcial en la iteración r de esa misma división. El cociente y el resto parciales en esta iteración se obtienen a partir de los de la etapa anterior con las

ecuaciones:

$$Q^{(r)}(x) = Q^{(r-1)}(x) + R_{n-r}^{(r-1)} x^{k-r}$$
$$R^{(r)}(x) = R^{(r-1)}(x) - R_{n-r}^{(r-1)} x^{k-r} g(x)$$

y las condiciones iniciales $Q^{(0)}(x) = 0$ y $R^{(0)}(x) = v(x)$. En el caso de la codificación de códigos cíclicos, basta con fijar $R^{(0)}(x) = x^{n-k}m(x)$ para, al cabo de k iteraciones, obtener el resto de la división entre $g(x)$. Observe que, en el circuito, son n las iteraciones necesarias, pues las $n-k$ primeras sirven únicamente para establecer la condición inicial $R^{(0)}(x) = x^{n-k}m(x)$.

Puesto que el circuito es lineal, puede obtenerse el mismo resultado final con el esquema de la figura 7.5, en el que los bits del mensaje $m(x)$ se introducen en el sumador del extremo derecho. En este caso, el contenido del registro de memoria en cada etapa son los coeficientes del producto de $g(x)$ por el cociente parcial de la división de $x^{n-k}m(x)$ entre $g(x)$. El sumador del extremo derecho actúa realmente como un comparador binario, calculando sucesivamente los coeficientes del polinomio cociente. Después de k iteraciones, el registro contendrá el resto final de la división, es decir, los $n-k$ dígitos de redundancia. Al igual que antes, para el caso particular de códigos binarios, $g_0 = 1$ y $-1 = 1$.

Este procedimiento se puede simbolizar como sigue. Sea $P^{(r)}(x)$ el producto de $g(x)$ por el resto parcial de la división de $v(x)$ entre $g(x)$; $P^{(r+1)}(x)$ es función de $P^{(r)}(x)$ conforme a las ecuaciones:

$$p_{k-r}^{(r)} = -\frac{v_{n-r} + p_{k-r}^{(r-1)}}{g_{n-k}}$$
$$P^{(r)}(x) = P^{(r-1)}(x) + p_{k-r}^{(r-1)} x^{k-r} g(x)$$

para $r = 1, \ldots, k$, con la condición de partida $P^{(0)}(x) = 0$ y siendo $p_j^{(r)}$ el coeficiente de x^j en $P^{(r)}(x)$. Observe que sólo es necesario memorizar k coeficientes de $P^{(r)}(x)$ dado que, por construcción, sus r primeros términos —los de las potencias más altas—, coiciden con los del dividendo, que ya se han procesado y no intervienen en las iteraciones posteriores. Después de k iteraciones, el resto de la división de $v(x)$ entre $g(x)$ es justamente

$$v(x) - P^{(k)}(x).$$

En el supuesto de aplicar este algoritmo a un código cíclico, el circuito de la figura 7.5 es el resultado de haberlo particularizado para $g_{n-k} = 1$ y $v(x) = x^{n-k}m(x)$.

Los circuitos divisores requieren menos memoria que los multiplicadores cuando $n-k < k$, es decir, cuando $R_c = \frac{k}{n} > \frac{1}{2}$. En la práctica, sólo en los

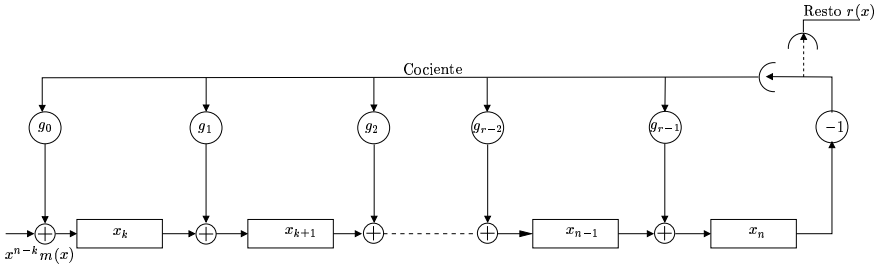


FIGURA 7.4. Circuito divisor de polinomios genérico.

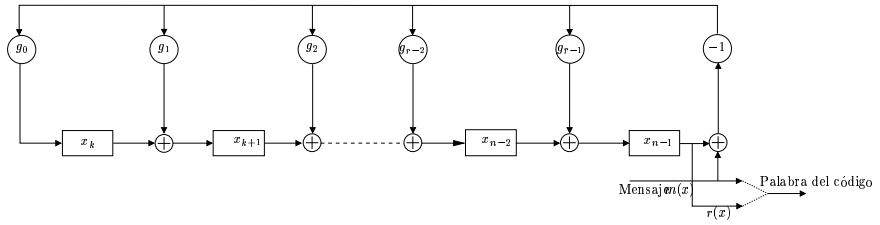


FIGURA 7.5. Circuito divisor polinómico equivalente.

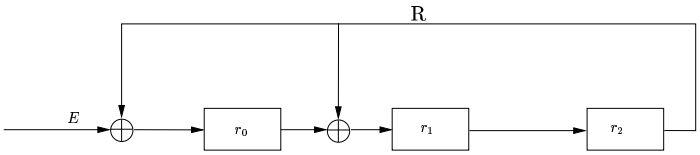


FIGURA 7.6. Circuito codificador del código cíclico sistemático generado por $g(x) = x^3 + x + 1$.

E	R	r_0	r_1	r_2
		0	0	0
1	0	1	0	0
1	0	1	1	0
1	0	1	1	1
1	1	0	0	1
0	1	1	1	0
0	0	0	1	1
0	1	1	1	1

Tabla 7.5. Operaciones del circuito divisor de la figura 7.6.

canales limitados severamente por una tasa de error inaceptable se necesita recurrir a códigos con tasas menores que $1/2$, por lo que, en la mayor parte de las aplicaciones, los cálculos de generación y verificación de los símbolos de redundancia se realizan con circuitos divisores.

Tanto los circuitos multiplicadores como los divisores son más simples que el circuito de generación de bits de paridad de un código de bloques sistemático que no sea cíclico, que precisa una memoria de k bits y $n-k$ funciones lógicas para generar todos los símbolos de redundancia.

Ejemplo 7.5. La figura 7.6 representa el circuito codificador por división del código cíclico sistemático $[7,4]$ engendrado por x^3+x+1. Las operaciones que lleva a cabo para calcular la redundancia del mensaje 1111 se ilustran en detalle en la tabla 7.5.

La primera columna contiene los bits entrantes al circuito, que son los del mensaje más tantos ceros como símbolos de redundancia; la segunda columna señala el valor del bit de la rama de realimentación en cada etapa; y la tercera, el valor almacenado en cada una de las celdas del registro de desplazamiento.

Es fácil verificar que

$$m(x)x^3 = x^6 + x^5 + x^4 + x^3 = (x^3 + x^2 + 1)(x^3 + x + 1) + x^2 + x + 1$$

y que el contenido de las celdas del registro de desplazamiento en cada etapa coincide con los restos parciales de esta división. ■

7.5. Detección de errores

La propiedad cíclica implica que las ecuaciones de comprobación de paridad, en las que se basa el funcionamiento del decodificador, equivalen al cálculo del resto de una división polinómica; esta división se puede llevar a cabo, por ejemplo, mediante un registro de desplazamiento realimentado como los vistos en el apartado anterior. En efecto, si tomamos H como la única matriz de la forma (7.3), entonces, para cualquier vector $\mathbf{y}$, el vector $\mathbf{y}H^T$ coincide con los coeficientes de grado $k, \ldots, n-1$ del polinomio

$$y(x)h(x) \mod x^n - 1.$$

Pero estos coeficientes son, precisamente, los del mismo grado en

$$h(x)\big(y(x) \mod g(x)\big)$$

y dependen únicamente del resto de la división de $y(x)$ entre $g(x)$.[4] Por tanto, existe una relación uno a uno entre los vectores $\mathbf{y}H^T$ y los polinomios de grado menor que $n-k$, en virtud de la cual podemos adoptar, para un código cíclico, la siguiente definición de síndrome.

Definición 7.3 (Síndrome). *Dado un código cíclico $[n,k]$ de polinomio generador $g(x)$, se define el síndrome de un polinomio $y(x)$ de grado menor que n como*

$$s(x) = y(x) \mod g(x).$$

Para la representación del vector error se va a utilizar también la notación polinómica. La siguiente definición es obvia.

Definición 7.4 (Polinomio error). *El polinomio error es la representación polinómica del vector error,*

$$e(x) = e_{n-1}x^{n-1} + \cdots + e_1 x + e_0.$$

Así pues, el polinomio $y(x)$ recibido cuando se transmite la palabra del código $\mathbf{v}$, $\mathbf{y} = \mathbf{v} + \mathbf{e}$, se puede escribir como $y(x) = v(x) + e(x)$. En estas condiciones, su síndrome

$$y(x) = v(x) + e(x) = e(x) \mod g(x)$$

es un polinomio de grado máximo $n-k-1$ que depende solamente del polinomio error $e(x)$. Como todas las palabras de un código cíclico contienen

[4] Si $F_2[x]$ es el conjunto de polinomios con coeficientes binarios, entonces dichos coeficientes dependen únicamente de la clase de equivalencia del conjunto cociente $F_2[x]/g(x)$ a la que pertenece $y(x)$.

como factor al polinomio generador, es inmediata la validez del siguiente teorema.

Teorema 7.5. *El polinomio $y(x)$ es una palabra del código si y solamente si su síndrome es nulo.*

Demostración. Del teorema 7.4 y la definición de síndrome

$$y(x) \in \mathcal{C} \Leftrightarrow y(x) = 0 \mod g(x) \Leftrightarrow s(x) = 0. \qquad \blacktriangleright$$

Se van a analizar de inmediato algunas circunstancias en que un cierto polinomio $y(x)$ es indivisible entre el generador de un código cíclico, condición suficiente para que aquél posea síndrome no nulo, y sea así posible detectar un error en la transmisión. A causa de su estructura algebraica y geométrica, se presentan para los códigos cíclicos situaciones que no se dan en otros códigos lineales.

Una primera diferencia simple es que todos los polinomios del código difieren en dos símbolos cuanto menos, es decir, que la distancia de un código cíclico no puede ser menor que 2.

Teorema 7.6. *Un código cíclico detecta todos los errores simples.*

Demostración. La única raíz de $e(x) = x^i$, $0 \leq i < n$, es $x = 0$. Por otra parte, $g(0) \neq 0$ por ser su término independiente no nulo. Por lo tanto $e(x)$ no puede ser un múltiplo de $g(x)$, y resulta que $e(x) \neq 0 \mod g(x)$. $\blacktriangleright$

Una segunda característica es que, seleccionando con propiedad el polinomio generador, resulta sencillo detectar errores múltiples.

Teorema 7.7. *Si el polinomio generador de un código cíclico binario es de la forma $g(x) = (x^h - 1) \cdot m(x)$, para algún $h > 0$, entonces todos los polinomios de error con un número impar de términos no nulos son detectables.*

Demostración. Si $e(x)$ tiene un número impar de términos no nulos, $e(1) = 1$. Por otra parte, $g(1) = (x^h - 1) \cdot m(x)|_{x=1} = 0$. Por lo tanto, $g(x)$ no puede ser factor de $e(x)$, y se cumple que $e(x) \mod g(x) \neq 0$.

Obsérvese que este teorema equivale a afirmar que si $g(x) = (x^h - 1) \cdot m(x)$, entonces todas las palabras del código son de peso Hamming par. $\blacktriangleright$

Cierta clase de polinomios, la de los polinomios primitivos, es útil en especial en la detección de los errores de dos posiciones. Definamos de forma precisa ambos conceptos, el de error doble y el de polinomio primitivo.

Grado	Polinomio primitivo	Grado	Polinomio primitivo
2	x^2+x+1	16	$x^{16}+x^{12}+x^3+x+1$
3	x^3+x+1	17	$x^{17}+x^3+1$
4	x^4+x+1	18	$x^{18}+x^7+1$
5	x^5+x^2+1	19	$x^{19}+x^5+x^2+x+1$
6	x^6+x+1	20	$x^{20}+x^3+1$
7	x^7+x^3+1	21	$x^{21}+x^2+1$
8	$x^8+x^4+x^3+x+1$	22	$x^{22}+x+1$
9	x^9+x^4+1	23	$x^{23}+x^5+1$
10	$x^{10}+x^3+1$	24	$x^{24}+x^7+x^2+x+1$
11	$x^{11}+x^2+1$	25	$x^{25}+x^3+1$
12	$x^{12}+x^6+x^4+x+1$	26	$x^{26}+x^6+x^2+x+1$
13	$x^{13}+x^4+x^3+x+1$	27	$x^{27}+x^5+x^2+x+1$
14	$x^{14}+x^{10}+x^6+x+1$	28	$x^{28}+x^3+1$
15	$x^{15}+x+1$		

Tabla 7.6. Algunos polinomios primitivos sobre GF(2).

Definición 7.5 (Polinomio primitivo). *Se dice que un polinomio de grado s con coeficientes binarios, $p_s(x)$, es un polinomio primitivo si cumple las siguientes condiciones:*[5]

a) es irreducible, es decir, no es divisible por ningún otro polinomio con coeficientes binarios salvo él mismo y el polinomio constante 1;

b) $\min\{m : p_s(x) \,|\, x^m - 1\} = 2^s - 1$.

Definición 7.6 (Error doble). *Un polinomio de error de la forma $e(x) = x^i + x^j$ se denomina error doble de distancia $|i-j|$.*

Teorema 7.8. *Si el polinomio generador es múltiplo de un polinomio primitivo de grado s, todos los errores dobles de distancia menor que $2^s - 1$ son detectables.*

Demostración. Si $e(x) = x^i + x^j = x^j(x^{i-j}+1)$, como ni x^j ni $x^{i-j}+1$ (cuando $i - j < 2^s - 1$) son, por definición, múltiplos de $g(x)$, $e(x)$ no es divisible por $g(x)$, y el error es detectable. ▶

Corolario 7.9. *Si $g(x)$ contiene como factor un polinomio primitivo de grado s tal que $n \leq 2^s - 1$, todos los errores dobles son detectables.*

[5]Estas condiciones son equivalentes a afirmar que las raíces del polinomio son elementos primitivos de GF(2^s), que es el menor cuerpo que las contiene.

Existen polinomios primitivos binarios de cualquier grado,[6] algunos de los cuales se listan en la tabla 7.6. Bastan para casi todas las aplicaciones prácticas, dado que la selección de un polinomio primitivo de grado bajo como (divisor del) polinomio generador de un código cíclico enseguida lleva a longitudes de código muy elevadas.

Una tercera consecuencia de la ciclicidad, sumamente importante, es que los códigos cíclicos aseguran la detección de errores que afectan a posiciones consecutivas en una palabra del código, una situación frecuente en los canales con memoria. Los patrones de error con esta característica se conocen como ráfagas de errores. A continuación se define de manera más precisa el concepto de ráfaga de errores.

Definición 7.7 (Ráfaga de errores). *Un polinomio error es una ráfaga de longitud s si es posible escribirlo como*

$$x^i(1 + e_1x + \cdots + e_{s-2}x^{s-2} + x^{s-1}) \mod x^n - 1, \quad e_j \in \mathrm{GF}(2)$$

y s es el menor entero con esta propiedad.

Obsérvese que la longitud de una ráfaga está, en realidad, determinada por la mínima distancia posible entre los unos de los extremos en cualquiera de los desplazamientos del vector error, de manera que, por ejemplo, el vector 010001 ($e(x) = 1 + x^4$) no es una ráfaga de longitud 5, sino una ráfaga de longitud 3, pues dos desplazamientos producen el vector 000101, es decir

$$e(x) = x^4(1 + x^2) \mod x^6 - 1.$$

El teorema siguiente establece parcialmente en qué condiciones es posible detectar ráfagas de error.

Teorema 7.10. *Un código cíclico $[n, k]$ detecta todas las ráfagas de error de longitud menor o igual que $n - k$.*

Demostración. Supongamos que

$$e(x) = x^i(x^{s-1} + e_1x^{s-2} + \cdots + e_{s-2}x + 1) \mod x^n - 1.$$

Su síndrome será

$$s(x) = x^i(x^{s-1} + e_1x^{s-2} + \cdots + e_{s-2}x + 1) \mod g(x),$$

y como x^i no es factor de $g(x)$, si $s \leq n - k$, $x^{s-1} + e_1x^{s-2} + \cdots + e_{s-2}x + 1$ no es divisible por $g(x)$, resulta que $s(x) \neq 0$. ▶

[6] En el teorema 9.22 se demuestra la existencia de polinomios primitivos de grado cualquiera.

Pero, además, la probabilidad de detectar ráfagas de error de longitud mayor es muy elevada, como ponen de manifiesto los siguientes dos teoremas.

TEOREMA 7.11. *En un código cíclico binario* $[n, k]$*, la probabilidad de que una ráfaga de error de longitud* $n - k + 1$ *no sea detectada es menor que* $(\frac{1}{2})^{n-k-1}$.

DEMOSTRACIÓN. En efecto, suponiendo que los símbolos del vector recibido son equiprobables (lo que para el receptor equivale al caso peor), hay un total de 2^{n-k-1} ráfagas distintas de longitud $n - k + 1$ y solamente una de ellas $(e(x) = g(x))$ es indetectable. ▶

TEOREMA 7.12. *En un código cíclico binario* $[n, k]$*, la probabilidad de que una ráfaga de error de longitud* $l > n - k + 1$ *sea indetectable es menor que* $2^{-(n-k)}$.

DEMOSTRACIÓN. Una ráfaga de longitud l, $e(x) = x^i e'(x)$, es indetectable solamente si $e'(x) = a(x)g(x)$, lo que puede suceder de $2^{l-2-(n-k)}$ formas diferentes. Pero hay en total 2^{l-2} ráfagas de longitud l; si éstas se suponen equiprobables, la probabilidad de una ráfaga no detectada es $2^{-(n-k)}$. ▶

EJEMPLO 7.6. El polinomio $g(x) = x^{16} + x^{12} + x^5 + 1$ es el producto de $x + 1$ y un polinomio primitivo de grado 15,

$$x^{16} + x^{12} + x^5 + 1 = (x + 1)(x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x + 1).$$

Al ser $x+1$ divisor de $x^{2^{15}-1}+1 = x^{32767}+1$, y primo relativo del polinomio primitivo, $g(x)$ divide a $x^{32767} + 1$. Luego es polinomio generador de un código cíclico binario $[32767, 32751]$. Dicho código goza de las siguientes propiedades de detección:

a) Detecta todos los errores simples (véase el teorema 7.6).

b) Detecta todos los errores impares, porque contiene un factor $x + 1$ (véase el teorema 7.7).

c) Detecta todos los errores dobles, pues su polinomio generador contiene un polinomio primitivo de grado 15, y $2^{15} - 1 \geq 32\,767$ (véase el corolario 7.9).

d) Detecta todas las ráfagas de errores de longitud menor o igual a 16 (véase el teorema 7.10).

e) Detecta las ráfagas de error de longitud 17 con probabilidad al menos $1 - \frac{1}{2^{15}} \approx 1 - 3{,}05 \cdot 10^{-5}$ (véase el teorema 7.11).

f) Detecta las ráfagas de error de longitud mayor que 17 con probabilidad mayor que $1 - \frac{1}{2^{16}} \approx 1 - 1{,}52 \cdot 10^{-5}$ (véase el teorema 7.12). ■

7.6. Decodificación de códigos cíclicos

7.6.1. Corrección de errores

Hasta el momento sólo se han explorado las consecuencias que se derivan de la propiedad cíclica con el propósito de simplificar el circuito de codificación, en tanto que se mantiene inalterado el procedimiento de decodificación: cálculo del síndrome (con el mismo circuito codificador, eso sí), asociación a un vector de error tabulado y posible corrección del error. Se podría pensar que la propiedad cíclica también implica relaciones algebraicas interesantes para el proceso de decodificación, que aprovechadas de manera conveniente simplificarían en parte la construcción de los circuitos decodificadores. Así ocurre, y la clave de esta idea descansa en la siguiente identidad entre polinomios.

Teorema 7.13 (Meggitt). *Sean $g(x)$, $h(x)$, $s(x)$ y $e(x)$ polinomios tales que*

a) $g(x)h(x) = x^n - 1$

b) $s(x) = e(x) \mod g(x)$.

Se verifica entonces que

$$[xe(x) \mod x^n - 1] \mod g(x) = xs(x) \mod g(x).$$

Demostración. Operando simbólicamente

$$\begin{aligned} xs(x) \mod g(x) &= x\,[e(x) \mod g(x)] \mod g(x) \\ &= xe(x) \mod g(x) \\ &= [xe(x) \mod x^n - 1] \mod g(x) \end{aligned}$$

en donde la última igualdad es una consecuencia simple de que $g(x)$ sea un divisor de $x^n - 1$. ▶

En particular, si $e(x)$ es un polinomio error con síndrome $s(x)$, el síndrome de su desplazamiento cíclico $xe(x) \mod x^n - 1$ será

$$s^{(1)}(x) = xs(x) \mod g(x)$$

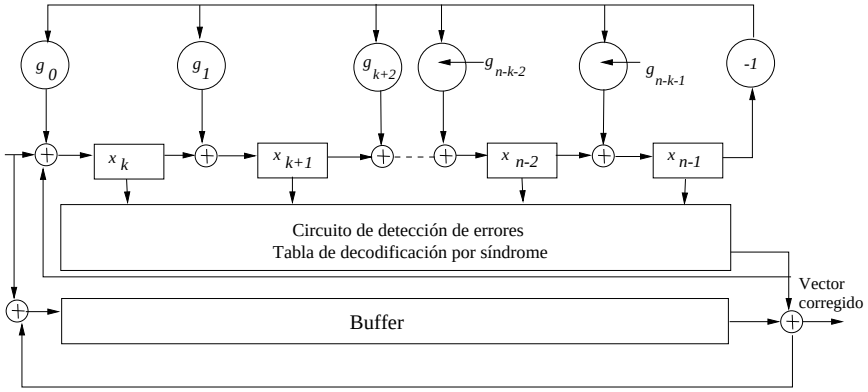


FIGURA 7.7. Esquema genérico de un decodificador con corrección de errores.

la reducción módulo $g(x)$ del síndrome $s(x)$ desplazado en una posición. Por inducción, el síndrome del i-ésimo desplazamiento cíclico de $e(x)$ será

$$s^{(i)}(x) = x^i s(x) \mod g(x)$$

la reducción módulo $g(x)$ del i-ésimo desplazamiento cíclico del síndrome original, considerando a éste como un vector de $n-k$ componentes.

Así pues, el teorema de Meggitt nos dice que no es preciso calcular previamente ni tampoco mantener en memoria los síndromes de los desplazamientos cíclicos de un vector de error del cual ya se conoce su síndrome; se pueden calcular iterando una o más veces el registro de desplazamiento realimentado del decodificador, partiendo del contenido inicial $s(x)$ (recordemos que el desplazamiento equivale a la multiplicación por x, y la realimentación, al cálculo del resto de la división entre $g(x)$).

Por lo tanto, para la tarea de corregir errores, es suficiente comparar los sucesivos desplazamientos del síndrome presente en el circuito decodificador con un subconjunto de $1/n$ de las entradas de la tabla de síndromes original. Los decodificadores con una arquitectura como la descrita se conocen como decodificadores Meggitt, y su esquema general se muestra en la figura 7.7.

La principal ventaja de un decodificador Meggitt estriba en que el receptor puede decodificar la secuencia recibida en serie, símbolo a símbolo, y corregir los errores (verificar si pertenecen a la tabla de síndromes) con el *mismo* circuito combinacional. Sea $s_r(x)$ el síndrome del vector recibido, $s_e(x)$ el de una secuencia de error corregible con un error en el primer símbolo ($e_{n-1} = 1$), y $z(x)$ el síndrome de x^{n-1}. Un decodificador Meggitt realiza este algoritmo:

1. Inicialización: $s^{(0)}(x) = s_r(x)$, $i = 0$; almacenar en un registro el vector recibido $(y_{n-1}, \ldots, y_0)$.

2. Iteración:

   a) si $s^{(i)}(x) \neq s_e(x)$, entonces el vector recibido no contiene error en el símbolo y_{n-1-i}; emitir y_{n-1-i} y desplazar cíclicamente el vector recibido y el contenido del registro de síndrome

   $$s^{(i+1)}(x) = xs^{(i)}(x) \mod g(x);$$

   b) si $s^{(i)}(x) = s_e(x)$, entonces el símbolo recibido y_{n-1-i} es erróneo; emitir $y_{n-1-i} + e_{n-1}$, sumar el síndrome de x^{n-1} al contenido del registro de síndrome y desplazarlo cíclicamente

   $$s^{(i+1)}(x) = x\big(s^{(i)}(x) + z(x)\big) \mod g(x);$$

   c) $i = i + 1$; repetir los pasos anteriores si $i < n$.

Ejemplo 7.7. Para concretar los pasos del algoritmo, consideremos el código Hamming binario $[15, 11]$ de polinomio generador $x^4 + x + 1$. Por ser un código perfecto corrector de errores simples, hay en su tabla de decodificación un solo vector de error, $e(x) = x^{14}$, con el bit más significativo a uno, cuyo síndrome vale $s(x) = x^3 + 1$. El circuito decodificador de Meggitt se ha representado en la figura 7.8 y opera como sigue. Tras haber introducido en el circuito los 15 símbolos de la secuencia observada a la salida del canal, el registro $s_0s_1s_2s_3$ contiene el síndrome del vector recibido. Si este síndrome es igual a 1001, entonces el decodificador seleccionará $e(x) = x^{14}$ y el error se habrá corregido; caso de no coincidir, si, tras un desplazamiento más, $s_0s_1s_2s_3$ es ahora igual a 1001, se habrá descubierto que el error era $e(x) = x^{13}$; y así se continuaría sucesivamente. El proceso de decodificación en serie con eliminación de errores simples termina, para este código, en el instante en que se corrige un error, y requiere en total, en el caso peor, 30 desplazamientos del registro de síndrome. A cambio de este retardo, vea que el circuito corrector de errores consta de una sola puerta lógica. ■

Si el código permite corregir más errores que los simples, entonces aparece una dificultad: cada vez que se corrige un bit se ha de introducir en el registro de desplazamiento el síndrome del vector corregido, que posiblemente conste de más de un coeficiente no nulo. Así pues, habría que acceder a cada uno de los biestables del registro de síndrome para escribir en ellos los nuevos coeficientes, complicando la circuitería del decodificador.

En lugar de la definición original, supongamos que el síndrome se calculase como

$$s'(x) = x^{n-k}y(x) \mod g(x),$$

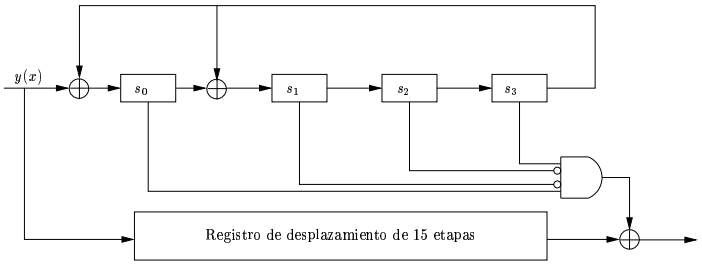


FIGURA 7.8. Decodificador Meggitt del código Hamming [15,11].

es decir, como el síndrome del desplazamiento cíclico de $n-k$ posiciones de $y(x)$. Esta nueva formulación no supone cambio alguno en la regla de decisión: existe una relación uno a uno entre los cogrupos del código y los síndromes, y puesto que x^{n-k} y $g(x)$ son primos relativos, las palabras del código siguen siendo las únicas con síndrome $s'(x)$ nulo.

Asociado a $e(x) = x^{n-1}$, un error en el bit situado más a la izquierda, se tendrá ahora el nuevo síndrome

$$s'(x) = x^{n-k}x^{n-1} = x^{2n-k-1} = x^{n-k-1} \mod g(x)$$

que consta de un único coeficiente, justo en el bit más significativo. El efecto del error $s'(x)$ puede, por consiguiente, eliminarse cambiando el bit más significativo del registro de desplazamiento del divisor.

En la figura 7.9 se ha modificado el circuito decodificador de la figura 7.8 para tener en cuenta este caso. Los bits del mensaje se introducen ahora por el extremo derecho del registro (una operación equivalente a la premultiplicación por x^{n-k}), que tras 15 desplazamientos contendrá el valor del síndrome modificado $s'(x) = x^4 y(x) \mod g(x)$. El síndrome correspondiente ahora al error x^{14} es el polinomio x^3. Cuando se detecta, el bit de realimentación entre la salida del detector y el registro de desplazamiento provoca que, en la siguiente iteración, el registro contenga ceros, señalando con ello el final de la decodificación.

7.6.2. Captura de errores

En la práctica, con códigos de gran longitud, el circuito de corrección de errores puede volverse en exceso complicado. Sin embargo, si nos limitamos a corregir sólo ciertos vectores de error, es posible simplificarlo y construir decodificadores Meggitt sencillos y eficientes.

Supongamos un código lineal sistemático con matriz de comprobación

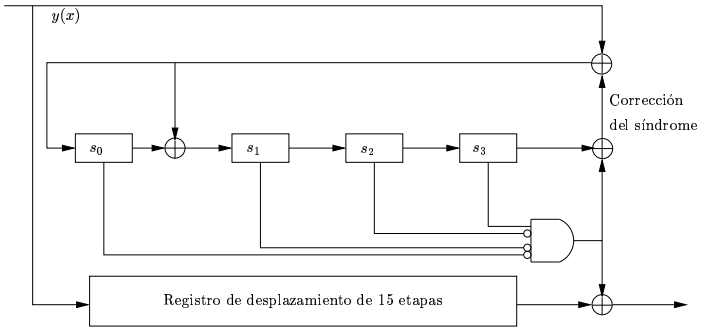


Figura 7.9. Decodificador Meggitt con corrección de errores simples para un código Hamming [15,11].

de paridad

$$H_{n-k\times n} = \begin{pmatrix} -A^T & I_{n-k} \end{pmatrix}$$

e imaginemos que al transmitir una de las palabras del código sólo ocurren errores en

$$t \leq \left\lfloor \frac{d_{\mathcal{C}} - 1}{2} \right\rfloor = t^*$$

símbolos, siendo $d_{\mathcal{C}}$ la distancia del código.

Teorema 7.14. *Si el vector error en el canal,* $\mathbf{e}$*, tiene peso* t*, y si el síndrome del vector recibido,* $\mathbf{s} = \mathbf{e}H^T$*, es de peso Hamming menor o igual que* t*, entonces los errores están confinados en las posiciones de los símbolos de redundancia y vienen dados por el vector* $\mathbf{s}$*. Si el peso Hamming del vector síndrome es mayor que* t^**, entonces al menos un símbolo de información es incorrecto.*

Demostración.

a) Si el vector error $\mathbf{e} = (e_{n-1}, \ldots, e_0)$ sólo contiene t errores en la parte de redundancia ($e_i \neq 0 \Rightarrow i < n-k$), entonces su síndrome

$$\mathbf{e}H^T = \mathbf{e}(-A^T\ I)^T = (e_{n-k-1}, \ldots, e_0)^T$$

tiene peso Hamming t, y $\mathbf{e}$ es el vector de peso mínimo asociado a ese síndrome.

b) Si al menos uno de los símbolos de información de $\mathbf{e}$ es erróneo, la palabra del código

$$\begin{aligned}(e_{n-1}, \ldots, e_{n-k})(I\ A) &= (e_{n-1}, \ldots, e_{n-k}, (e_{n-1}, \ldots, e_{n-k})A) \\ &= (\mathbf{e}_1, \mathbf{e}_2)\end{aligned}$$

es no nula y tiene peso $p_H(\mathbf{e}_1) + p_H(\mathbf{e}_2) \geq 2t^* + 1$. Pero como

$$\begin{aligned}\mathbf{s} = \mathbf{e}H^T &= [(e_{n-1}, \ldots, e_{n-k})A + (e_{n-k-1}, \ldots, e_0)]^T \\ &= \mathbf{e}_2^T + (e_{n-k-1}, \ldots, e_0)^T\end{aligned}$$

resulta que

$$\begin{aligned}p_H(\mathbf{s}) &\geq p_H\big((e_{n-1}, \ldots, e_{n-k})A\big) - p_H\big((e_{n-k-1}, \ldots, e_0)\big) \\ &\geq 2t^* + 1 - p_H(\mathbf{e}_1) - p_H(\mathbf{e}_2) \\ &= 2t^* + 1 - p_H(\mathbf{e}) \\ &\geq t^* + 1.\end{aligned}$$

Esto es, todos los vectores de error que señalan que algún símbolo de información es incorrecto están asociados a síndromes de peso mayor que t^*. ▶

En otras palabras, cuando en un código lineal sistemático los errores en el canal afectan a t^* (o menos) símbolos de redundancia, el síndrome del vector recibido da directamente los símbolos erróneos, y el vector transmitido es

$$\mathbf{x} = \mathbf{y} + (0, \ldots, 0, \mathbf{s}).$$

Por tanto, en códigos sistemáticos, los errores corregibles que afectan únicamente a la parte de redundancia son fácilmente identificables: basta con verificar si el peso de su síndrome es menor o igual a t^*. Esa detección del peso del síndrome se puede realizar con un circuito combinacional muy sencillo. Y, además, la corrección del error es trivial, porque es suficiente con sumar el contenido del registro del síndrome al vector recibido.

Consideremos ahora la colección de automorfismos del código. Estos automorfismos son simplemente todas las permutaciones de coordenadas que convierten una palabra del código en otra palabra del código. De la misma manera, los vectores de error corregibles se transforman en otros vectores corregibles por la acción de cualquiera de estas permutaciones. Como acabamos de ver, algunos patrones de error son particularmente fáciles de descubrir y de corregir, lo que significa que para ellos el circuito de decodificación es más sencillo. Podemos, por lo tanto, abordar la decodificación de un código cíclico sin necesidad de un circuito lógico complejo aplicando la siguiente idea. Elijamos un subconjunto de automorfismos del código capaces de transformar cualquier error corregible en uno que el decodificador pueda reconocer con facilidad. Apliquemos al vector recibido cada una de estas permutaciones hasta que se convierta en un vector que el circuito de decodificación pueda detectar y, cuando esto ocurra, procedamos a corregir el error. Por último, invirtamos la permutación aplicada para obtener el vector transmitido.

La aplicación general de este razonamiento da el siguiente algoritmo formal de decodificación (debido a F. J. MacWilliams) para corregir en un código lineal todos los vectores de error de peso menor o igual a t con $t \leq \lfloor (d_{\mathcal{C}} - 1)/2 \rfloor$.

Sea $\mathcal{P}$ una familia de permutaciones de las posiciones $\{0, 1, \ldots, n-1\}$ de los símbolos en las palabras del código con las siguientes propiedades:

a) $\sigma_j \in \mathcal{P}$ preserva el conjunto de palabras del código, es decir,

$$\mathbf{x} \in \mathcal{C} \Rightarrow \sigma_j(\mathbf{x}) = (x_{\sigma_j(n-1)} \ldots x_{\sigma_j(0)}) \in \mathcal{C}$$

σ_j pertenece al grupo de automorfismos de permutación de $\mathcal{C}$.

b) Para cualquier vector de error con peso Hamming menor o igual que t, al menos una permutación $\sigma_j \in \mathcal{P}$ traslada todos los símbolos erróneos a las últimas $n - k$ posiciones.

Algoritmo de decodificación por permutación

---

1. Si se recibe el vector $\mathbf{y}$, para cada permutación $\sigma_j \in \mathcal{P}$ con las propiedades citadas, calcular los vectores $\sigma_j(\mathbf{y})$ y su síndrome hasta obtener un síndrome $\mathbf{s}_j = (e_{n-k-1}, \ldots, e_0)$ con peso Hamming menor o igual a t.
2. Decodificar el vector recibido como
$$\sigma_j^{-1}\big(\sigma_j(\mathbf{y}) + (0, \ldots, 0, e_{n-k-1}, \ldots, e_0)\big).$$
3. Cuando, para toda permutación de $\mathcal{P}$, el peso Hamming de $\mathbf{s}_j$ es mayor que t, han ocurrido más de t errores y éstos no son corregibles.

Es claro que la posibilidad de aplicar con eficacia este algoritmo depende de haber obtenido un conjunto mínimo de permutaciones con las propiedades indicadas, algo que para un código lineal arbitrario constituye un problema nada trivial.

Ejemplo 7.8.

a) Sea σ la permutación cíclica, la definida por la rotación cíclica de los símbolos de un vector código, e I la permutación identidad. El conjunto de permutaciones $\{I, \sigma^2, \sigma^4\}$ traslada cualquier error simple del código Hamming $[7, 4]$ hacia la parte de redundancia.

b) Considérese el código Golay $(23, 12)$. La permutación π_2 definida por $i\pi_2 = 2i \mod 23$ para $0 \leq i \leq 22$ es un automorfismo del código cuando se aplica a los índices de las coordenadas. El conjunto de 92 permutaciones $\{\sigma^i\pi_2^j, 0 \leq i \leq 22, j = 0, 1, 2, 11\}$ traslada cualquier patrón de error de peso menor o igual que 3 a la parte de redundancia, y se podría utilizar para el algoritmo de decodificación por permutación con cualquiera de las representaciones cíclicas del código.

c) El código cíclico $[31, 21, 5]$ generado por $(x^5 + x^2 + 1)(x^5 + x^4 + x^3 + x^2 + 1)$ tiene a $\pi_2 : i \to 2i \mod 31$ como uno de sus automorfismos, con la particularidad de que $\pi_2^5 = I$. El conjunto de 155 automorfismos $\{\sigma^i\pi_2^j, 0 \leq i \leq 30, 0 \leq j \leq 4\}$ desplazan todos los errores dobles a la parte de redundancia, por lo que podrían utilizarse en un algoritmo de decodificación por permutación. ■

Pero, en el caso de un código cíclico, un conjunto (no siempre mínimo) de permutaciones que satisface las condiciones anteriores es el formado por todas las rotaciones cíclicas de los símbolos, lo que nos da entonces el siguiente algoritmo para corregir *todas* las ráfagas de error de longitud menor o igual que $n - k$ y peso no mayor que $\lfloor (d_{\mathcal{C}} - 1)/2 \rfloor$:

1. Calcular el síndrome del vector recibido $y(x)$.
2. Iterar i veces el registro de síndrome hasta que su contenido tenga peso Hamming menor o igual que t^*.
3. Sumar el contenido del registro de síndrome al i-ésimo desplazamiento cíclico del vector recibido, y desplazar la suma cíclicamente i veces hacia la derecha; el resultado es la palabra del código emitida.
4. Si, tras n desplazamientos del registro de síndrome, no se ha conseguido ninguno con peso Hamming menor o igual que t^*, entonces han ocurrido más de t^* errores.

Este método de iterar el registro de síndromes hasta que su contenido coincida con un vector de error corregible se denomina *método de captura de errores*. En particular, garantiza que se corregirán todos los errores de peso menor o igual que

$$\left\lfloor \frac{n-1}{k} \right\rfloor$$

que es un número pequeño si el código tiene una tasa elevada. Por ejemplo, para un código Hamming binario $[7, 4]$, indica que se podrán corregir los errores simples ($\lfloor 6/4 \rfloor = 1$), pero en un código cíclico $[31, 21]$, que corrige además errores dobles, el método de captura de errores sólo corrige todos

los de peso $\lfloor 30/21 \rfloor = 1$, dejando algunos dobles sin corregir. Así pues, el interés práctico de la decodificación por permutación o captura de errores se limita a unos pocos códigos pequeños.

Los decodificadores Hamming de las figuras 7.8 y 7.9 son efectivamente decodificadores de captura de errores de 1 bit.

7.7. Estructura cíclica de los códigos Hamming*

Un código Hamming binario de longitud $2^m - 1$ con $m \geq 3$ es equivalente a un código cíclico generado por un polinomio primitivo $p_m(x)$ de grado m. Para probar esta aseveración no hay más que relacionar propiedades que ya resultan conocidas.[7]

Teorema 7.15. *El menor código cíclico binario con un polinomio binario primitivo de grado m como generador es el código Hamming $\mathcal{H}_m$.*

Demostración. [1] Sea $p_m(x)$ un polinomio binario primitivo de grado m, que existe siempre. Por la propia definición de polinomio primitivo, es claro que el menor código cíclico engendrado por $p_m(x)$ tiene parámetros $[2^m - 1, 2^m - m - 1]$ y distancia mayor o igual que 3. Ahora bien, cualquier código $[2^m - 1, 2^m - m - 1]$ corrector de errores simples es, salvo equivalencia, único, y equivalente por tanto al código Hamming $\mathcal{H}_m$. ▶

Cabe una prueba alternativa igualmente directa empleando el lenguaje de los cuerpos finitos.[8]

Demostración. [2] Sea $\alpha \in \mathrm{GF}(2^m)$ un elemento primitivo y

$$p_m(x) = \prod_{i=0}^{m-1} (x - \alpha^{2^i})$$

un polinomio primitivo de grado m.

El polinomio $v(x) = \sum_{i=0}^{2^m-1} v_i x^i$ pertenece al código cíclico engendrado por $p_m(x)$ solamente si admite a $\alpha, \ldots, \alpha^{2^{m-1}}$ como raíces, es decir, si

$$v(\alpha^{2^k}) = \sum_{j=0}^{2^m-1} v_j \alpha^{j 2^k} = 0, \quad \forall k = 0, \ldots, m-1.$$

[7] Obviamos el caso $m = 2$ porque se trata de un simple código de repetición de longitud 3.

[8] Omita la lectura de la segunda demostración si no piensa estudiar los conceptos presentados en el capítulo 9.

Pero como $v_j \in \mathrm{GF}(2)$, $v(\alpha^{2^k}) = v(\alpha)^{2^k}$, y lo anterior se cumple si $v(\alpha) = 0$. O bien si

$$(v_{2^m-1}, \ldots, v_0) \begin{pmatrix} \alpha^{2^m-1} & \ldots & \alpha & 1 \end{pmatrix}^T = 0$$

lo que quiere decir que

$$H = \begin{pmatrix} \alpha^{2^m-1} & \ldots & \alpha & 1 \end{pmatrix}$$

es matriz de comprobación de paridad del código. H se puede convertir en una matriz binaria sustituyendo cada elemento de $\mathrm{GF}(2^m)$ por el correspondiente vector columna de m símbolos binarios que lo representa, y se tiene así que las columnas de H son todos los posibles vectores binarios no nulos de longitud m. ▶

Para hacer esta equivalencia más explícita, vamos a deducir de nuevo la ciclicidad de los códigos Hamming analizando la estructura de la matriz comprobadora de paridad deducida de la matriz generadora sistemática del código $G = [I_{2^m-m-1} \mid A_{2^m-m-1 \times m}]$.

Al dividir el polinomio x^{m+i} por el polinomio generador primitivo $p_m(x)$, para $0 \leq i < 2^m - m - 1$, se obtiene

$$x^{m+i} = a_i(x)p_m(x) + r_i(x)$$

en donde el resto $r_i(x)$ es un polinomio de grado menor que m cuyos coeficientes son una de las filas en la submatriz A. Como x no es factor del polinomio primitivo $p_m(x)$, $p_m(x)$ no divide a x^{m+i}, y en consecuencia $r_i(x) \neq 0$. Pero además de ser distinto de cero, $r_i(x)$ contiene al menos dos términos no nulos. De no ser así, si $r_i(x)$ fuese de la forma x^j con $0 \leq j < m$, se podría escribir

$$x^{m+i} = a_i(x)p_m(x) + x^j$$

o bien que

$$x^j(x^{m+i-j} - 1) = a_i(x)p_m(x).$$

No obstante, esta ecuación implica que $p_m(x)$ divide a $x^{m+i-j} - 1$, lo que no es posible porque $p_m(x)$ es primitivo de grado m, y $m + i - j < 2^m - 1$. También se deduce que dos filas cualesquiera de A son distintas entre sí, porque si $i \neq j$, los restos $r_i(x)$ y $r_j(x)$ son diferentes. Por contradicción, supuesto que $r_i(x) = r_j(x)$, la resta de las relaciones

$$\begin{aligned} r_i(x) + x^{m+i} &= a_i(x)p_m(x) \\ r_j(x) + x^{m+j} &= a_j(x)p_m(x) \end{aligned}$$

daría como resultado, suponiendo que $i < j$,

$$x^{m+i}(x^{j-i} - 1) = \big(a_i(x) - a_j(x)\big)p_m(x)$$

que, de nuevo, debe ser rechazado porque, como $j-i<2^m-1$, $x^{j-i}-1$ no puede contener como factor a $p_m(x)$.

Resumiendo todo lo anterior: las 2^m-m-1 filas de la submatriz A son todas distintas entre sí y contienen no menos de dos elementos distintos de cero. Así que las columnas de $H=[-A^T\,|\,I_m]$ son los 2^m-1 vectores binarios no nulos de longitud m, y H es matriz comprobadora de paridad de un código Hamming.

7.7.1. Códigos símplex

Consideremos ahora la estructura del código dual del código Hamming engendrado por $p_m(x)$. Este código dual es un código cíclico de parámetros $[2^m-1,m]$, y su polinomio generador es, según resulta ya conocido (véase el apartado 7.2), el polinomio recíproco de $p_m(x)$, esto es, el polinomio de grado 2^m-m-1

$$g_d(x)=-\frac{x^{2^m-1}-1}{x^m p_m(x^{-1})}.$$

El denominador, $x^m p_m(x^{-1})$, resulta ser un polinomio mónico de grado m, irreducible (por serlo $p_m(x)$), y tal que x^n-1 es un múltiplo suyo si $n=2^m-1$, pero no cuando $n<2^m-1$. Luego $q_m(x)=x^m p_m(x^{-1})$ es también un polinomio primitivo de grado m.

Pues bien, ocurre que al código dual así construido pertenecen únicamente el vector nulo y los 2^m-1 desplazamientos cíclicos del polinomio generador $g_d(x)$. El método más directo para probar esta afirmación es confirmar que todos los desplazamientos cíclicos de $g_d(x)$ son distintos entre sí; de no cumplirse esto, habría al menos dos desplazamientos cíclicos, $g_d^{(i)}(x)$ y $g_d^{(j)}(x)$, de i y j posiciones, respectivamente, que darían lugar al mismo vector código

$$\begin{aligned}x^i g_d(x)&=a_i(x)(x^{2^m-1}-1)+g_d^{(i)}(x)\\ x^j g_d(x)&=a_j(x)(x^{2^m-1}-1)+g_d^{(j)}(x).\end{aligned}$$

Y si, de acuerdo con la hipótesis, $g_d^{(i)}(x)=g_d^{(j)}(x)$, la resta de las ecuaciones anteriores permite escribir

$$\begin{aligned}x^i(x^{j-i}-1)g_d(x)&=\big(a_i(x)-a_j(x)\big)(x^{2^m-1}-1)\\ &=\big(a_i(x)-a_j(x)\big)g_d(x)q_m(x)\end{aligned}$$

o, lo que es lo mismo,

$$x^i(x^{j-i}-1)=\big(a_i(x)-a_j(x)\big)q_m(x).$$

Pero, por ser $q_m(x)$ primitivo, ninguno de los factores x^i o $x^{j-i}-1$ ($j-i<2^m-1$) puede ser múltiplo suyo, de donde se tiene que $g_d^{(i)}(x) \neq g_d^{(j)}(x)$ $\forall i \neq j, 0 \leq i,j < 2^m-1$, como se quería probar.

Una matriz generadora del código dual $[2^m-1, m]$ es la propia matriz comprobadora de paridad del código Hamming cíclico en cuestión, que podemos escribir por bloques como

$$H_{m\times 2^m-1} = \begin{pmatrix} I_m & A_{m\times 2^m-m-1} \end{pmatrix}.$$

Las columnas de la submatriz A son todos los vectores binarios de longitud m con al menos dos unos. En cada una de las filas de H hay 2^{m-1} unos y $2^{m-1}-1$ ceros, y puesto que las filas son palabras del código, se puede asegurar que todas las palabras del código no nulas del dual constan de 2^m-1 unos, por ser todas ellas desplazamientos cíclicos de una cualquiera.

Por lo tanto, si $\mathcal{C}(n,k)$ es el código dual de un código Hamming con $m \geq 3$ bits de redundancia, se ha visto que

$$\begin{aligned} n &= 2^m-1 \\ k &= m \\ d_{\mathcal{C}} &= 2^{m-1} \end{aligned}$$

y también que $\mathcal{C}[n,k]$ es un código cíclico, y que sus palabras son el vector nulo y todos los desplazamientos cíclicos del polinomio generador. Además, como

$$d_H(\mathbf{x},\mathbf{y}) = p_H(\mathbf{x}+\mathbf{y}) = 2^{m-1}$$

la distancia entre dos palabras del código cualesquiera es un invariante, hecho que motiva que a los códigos duales de un Hamming se los denomine *códigos símplex*. La de los símplex es, por tanto, una de las familias de códigos que satisfacen con igualdad la cota de Griesmer: en efecto,

$$\sum_{i=0}^{m-1} \left\lceil \frac{d_{\mathcal{C}}}{2^i} \right\rceil = \sum_{i=0}^{m-1} 2^i = 2^m-1 = n.$$

En general, si $f(x)$ es un divisor irreducible de x^n-1, el código cíclico generado por $(x^n-1)/f(x)$ es *irreducible* o *mínimo* (no contiene ningún subcódigo cíclico no trivial). Los códigos símplex son irreducibles.

Ejemplo 7.9. Para ilustrar un código símplex sencillo, considérese el código Hamming $[3,1]$ binario, que es también un elemental código de repetición de tres bits:

$$\mathcal{C}[3,1] = \{000, 111\}.$$

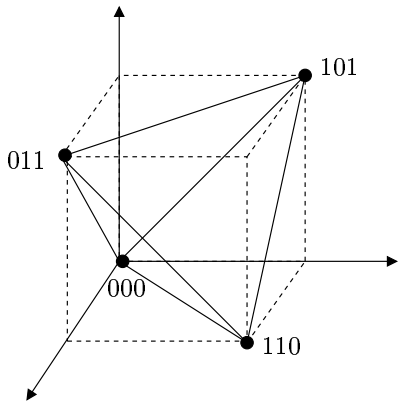


Figura 7.10. Código símplex dual del Hamming[3, 1].

Su código dual $[3, 2]$ tiene por palabras del código a

$$\mathcal{C}^{\perp} = \{000, 101, 011, 110\}$$

que son todos los desplazamientos cíclicos de $x + 1$ más el vector nulo. La representación gráfica en ejes coordenados de estas palabras del código son los vértices de un tetraedro regular (figura 7.10). En $n > 3$ dimensiones, la figura geométrica resultante sería un símplex, un conjunto de $n+1$ puntos de un hipercubo n-dimensional situados a la misma distancia (euclídea, además de Hamming) unos de otros. De ahí el nombre de esta clase de códigos. ■

Puesto que todos los vectores del código no nulos tienen el mismo peso Hamming, resulta sencillo escribir el polinomio enumerador de pesos del dual:

$$B(x) = 1 + (2^m - 1)x^{2^{m-1}}.$$

Y utilizando la identidad de MacWilliams (véase el apartado 6.10), obtenemos el polinomio enumerador del código Hamming original de longitud $n = 2^m - 1$:

$$\begin{aligned} A(x) &= 2^{-(n-k)}(1+x)^n B\left(\frac{1-x}{1+x}\right) \\ &= 2^{-(n-k)}(1+x)^n \left(1 + n\left(\frac{1-x}{1+x}\right)^{\frac{n+1}{2}}\right) \\ &= 2^{-(n-k)}\left((1+x)^n + n(1-x^2)^{\frac{n-1}{2}}(1-x)\right). \end{aligned} \tag{7.6}$$

EJEMPLO 7.10. Tomemos $m = 3$. El polinomio enumerador de pesos del código dual del Hamming $[7, 4]$ vale

$$B(x) = 1 + 7x^4$$

pues, como se ha probado, todas las palabras del código no nulas son de peso 4. Sustituyendo $n = 2^3 - 1 = 7$ y $n - k = 3$ en la fórmula (7.6), el polinomio enumerador de pesos del código Hamming es

$$A(x) = \frac{1}{8}\left((1-x)^7 + 7(1-x^2)^3(1-x)\right).$$

Desarrollando esta expresión y cancelando términos comunes, se deduce que

$$A(x) = 1 + 7x^3 + 7x^4 + x^7$$

lo que confirma una vez más que el código Hamming $[7, 4]$ consta del vector nulo, 7 vectores de peso 3, 7 vectores de peso 4 y un vector de peso 7. ■

7.8. Códigos cíclicos acortados

A menudo ocurre que no es posible encontrar un código cíclico con una longitud o número de bits de información por bloque que se ajusten a las condiciones en las que opera un sistema de transmisión de datos dado. Por fortuna, es sencillo modificar un código existente para adaptarlo a una longitud determinada.

En un código cíclico sistemático $\mathcal{C}[n, k]$, considérense los vectores del código cuyos primeros $l < k$ bits de información son nulos, correspondientes a mensajes con sus l primeros bits a cero. Existen en total 2^{k-l} vectores de esta forma, y es obvio que constituyen un subcódigo lineal de $\mathcal{C}$. Si se eliminan los l primeros bits de estas palabras del código, que valen cero y no aportan información alguna, se obtiene un conjunto de 2^{k-l} vectores que forman un código lineal binario $[n-l, k-l]$. Este código se denomina *código acortado*, y no es en general un código cíclico. De todas formas, pese a no ser cíclico, el código acortado posee, al menos, la misma capacidad detectora y correctora de errores que el código del que se deduce, porque sus secuencias código extendidas con l ceros son un subconjunto de las de aquél.

Las operaciones de codificación y cálculo del síndrome en un código acortado pueden llevarse a cabo con los mismos circuitos que se emplean con el código original, ya que los l ceros iniciales de los que se prescinde no afectan al resultado de ninguno de esos dos procesos de cálculo. Por el contrario, la decodificación con corrección o con captura de errores sí precisa de modificaciones en los circuitos que tengan en cuenta ahora la no ciclicidad de las palabras del código. No es difícil desarrollar los circuitos apropiados, pero no se describirán aquí esas modificaciones.

7.9. Códigos cíclicos irreducibles*

Un código cíclico contiene a otro cuando el polinomio generador del de mayor dimensión divide al del menor dimensión.

Teorema 7.16. *Sea $\mathcal{C}_1$ un código cíclico de longitud n y polinomio generador $g_1(x)$, y sea $\mathcal{C}_2$ un código cíclico de longitud n y polinomio generador $g_2(x)$;*

$$\mathcal{C}_1 \subseteq \mathcal{C}_2 \Leftrightarrow g_2(x) \mid g_1(x).$$

Demostración.

$$v(x) \in \mathcal{C}_1 \Leftrightarrow g_1(x) \mid v(x) \Rightarrow g_2(x) \mid v(x) \Leftrightarrow v(x) \in \mathcal{C}_2. \qquad \blacktriangleright$$

Diremos que un código cíclico es *irreducible* o mínimo si no contiene a ningún otro código cíclico (salvo el vector $\mathbf{0}$). A la vista del teorema anterior, los códigos irreducibles son fáciles de caracterizar.

Teorema 7.17. *Sea $F_q[x]$ el conjunto de todos los polinomios con coeficientes tomados de un cuerpo de q elementos. Sea $x^n - 1 = f_1(x)f_2(x)\cdots f_s(x)$ la descomposición en factores irreducibles en $F_q[x]$.*

a) El código cíclico de polinomio generador $x^n - 1/f_i(x)$ es irreducible.

b) $F_q[x]/x^n - 1$ es el espacio vectorial suma directa de los códigos cíclicos irreducibles.

Demostración.

a) El único polinomio al que $x^n - 1/f_i(x)$ divide es $x^n - 1$.

b) Los códigos irreducibles son disjuntos dos a dos. Además, puesto que $f_1(x), \ldots, f_s(x)$ son irreducibles, por el algoritmo de división euclídeo

$$1 = \sum_{i=1}^{s} a_i(x) f_i(x)$$

de donde se concluye que

$$\langle x^n - 1/f_1(x)\rangle + \langle x^n - 1/f_2(x)\rangle + \cdots + \langle x^n - 1/f_s(x)\rangle = \mathcal{R}_n$$

en donde $\langle g(x)\rangle$ representa el código cíclico generado por $g(x)$. Esto es, cualquier vector de $\mathcal{R}_n$ se puede escribir de manera única como una suma de vectores de sus códigos cíclicos irreducibles. $\blacktriangleright$

Una consecuencia útil de este teorema es la siguiente. Llamemos código complementario de un código cíclico al código formado por la suma de todos los códigos irreducibles que no están contenidos en $\mathcal{C}$. El código complementario satisface las condiciones

$$\mathcal{C} + \mathcal{C}' = \mathcal{L}_n$$
$$\mathcal{C} \cap \mathcal{C}' = \{\mathbf{0}\}.$$

Si $x^n - 1 = f_1(x) \cdots f_s(x) f_{s+1}(x) \cdots f_r(x)$ es la descomposición de $x^n - 1$ en factores irreducibles sobre $F_q[x]$, y el polinomio generador de $\mathcal{C}$ es $g(x) = f_1(x) \cdots f_s(x)$, es claro que

$$\mathcal{C}' = \sum_{j=s+1}^{r} \langle x^n - 1/f_j(x) \rangle = \langle (x^n - 1)/(f_{s+1}(x) \cdots f_r(x)) \rangle \,.$$

Por tanto, el código complementario tiene por polinomio generador a $(x^n - 1)/h(x)$.

Notas bibliográficas

La teoría algebraica de los códigos cíclicos se desarrolló desde finales de los años cincuenta, cuando comenzaron a investigarse los primeros códigos polinómicos de la clase BCH como extensiones de los Hamming binarios [10]. Muchas de las contribuciones iniciales pertenecen a Asmuss *et al.* [2]. Además de una presentación exhaustiva de los aspectos básicos, Pless y Huffman [53, cap. 1 y cap. 11] ofrecen una discusión sobre algunas cuestiones teóricas irresueltas. El problema de la obtención de la distancia de un código cíclico, en particular, ha originado numerosas publicaciones, y en [68] se presenta una manera unificada de abordar la deducción de las cotas más conocidas. El método de decodificación por permutación es original de MacWilliams [41]. Blahut [7, 8] y Lin y Costello [39] contienen amplio material sobre los circuitos de codificación y decodificación de los códigos cíclicos. Blahut [7, 8] también dedica amplio espacio a la caracterización espectral de los códigos cíclicos, un enfoque que inicialmente formularon G. Solomon y H. Mattson [44].

7.A. Códigos cíclicos, anillos e ideales

Los códigos cíclicos poseen una estructura algebraica simple y elegante, a la que se llega sin más que analizar las propiedades de la suma y el producto de polinomios (módulo $x^n - 1$) definidos sobre un cuerpo finito.

Sea $F_q[x]$ el conjunto de polinomios en la variable x con coeficientes del cuerpo $\mathrm{GF}(q)$. $F_q[x]$ es un conjunto que, con las operaciones usuales de suma y multiplicación de polinomios, cumple las siguientes propiedades:

a) $(F_q[x], +)$ es un grupo conmutativo; esto es, la suma de polinomios es asociativa, conmutativa, posee elemento neutro y cualquier polinomio tiene simétrico.

b) El producto de polinomios es asociativo y conmutativo.

c) El producto de polinomios es distributivo respecto de la suma.

Considérense sólo estas propiedades y hágase abstracción de la naturaleza concreta de los elementos del conjunto soporte.

Definición 7.8 (Anillo). *Sea $\mathcal{R}$ un conjunto de elementos y sean $(+, \cdot)$, suma y producto, dos operaciones internas definidas sobre pares de elementos de $\mathcal{R}$. Si*

a) $(\mathcal{R}, +)$ es un grupo conmutativo
b) El producto es asociativo
c) El producto es distributivo respecto de la suma

entonces la estructura algebraica $(\mathcal{R}, +, \cdot)$ se denomina anillo.

Cuando para el producto existe elemento unidad, un anillo se llama *unitario*; cuando el producto es una operación conmutativa, se dice que el anillo es *conmutativo*. $F_q[x]$ es un anillo unitario y conmutativo. En general, si F es un cuerpo, el conjunto $F[x]$ de polinomios con coeficientes de F junto con las operaciones de suma y producto entre polinomios es un anillo unitario, que será conmutativo cuando lo sea F.

Véase que los axiomas que definen un anillo son menos restrictivos que los que definen un cuerpo nada más que en lo referente al producto, pues no se exige que un elemento posea inverso. Pueden existir en un anillo unitario, no obstante, ciertos elementos para los que sí se disponga de inverso, los cuales se designan entonces como elementos invertibles: en el caso de $F_q[x]$ los únicos elementos invertibles son los polinomios constantes distintos de cero. Los elementos invertibles de un anillo unitario forman grupo para la

operación producto. Si dos elementos a y b no nulos son tales que $ab = 0$, se llaman *divisores de cero*. Por ejemplo, el conjunto de las matrices reales de orden n es un anillo con divisores de cero para las operaciones ordinarias de suma y producto entre matrices. Un anillo (no trivial) sin divisores de cero es un *dominio de integridad* (por ejemplo, el conjunto $\mathbb{Z}$ de los números enteros), y en él es válida la regla de simplificación para el producto: $ac = bc \Rightarrow a = b$ cuando $c \neq 0$. Todo dominio de integridad compuesto por un número finito de elementos es un cuerpo.

Fijo $\pi(x) \in F_q[x]$, dos polinomios $a(x)$ y $b(x)$ de $F_q[x]$ son *congruentes* si proporcionan el mismo resto al ser divididos por $\pi(x)$. Es fácil comprobar que la relación de congruencia es una equivalencia compatible con las operaciones de suma y producto definidas en $F_q[x]$. El conjunto cociente de esta relación se simboliza por $F_q[x]/\pi(x)$ y consta de las clases

$$[r(x)] = \{r(x) + a(x)\pi(x),\ a(x) \in F_q[x]\}$$

de representante $r(x) \in F_q[x]$, siendo $r(x)$ un polinomio cualquiera de grado menor que $\pi(x)$. Las operaciones de suma y producto inducidas por la congruencia en $F_q[x]/\pi(x)$ se definen, para cualesquiera operandos $a(x)$ y $b(x)$, por

$$\begin{aligned} a(x) + b(x) &\equiv (a(x) + b(x)) \mod \pi(x) \\ a(x) \cdot b(x) &\equiv (a(x)b(x)) \mod \pi(x) \end{aligned}$$

Consisten simplemente en tratar a los operandos como polinomios ordinarios de $F_q[x]$ y en reducir el resultado módulo $\pi(x)$. Por consiguiente, satisfacen las mismas propiedades que la suma y el producto de $F_q[x]$.

TEOREMA 7.18. $\forall \pi(x) \in F_q[x],\ (F_q[x]/\pi(x), +, \cdot)$ *es un anillo unitario conmutativo.*

La demostración del siguiente teorema se apoya en la unicidad de la factorización de polinomios por medio del algoritmo de división euclídeo, pero se pospone hasta el capítulo 9.

TEOREMA 7.19. $(F_q[x]/\pi(x), +, \cdot)$ *es un cuerpo conmutativo si y solamente si* $\pi(x) \in F_q[x]$ *es irreducible en* $F_q[x]$.

Si un subconjunto $\mathcal{S} \neq \emptyset$ de un anillo $\mathcal{A}$ es cerrado para las operaciones de suma y producto, es decir, si

$$\forall\ s_1, s_2 \in \mathcal{S},\ s_1 - s_2 \in \mathcal{S},\ s_1 \cdot s_2 \in \mathcal{S},\ s_2 \cdot s_1 \in \mathcal{S}$$

entonces $\mathcal{S}$ es un *subanillo* de $\mathcal{A}$.

En muchos anillos se da una situación más general: para algunos subanillos suyos ocurre que el producto entre un elemento del anillo y un elemento del subanillo pertenece también al subanillo.

Definición 7.9 (Ideal). *Sea* $(\mathcal{A}, +, \cdot)$ *un anillo e* $\mathcal{I} \neq \emptyset$ *un subconjunto suyo. Si*

$$\begin{aligned} &\forall s_1, s_2 \in \mathcal{I}, \quad s_1 - s_2 \in \mathcal{I} \\ &\forall a \in \mathcal{A}, i \in \mathcal{I}, \quad a \cdot i \in \mathcal{I}, i \cdot a \in \mathcal{I} \end{aligned}$$

entonces $(\mathcal{I}, +, \cdot)$ *es un ideal del anillo* $\mathcal{A}$*.*

Por ejemplo, para $p \in \mathbb{N}$ fijo, el conjunto $p\mathbb{Z} = \{np : n \in \mathbb{Z}\}$ de múltiplos enteros de p es un ideal de $\mathbb{Z}$.

Un tipo particularmente sencillo de ideal es un *ideal principal*, compuesto por todos los múltiplos de un elemento i del ideal. Es habitual designar a ese elemento como *generador* y representar el ideal por $\langle i \rangle$. Cuando todos los ideales de un anillo son principales, se tiene un *anillo de ideales principales.*

De las definiciones anteriores resultan las siguientes proposiciones.

Teorema 7.20. $F_q[x]/x^n - 1$ *es un anillo de ideales principales.*

Demostración. Cualquier ideal $\mathcal{I} \subseteq F_q[x]/x^n - 1$ debe contener un polinomio no nulo $g(x)$ de grado mínimo. Si $a(x) \in \mathcal{I}$, entonces escribiendo

$$a(x) = b(x)g(x) + r(x) \Rightarrow r(x) = a(x) - b(x)g(x) \in \mathcal{I}$$

vemos que necesariamente $r(x) = 0$ y, por tanto, $\mathcal{I} = \langle g(x) \rangle$. ▶

Teorema 7.21. *Un código cíclico* $\mathcal{C}[n, k]$ *sobre* $\mathrm{GF}(q)$ *es un ideal (principal) del anillo de polinomios* $F_q[x]/x^n - 1$.

Demostración. Es obvio que $\mathcal{C}[n, k] \subseteq F_q[x]/x^n - 1$. Además,

$$\forall\ p_1(x), p_2(x) \in \mathcal{C}[n, k],\ p_1(x) - p_2(x) \in \mathcal{C}[n, k]$$

y $\forall\ p(x) \in \mathcal{C}[n, k], a(x) \in F_q[x]$,

$$a(x) \cdot p(x) = p(x) \cdot a(x) = \sum_{i=0}^{\text{grado de } a(x)} a_i p^{(i)}(x) \in \mathcal{C}[n, k]$$

de suerte que $\mathcal{C}[n, k]$ es un ideal de $F_q[x]/x^n - 1$. ▶

TEOREMA 7.22. *Si $\mathcal{C}^a$ es el código acortado obtenido al eliminar los primeros $l < k$ símbolos de un código cíclico $\mathcal{C}[n,k]$, entonces $\mathcal{C}^a$ es un ideal de $F_q[x]/f(x)$, para cierto $f(x) \in F_q[x]$. Recíprocamente, un ideal del anillo $F_q[x]/f(x)$ es un código cíclico acortado.*

DEMOSTRACIÓN. Sea $v(x) \in \mathcal{C}^a$ un vector del código acortado y sea $f(x)$ una palabra de $\mathcal{C}$ de grado $n-l$. Entonces

$$xv(x) \mod f(x) = \begin{cases} xv(x), & \text{si grado}(xv(x)) < n-l \\ xv(x) - f(x), & \text{si grado}(xv(x)) = n-l. \end{cases}$$

En cualquiera de los dos casos, grado$(xv(x) \mod f(x)) < n-l$, y por tanto $xv(x) \mod f(x) \in \mathcal{C}^a$, ya que es una palabra del código con los l primeros símbolos a cero. Esto prueba que $\mathcal{C}^a$ es un ideal de $F_q[x]/f(x)$.

Recíprocamente, si $f(x)$ es cualquier palabra del código de grado $n-l$ de $\mathcal{C}$, e $\mathcal{I}$ un ideal de $F_q[x]/f(x)$, el menor ideal que contiene a los polinomios

$$\mathcal{I}^* = \{x^j i(x) \mod x^n - 1 : i(x) \in \mathcal{I}, j \geq 0\}$$

es un ideal de $F_q[x]/x^n - 1$ que contiene a $f(x)$, o sea un código cíclico del cual $\mathcal{I}$ es un código acortado. ▶

EJEMPLO 7.11. A partir del código cíclico binario $[7,3]$ generado por $g(x) = x^4 + x^3 + x^2 + 1$ se obtiene el código recortado $[6,2]$

$$\begin{aligned} \mathcal{C}^a &= \{000000, 111010, 100111, 011101\} \\ &= \{0, x^5 + x^4 + x^3 + x, x^5 + x^2 + x + 1, x^4 + x^3 + x^2 + 1\}, \end{aligned}$$

un conjunto de polinomios del cual el mónico de menor grado, $x^4+x^3+x^2+1$, es generador, porque

$$\begin{aligned} x^5 + x^4 + x^3 + x &= x(x^4 + x^3 + x^2 + 1) \\ x^5 + x^2 + x + 1 &= (x+1)(x^4 + x^3 + x^2 + 1). \end{aligned}$$

Pues bien, si $f(x) = a(x)(x^4 + x^3 + x^2 + 1)$ es un polinomio binario de grado 6, $\mathcal{C}^a$ es un ideal de GF(2)$[x]/f(x)$. Por ejemplo, para $f(x) = x^2(x^4 + x^3 + x^2 + 1) = x^6 + x^5 + x^4 + x^2$, se tiene

$$\begin{aligned} x(x^4 + x^3 + x^2 + 1) &\mod f(x) = x^5 + x^4 + x^3 + x \in \mathcal{C}^a \\ x(x^5 + x^4 + x^3 + x) &\mod f(x) = 0 \in \mathcal{C}^a \\ x(x^5 + x^2 + x + 1) &\mod f(x) = x^5 + x^4 + x^3 + x \in \mathcal{C}^a \end{aligned}$$

lo que prueba que $\mathcal{C}^a$ es un ideal. Observe que $x^j \mathcal{C}^a = 0 \mod f(x)$ para $j \geq 2$. Para finalizar, compruebe que

$$\langle g(x) \rangle = \mathcal{C}^a \cup x \cdot \mathcal{C}^a \cup x^2 \cdot \mathcal{C}^a \cup x^3 \cdot \mathcal{C}^a.$$ ■

n	Factorización (en octal)
1	6
7	6.54.64
9	6.7.444
15	6.7.46.62.76
17	6.471.727
21	6.7.54.64.634.724
23	6.5343.6165
25	6.76.4102041
27	6.7.444.4004004
31	6.45.51.57.67.73.75
33	6.7.4522.6106.7776
35	6.54.64.76..57134.72364
39	6.7.57074.74364.77774
41	6.5747175.6647133
43	6.47771.52225.64213
45	6.7.46.62.76.444.40044.44004
47	6.43073357.75667061
49	6.54.64.40001004.40200004
51	6.7.433.471.637.661.727.763
55	6.76.7776.5551347.7164555
57	6.7.5604164.7565674.7777774
63	6.7.54.64.414.444.534.554.604.634.664.714.724
127	6.406.422.436.442.472.516.526.562.576.602.626.646. 652.712.736.742.756.772

Tabla 7.7. Factorización de $x^n - 1$ en polinomios binarios irreducibles para $n \leq 63$ y $n = 127$. Los factores de los polinomios de exponente par, $x^{2m} - 1$, no se han incluido, ya que $x^{2m} - 1 = (x^m - 1)^2$. Tampoco aparece la factorización de algunos polinomios cuyo exponente n es un número primo porque para ellos es $x^n - 1 = (x - 1)(x^{n-1} + \cdots + 1)$, con ambos factores irreducibles. La representación de los factores se ha codificado en octal, con los términos de menor grado a la izquierda, y el punto actuando de separador de los factores. Así, por ejemplo, la segunda entrada en la tabla significa que $1 + x^7 = 6.54.64 = 110.101100.110100(\text{en binario}) = (1 + x)(1 + x^2 + x^3)(1 + x + x^3)$.

CAPÍTULO 8

Protocolos de retransmisión

En un sistema de comunicaciones los códigos de control de errores se pueden utilizar cuando menos de dos formas distintas:

- Como códigos correctores de errores. Los sistemas de transmisión de datos concebidos para operar con técnicas de corrección de errores en recepción (técnicas *FEC, Forward Error Correction*) intentan determinar los errores producidos, y corregirlos si los hubiera, en cada bloque de bits recibido. En caso de no poder hacerlo, la decodificación del mensaje es incorrecta y se entregan datos equivocados en el punto de destino. Para que la probabilidad de este evento se mantenga en unos límites tolerables, se requiere un número elevado de bits de redundancia por bloque (alrededor de dos por cada bit de información) que permitan asegurar la corrección de suficientes patrones de error. El uso de las técnicas FEC exige, por tanto, equipos de codificación y decodificación con una capacidad computacional grande.

- Como códigos de detección de errores. Con este modo de operar, cada vez que el receptor detecta un error en un bloque de bits, solicita al emisor la retransmisión del bloque afectado, enviándole una señal por un canal de retorno. Por lo tanto, la única posibilidad de que el destino obtenga datos erróneos con esta clase de técnicas de petición automática de retransmisión (técnicas *ARQ, Automatic Repeat reQuest)* es que falle la detección de la presencia de errores. La ventaja fundamental de este esquema es, claro está, que se requiere menor redundancia por cada símbolo de la fuente para la detección que para la corrección de los posibles fallos. A cambio, algunos datos hay que transmitirlos más de una vez. Tampoco es una técnica posible en todos los sistemas de telecomunicaciones. En particular, no sirve si no hay canal de

retorno o si los requisitos temporales impiden utilizarlo, como sucede por ejemplo cuando hay que transmitir o reproducir información en tiempo real.

El uso de las técnicas ARQ plantea la necesidad de definir un conjunto de normas para determinar en qué casos y cuándo pedir (el destino) o generar (la fuente) las retransmisiones. Tal conjunto de reglas lógicas ha de ser:

- Completo: debe prever todos los casos que puedan producirse.
- Coherente: no debe haber reglas contradictorias.

El conjunto de reglas lógicas, convenios de representación y procedimientos operativos que gobiernan el intercambio de datos entre dos entidades (sean ya equipos de telecomunicaciones, ya usuarios) se denomina *protocolo*. Puesto que se trata, en lo fundamental, nada más que de una secuencia de instrucciones que especifica las acciones coordinadas del emisor y del receptor, también se podría definir un protocolo como un algoritmo distribuido, con la importante particularidad añadida de que los protocolos operan en canales que presentan errores de transmisión. En general, los algoritmos distribuidos son más difíciles de comprender y de concebir que los algoritmos secuenciales clásicos.

Este capítulo presenta los mecanismos de operación de los protocolos de retransmisión. Se tratan, en primer lugar, las diversas estrategias ARQ para el control de los errores, analizando sus reglas de funcionamiento y sus propiedades lógicas. Después, se cuantifica un importante parámetro de prestaciones: la utilización eficaz de la capacidad de transmisión.

8.1. Estrategias ARQ

El objetivo de una estrategia ARQ es el de proporcionar a los extremos de origen y de destino la abstracción de un canal fiable, es decir, de un canal capaz de preservar la integridad y la secuencia de los mensajes que se transmiten a través de él.

Para nuestros propósitos, se puede suponer que la información generada por la fuente estará representada como una secuencia de bits de contenido y longitud arbitrarios. Pero el emisor no va a transmitir esta cadena como un grupo contiguo e ininterrumpido de bits, sino que dividirá la secuencia de bits en fragmentos más pequeños, a los que añadirá cierta información adicional de control (por ejemplo, bits de redundancia), y transmitirá cada uno de los fragmentos individualmente. Estas unidades de transmisión de datos, compuestas por bits de información de la fuente (datos) y bits de

control, reciben el nombre de *tramas*. En cuanto que secuencias de bits a las que se añaden símbolos redundantes para el control de errores, lo que aquí hemos definido como tramas son, por tanto, palabras de un código de control de errores dado.

La división de los mensajes en tramas de longitud acotada, por lo general de unos pocos cientos o miles de bits, obedece a varias razones:

a) La probabilidad de un error de transmisión en un bloque de bits aumenta casi proporcionalmente a su longitud; por lo que es muy poco probable transmitir una secuencia larga de bits sin errores, salvo si se emplea un código de canal sofisticado, sacrificando con ello la eficiencia.

b) En algunos casos, el emisor ha de esperar a que la fuente genere todos los bits de datos de la trama antes de construirla y proceder a su transmisión; por lo que mayor número de bits por trama implica mayor retardo.

c) Los sistemas de transmisión suelen imponer un límite máximo a la longitud de un bloque de bits consecutivos que pueden enviar, normalmente para no incurrir en problemas de sincronización entre los equipos de transmisión y recepción.

Adoptaremos, por conveniencia, la hipótesis de que el receptor es capaz de detectar todas las tramas recibidas con algún error. Es conocido que la detección de todos los errores no es una suposición realista. Se justifica, en todo caso, porque el ámbito de la presente discusión se limita a probar que las estrategias ARQ funcionan con corrección y eficiencia, excepto cuando hay errores indetectables.

Se dice que una estrategia funciona correctamente cuando el usuario o el proceso destino obtienen una y sólo una copia sin errores de cada trama enviada; y se dice que la eficiencia es mayor cuando el tiempo que el canal permanece ocupado por tiempos de espera y retransmisiones innecesarias es pequeño. Precisaremos estos conceptos más adelante. De momento es suficiente con señalar que se trata de dos propiedades independientes, que es posible disociar. Porque, para determinar si un protocolo es correcto, basta con estudiar sus propiedades lógicas, prescindiendo de la cantidad de tiempo que transcurre entre una acción y otra; pero, para calcular la eficiencia, es preciso evaluar el tiempo que consume cada acción o realización del protocolo, sin que sea preciso examinar por completo el significado de cada una de las acciones elementales.

Por lo demás, el modelo de canal punto a punto que se considera en este capítulo se basa en las siguientes hipótesis:

1. Provoca errores y/o pérdidas de algunas tramas. Cuando se detecta error en una trama, el receptor no hace ningún intento de corregirlas; simplemente las ignora y solicita su retransmisión. Las pérdidas ocurren cuando se interrumpe el paso de los bits en el canal o cuando, por ejemplo, el receptor no delimita correctamente el principio o el final de alguna trama. A efectos del receptor, una trama perdida es en todo equivalente a una trama no recibida.

2. Entrega al nodo de destino todas las tramas exactamente en el mismo orden en que fueron transmitidas.

3. Afecta a cada trama con un retardo de propagación variable aunque finito. El retardo de propagación es el tiempo que la señal electromagnética tarda en recorrer el enlace y su valor es $t_p = \frac{L}{v_p}$ si L es la longitud física del medio de transmisión y v_p la velocidad de propagación de la señal. Sin embargo, la coherencia lógica de las reglas de retransmisión que se van a explicar no va a depender de este tiempo, de modo que, por generalidad, supondremos que el retardo de propagación es arbitrario, que no tiene por qué ser idéntico para todas las tramas y que está acotado.

Al margen de la idoneidad de estas hipótesis, conviene subrayar que conllevan una generalización de cualquier canal de comunicaciones punto a punto porque eliminan todos los requisitos temporales. Más allá de conservar el orden total de las tramas recibidas y mantener la causalidad,[1] no se supone ningún sincronismo entre los instantes de comienzo de transmisión de las tramas, ni entre éstos y los instantes en que se reciben. En otras palabras, las estrategias ARQ no deben conducir a errores lógicos de operación, cualesquiera que sean los retardos que introduzcan el emisor, el receptor o el canal, debiendo ser, por lo tanto, absolutamente independientes de las características de éste último.

Para apreciar el alcance de este requisito, conviene reparar en que la naturaleza de la comunicación entre dos puntos conectados por un medio de transmisión directo es doblemente asíncrona. Por una parte, el retardo desde que una trama comienza a ser transmitida hasta que es recibida completamente libre de errores es, como se verá, variable, debido a la posibilidad de tener que retransmitirla. Y, por otra parte, el intervalo de tiempo que transcurre entre la transmisión de dos tramas consecutivas es también variable, como consecuencia del carácter aleatorio según el cual una fuente

[1] Formalmente: para cualesquiera dos tramas p_1 y p_2 que se transmiten, respectivamente, en los instantes de tiempo t_1 y t_2 ($t_1 < t_2$), y se reciben en los instantes t'_1 y t'_2, siempre ocurre que $t'_1 \geq t_1$, $t'_2 \geq t_2$ y $t'_2 > t'_1$.

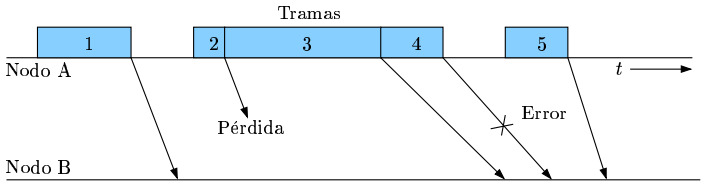


Figura 8.1. Modelo de transmisión de tramas.

produce información.[2] Planteado en términos genéricos, el problema final consiste en convertir un canal digital sometido a una tasa de error por bit inaceptable en un canal digital virtual, asíncrono y libre de errores.

La figura 8.1 resume en un diagrama de tiempos esquemático las características del canal sobre el que operan las estrategias de retransmisión cuya descripción se acomete en los siguientes apartados.

8.1.1. Parada y espera

El mecanismo más simple de retransmisión se denomina *parada y espera*. Su principio de funcionamiento es que el transmisor se asegure de que una trama haya sido recibida correctamente antes de transmitir la siguiente. Las reglas básicas de operación son sencillas:

R1 El emisor transmite una trama y permanece a la espera de recibir una confirmación por parte del receptor.

R2 Por cada trama recibida, el destino responde con un mensaje de control denominado asentimiento, que indica al emisor si en la trama se han detectado o no errores. El receptor acepta todas las tramas en las que no detecta error, y descarta (elimina o prescinde de) las demás. Los asentimientos que informan de algún error de transmisión se denominan asentimientos negativos, en tanto que los que señalan la ausencia de error reciben el nombre de asentimientos positivos.

R3 Si el emisor recibe un asentimiento negativo, entonces retransmite la última trama enviada. Si el asentimiento es positivo, transmitirá la siguiente trama cuando disponga de ella.

[2] A este respecto puede argumentarse que todas las fuentes generan información a tasa variable, y que es sólo el proceso de codificación de fuente el que convierte los datos en una secuencia de símbolos a intervalos regulares. De hecho, los codificadores modernos de señales de voz o vídeo generan secuencias de bits con velocidad no constante, reflejando así el comportamiento dinámico de la señal.

Este elemental procedimiento de intercambio de tramas y confirmaciones funciona perfectamente siempre y cuando no se pierda ninguna trama de datos ni de asentimiento. Ante un suceso así, y en concordancia con las reglas anteriores, ambos extremos permanecerían indefinidamente a la espera de un evento que no ocurriría: el emisor aguardaría una respuesta que no llegará, mientras que el destino no recibirá jamás la trama que espera. Una situación como la descrita, en la que dos (o más si los hubiera) participantes en un sistema distribuido no progresan por estar a la espera de que se realicen acciones externas, se conoce como situación de bloqueo (*deadlock*).

Una posibilidad para la ruptura de las situaciones de bloqueo es el uso de temporizadores en el extremo emisor. Si, agotado un tiempo de espera tras finalizar la transmisión de una trama, el emisor no ha recibido ningún asentimiento válido, entonces supone que se han perdido los datos y retransmite la última trama. Por supuesto, un valor elevado de temporización desperdicia tiempo de espera, mientras que un valor demasiado pequeño genera retransmisiones innecesarias. La selección de un valor óptimo para un canal dado depende de múltiples factores y no es, en general, una tarea simple.

Pero es justamente la nueva posibilidad de efectuar retransmisiones, combinada con la presencia de errores en el canal, la causa que puede inducir a confusión al receptor, y ello por dos motivos:

- Porque el nodo receptor no es capaz de distinguir entre una trama nueva y la retransmisión de una trama antigua.
- Porque el emisor asocia un asentimiento con la última trama de datos enviada.

En la figura 8.2 se han representado las dos causas posibles de ambigüedad: la incapacidad de distinguir entre una trama nueva y la retransmisión de la anterior, y la imposibilidad de asociar un asentimiento con la trama de datos que lo generó. En el escenario de la parte izquierda de la figura, el receptor no tiene forma de averiguar que la segunda trama no es sino una retransmisión de la primera trama recibida, causada por el vencimiento del temporizador. Comparar los datos de ambas tramas no sirve, ya que es perfectamente posible que el proceso emisor envíe dos mensajes con el mismo contenido. En el escenario de la parte derecha de la figura, el emisor desconoce si el último asentimiento recibido corresponde a la última trama enviada o a una trama anterior, puesto que los asentimientos se pueden perder o pueden llegar con retraso.

En ambos casos el fallo se soluciona numerando secuencialmente tanto las tramas de datos como los asentimientos, de forma que a todas las transmisiones de una misma trama se les asigne el mismo número de secuencia,

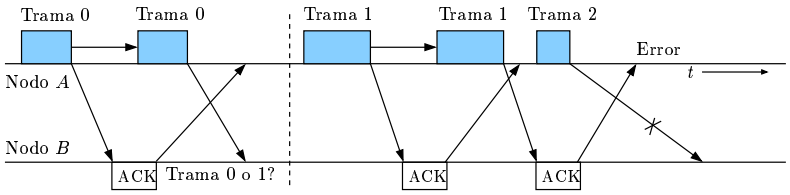


Figura 8.2. Problemas potenciales en parada y espera cuando las tramas de datos y los asentimientos no están numerados.

mientras que tramas distintas llevan siempre números de secuencia distintos. La numeración de las tramas de datos permite al receptor descubrir tramas duplicadas, y la numeración de los asentimientos sirve al emisor para determinar a qué trama de datos corresponde cada uno. El número de secuencia actúa en realidad como un nombre, un identificador único de cada trama de datos o asentimiento que establece una asociación biunívoca entre ellas.

Ahora bien, en parada y espera, el nodo receptor sólo necesita diferenciar entre una trama i y la anterior, $i-1$, por cuanto es imposible recibir la trama $i+1$ mientras el receptor no confirme la i. Si el receptor está a la espera de la trama i es porque ha recibido correctamente y ha confirmado la trama $i-1$. Pero antes de haber transmitido por primera vez la trama $i-1$, el emisor debe haber recibido la confirmación de la $i-2$. De modo que al receptor sólo pueden llegar o bien la trama i que se espera, o bien una retransmisión de la trama $i-1$ si en el emisor no consta aún el último asentimiento enviado.

Razonando análogamente, el nodo emisor sólo debe distinguir entre el asentimiento de la última trama de datos enviada, digamos la i, y el de la penúltima, $i-1$. Es imposible recibir un asentimiento de la trama $i-2$, pues, para poder transmitir la i-ésima, el emisor necesita antes un asentimiento de la $i-1$.

En definitiva, bastan dos números de secuencia (0 y 1, un único bit) tanto en las tramas de datos como en los asentimientos para tener la seguridad de que ambos extremos detectan todas las tramas duplicadas por retransmisión. De ahí que parada y espera se conozca también como *protocolo de bit alternante*.

En las implementaciones reales los asentimientos suelen llevar como número de secuencia el correspondiente a la siguiente trama de datos que el nodo destino espera recibir. Se trata nada más que de un convenio semántico, con un importante corolario práctico. De esta manera, en el emisor, un asentimiento con el mismo número de secuencia que el de la última trama de datos enviada se interpreta como un asentimiento negativo, es decir, el número de secuencia que transporta un asentimiento es siempre el número

de trama que el receptor solicita.

Tras incorporar estas modificaciones a las reglas originales, el funcionamiento de los nodos con parada y espera es como sigue:

R1 El emisor transmite una trama con el siguiente número de secuencia en módulo-2, inicia un temporizador y permanece a la espera de recibir una confirmación por parte del receptor.

R2 Si la trama se recibe correctamente, el receptor la acepta y responde con un asentimiento con número de secuencia igual al de la siguiente trama que espera recibir. Si se detectan errores, el receptor descarta la trama y responde con un asentimiento con número de secuencia igual al de la trama que espera recibir.

R3 Si el emisor recibe un asentimiento con número de secuencia igual al de la última trama que envió, o expira el temporizador sin haber obtenido respuesta, entonces retransmite la última trama; si recibe un asentimiento con número de secuencia distinto al de la última trama enviada, incrementa el número de secuencia y transmite la siguiente trama, en caso de disponer de ella.

Hay en estas reglas varios aspectos deliberadamente indefinidos. No indican en qué forma se han de representar en la trama los asentimientos ni los bits de control, ni tampoco señalan el valor del intervalo de temporización. Simplemente es notorio que la especificación de estas características no afecta a la forma de operar de la estrategia, aunque, como es natural, un protocolo real debe definir con precisión estos detalles.

Pese a que una prueba formal completa es muy laboriosa, se puede comprobar de manera intuitiva que ningún intercambio de tramas compatible con las reglas anteriores conduce a fallos de operación. Sin embargo, en general, la dificultad de comprensión de un algoritmo distribuido aumenta notablemente con el número de condiciones lógicas impuestas a cada participante. Por ello, en protocolos menos elementales no cabe confiar a la intuición la tarea de deducir la total coherencia lógica del comportamiento del sistema, y más aún cuando la descripción de este comportamiento suele hacerse en un lenguaje poco preciso como es el lenguaje natural.

Asegurarse de la corrección de un protocolo significa, como mínimo, garantizar dos propiedades fundamentales de comportamiento, una de avance y otra de coordinación. La primera de ellas es la ausencia de bloqueo, es decir, la inexistencia de algún estado en el que ninguno de los participantes del sistema pueda evolucionar. La noción de bloqueo debe entenderse de manera amplia: tampoco es tolerable la existencia de ciclos improductivos, aquéllos que consisten en una sucesión repetitiva de estados en los

cuales no se produce ningún avance neto del algoritmo. La segunda condición debe ser la certeza de que, desde cualquier estado inicial válido, todas las evoluciones posibles del sistema también son válidas, es decir, llevan a estados válidos de acuerdo con el comportamiento global que se desea. La verificación de ambas propiedades en un sistema complejo es un problema que se aborda sistemáticamente mediante métodos de especificación formal, que son lenguajes matemáticos algebraicos o lógicos (o, más a menudo, una combinación de ambos tipos) creados para facilitar el razonamiento simbólico y la deducción automática. Existen multitud de técnicas de verificación y validación formal de protocolos (tales como la teoría de autómatas finitos, el álgebra de procesos, diferentes clases de lógica o las redes de Petri) con diferente expresividad, pero este campo de estudio es muy amplio y excede con mucho el ámbito del presente texto. El lector interesado puede consultar alguna obra especializada, como [33].

8.1.2. Envío continuo con rechazo simple

Parada y espera resulta ser una estrategia poco eficaz porque el tiempo de espera por un asentimiento no se utiliza para transmitir nuevas tramas. Este tiempo es, como mínimo, la suma del tiempo de transmisión del asentimiento más dos veces el retardo de propagación de la señal eléctrica por el canal,[3] y puede llegar a ser relativamente grande frente al tiempo de transmisión de una trama de datos. En la estrategia de envío continuo con rechazo simple se elimina esa restricción superflua: el emisor puede transmitir varias tramas consecutivas sin haber recibido todavía el asentimiento de la primera. O, expresado de manera equivalente, al emisor le está permitido tener varias tramas pendientes de asentimiento.

El receptor opera, en esencia, del mismo modo que en parada y espera: acepta las tramas recibidas sin error en el orden correcto (en número de secuencia creciente) y responde individualmente a cada una con un asentimiento cuyo número indica el número de secuencia de la siguiente trama esperada. En la práctica, para prevenir las retransmisiones originadas por posibles pérdidas de los asentimientos, se adopta como convenio que una trama de asentimiento con número de secuencia i confirma no sólo la $i-1$ de datos sino también todas las anteriores. Según este criterio, los asentimientos positivos son acumulativos.

Un parámetro $n \geq 1$ limita el número máximo de tramas pendientes de confirmación en el emisor. Así, cuando i es el número de secuencia recibido en el último asentimiento, el emisor podrá transmitir las tramas con los nú-

[3] Cuando en el enlace hay transmisión de datos en ambos sentidos, el tiempo de espera puede llegar a ser hasta dos veces el tiempo máximo de transmisión de una trama de datos más el retardo de propagación de ida y vuelta.

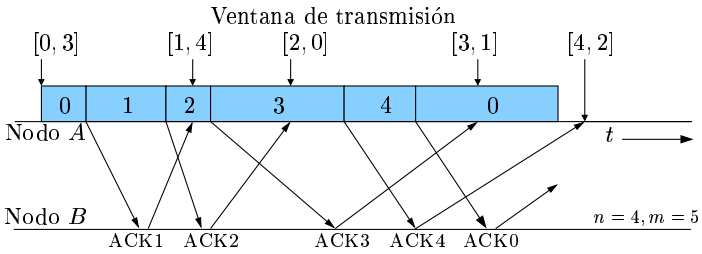


FIGURA 8.3. Estrategia de retransmisión de envío continuo con rechazo simple: operación sin errores de transmisión.

meros de secuencia $i, i+1, \ldots, i+n-1$. El intervalo de números de secuencia permisibles $[i, i+n-1]$ se puede visualizar como una *ventana deslizante* de emisión, un rango de números de secuencia válidos cuyos extremos inferior y superior se modifican cada vez que se recibe un asentimiento, de acuerdo con la siguiente regla: si el número del último asentimiento sin errores recibido es $j \geq i$, la ventana de emisión se desplaza de inmediato a $[j, j+n-1]$.

En caso de que el emisor haya transmitido n tramas consecutivas y no disponga del asentimiento de la primera de ellas, debe retroceder y transmitir de nuevo toda la ventana, comenzando por la primera trama sin confirmar. Este procedimiento es la razón de que el rechazo simple se conozca en la literatura anglosajona como *go-back-n*.

El receptor descarta todas las tramas con errores y todas aquéllas cuyo número de secuencia esté fuera de orden, aun si son correctas. Puede decirse, por tanto, que su ventana de recepción es de tamaño uno.[4]

En cuanto a la numeración, se demuestra que, para evitar fallos de protocolo, es condición necesaria y suficiente numerar tramas y asentimientos módulo m, siendo $m > n$. Porque si el número de secuencia de la trama de datos que el receptor espera es j, entonces éste debe haber recibido correctamente y debe haber confirmado todas las tramas hasta la $j-1$. Y, para que eso suceda, el extremo inferior de la ventana de emisión debe valer al menos $j-n$, porque de otro modo el emisor no podría haber transmitido la trama con número de secuencia $j-1$. Y el extremo inferior podrá valer como máximo j, puesto que el receptor todavía no ha recibido correctamente la trama con número j y no ha podido asentirla.

Así pues, el extremo inferior de la ventana de emisión estará en el rango $[j-n, j]$, y el emisor podrá transmitir n tramas consecutivas numeradas a partir de él. Para que sepa sin ambigüedad a qué trama de datos corresponde el ACK-j, necesita un mínimo de $n+1$ números de secuencia, de modo que el

[4] A este respecto, la estrategia de parada y espera es un caso particular del envío continuo con rechazo simple con ventana de emisión de tamaño 1.

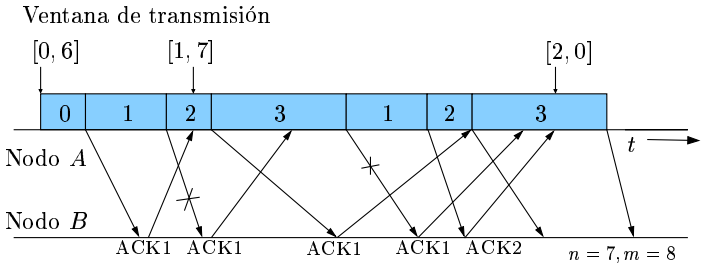


Figura 8.4. Estrategia de retransmisión de envío continuo con rechazo simple. Efecto de los errores en las tramas de datos.

extremo inferior de la ventana de emisión no comprenda nunca dos números de secuencia iguales. Para ello es suficiente numerar tramas y asentimientos módulo-m, siendo $m > n$. En realidad ni siquiera es necesario que se trate de números correlativos, sino que bastaría con cualquier secuencia periódica de $m > n$ números.

Ejemplo 8.1. Para constatar que la estrategia de retransmisión con envío continuo y rechazo simple es incorrecta cuando el módulo de la numeración secuencial de las tramas es menor o igual que el tamaño de la ventana, considérese, por ejemplo, $m = n$ y la siguiente secuencia de eventos ordenada en el tiempo:

1. El emisor transmite las tramas de datos 0, ..., $n - 1$, y se detiene a la espera de su confirmación.

2. Todas las tramas de datos se reciben sin error, son aceptadas y el receptor responde con los asentimientos ACK-1, ACK-2, ..., ACK-0. El receptor queda a la espera de la próxima trama de datos, cuyo número de secuencia será 0.

3. Todos los asentimientos se pierden.

4. Acabado el plazo de temporización en el emisor, se retransmiten las tramas de datos 0, ..., $n - 1$.

5. Las tramas retransmitidas se reciben sin error y en secuencia, y son aceptadas por el receptor como tramas de datos nuevas.

Es evidente que el fallo se debe a que el número de secuencia de la trama que se espera recibir pertenece a la ventana que el emisor todavía no ha desplazado. ■

Reuniendo todas las piezas descritas hasta aquí, procede especificar de forma más precisa las reglas de operación de emisor y receptor en la estrategia de rechazo simple con tamaño de ventana $n \geq 1$ y numeración módulo $m > n$. El emisor utiliza dos variables internas, $\mathrm{NS_{inf}}$ y $\mathrm{NS_{sup}}$: $\mathrm{NS_{inf}}$ representa el menor número de secuencia de las tramas que no han sido aún asentidas; $\mathrm{NS_{sup}}$ indica el número de secuencia que le corresponderá a la próxima trama de datos nueva. El receptor necesita sólo una variable, NA, el número de secuencia del próximo asentimiento.

En el emisor:

1. Inicialmente, las variables $\mathrm{NS_{inf}}$ y $\mathrm{NS_{sup}}$ deben valer 0.
2. Si la ventana no se ha agotado, $(\mathrm{NS_{sup}} - \mathrm{NS_{inf}}) \mod m < n$, y hay una trama nueva pendiente de transmisión, asignarle el número $\mathrm{NS_{sup}}$, transmitir la trama e incrementar $\mathrm{NS_{sup}}$ en una unidad, $(\mathrm{NS_{sup}} + 1) \mod m$.
3. Si se recibe un asentimiento libre de errores con número de secuencia NA dentro de la ventana,

$$(\mathrm{NA} - \mathrm{NS_{inf}}) \mod m \leq (\mathrm{NS_{sup}} - \mathrm{NS_{inf}}) \mod m,$$

   entonces actualizar el extremo inferior de la ventana, $\mathrm{NS_{inf}} = \mathrm{NA}$.
4. Si hay tramas pendientes de confirmación ($\mathrm{NS_{inf}} \neq \mathrm{NS_{sup}}$), retransmitirlas en orden creciente módulo-m de número de secuencia.

Los pasos 2, 3 y 4 pueden ejecutarse en cualquier orden.

En el receptor:

1. Inicialmente, la variable NA debe valer 0.
2. Si se recibe una trama sin errores con número de secuencia igual a NA, aceptarla. Incrementar NA a $(\mathrm{NA} + 1) \mod m$ y transmitir un asentimiento con este número.
3. En otro caso, descartar la trama recibida y enviar un asentimiento con número de secuencia NA.

El incremento de eficiencia con esta estrategia en relación con la de parada y espera depende cualitativamente del cumplimiento de cuatro condiciones:

- Que el tiempo que tarda en llegar un asentimiento sea grande en relación al tiempo necesario, en media, para transmitir una trama de datos. Así, durante ese intervalo cabe la posibilidad de enviar un número de tramas elevado.

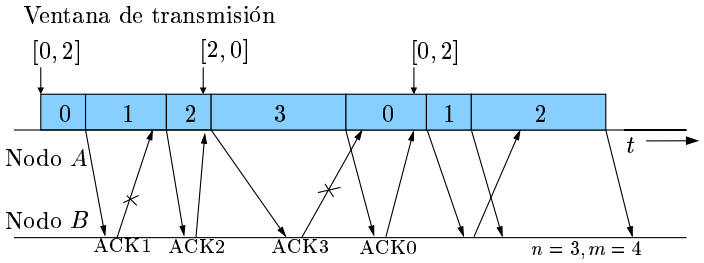


Figura 8.5. Estrategia de retransmisión de rechazo simple: efecto de los errores en los asentimientos.

- Que la ventana no se agote antes de recibir el ACK de la primera trama, de forma que, en ausencia de errores de transmisión, el emisor no deba retroceder nunca. El diseñador del protocolo puede imponer esta condición adoptando un tamaño de ventana suficientemente grande, de al menos dos veces el tiempo de propagación en el canal. Existen, sin embargo, algunas penalizaciones al empleo de ventanas de gran tamaño.
- Que el emisor sea advertido cuanto antes de los errores producidos en una trama de datos, pues todas las tramas posteriores a ella son rechazadas.
- Que los errores o pérdidas de asentimientos positivos no provoquen retransmisiones, pues son innecesarias. Hemos garantizado esta condición obligando a utilizar asentimientos acumulativos.

Las necesidades de memoria se han incrementado a n búferes en el emisor para mantener copia de las n últimas tramas de datos enviadas.

8.1.3. Envío continuo con rechazo selectivo

En la estrategia de envío continuo con rechazo selectivo, el receptor acepta todas las tramas de datos recibidas sin errores, aunque no lleguen en secuencia. Al igual que en el caso de rechazo simple, el número máximo de tramas pendientes de asentimiento en el emisor se limita a n. La diferencia estriba ahora en que el receptor también utiliza una ventana de tamaño n para los números de secuencia de las tramas que acepta: si i es el número de secuencia del último asentimiento enviado, cualquier trama de datos correcta con número de secuencia en el rango $[i, i+n-1]$ es aceptada.[5] Por tanto, y se trata de una diferencia sustancial en relación a las estrategias de parada

[5] La estrategia de envío continuo con rechazo simple resulta ser un caso particular del rechazo selectivo con un tamaño de ventana de recepción igual a 1.

y espera y de rechazo simple, en envío continuo con rechazo selectivo las tramas de datos *sin error* pueden recibirse en el destino en un orden distinto al de transmisión.

Los asentimientos se numeran estrictamente en secuencia; así es posible mantener el convenio de que confirman la trama i y todas las anteriores. En ausencia de la limitación del tamaño máximo de la ventana (que es de naturaleza práctica, porque ni emisor ni receptor disponen de memoria infinita), la estrategia de rechazo selectivo es ideal. De hecho, cada trama se transmite siempre el número mínimo de veces para hacerla llegar al nodo destino, puesto que el receptor la solicita expresamente con un asentimiento cada vez que percibe su ausencia. Si se parte de la interpretación de un asentimiento como solicitud de una trama de datos, debe resultar claro que la estrategia de envío continuo con rechazo selectivo es, más correctamente, una estrategia de envío continuo con *retransmisión* selectiva sólo de las tramas de datos con error.

De nuevo aparecen dificultades al considerar un campo de numeración de tamaño finito. En este caso, la condición necesaria y suficiente para no generar fallos es que los números de secuencia se calculen módulo m, siendo m al menos igual a dos veces el tamaño de las ventanas de emisión y recepción, $m \geq 2n$.

Veamos por qué: supongamos que la ventana de recepción, en un instante de tiempo dado, es $[j, j+n-1]$. Ello significa dos cosas:

1. Que el extremo inferior de la ventana de transmisión vale como mínimo $j-n$. Si así no ocurriera, el emisor no habría podido transmitir la trama con número de secuencia $j-1$, que no podría haber sido confirmada por el receptor, resultando imposible que éste quede a la espera de la trama j.

2. Que el extremo superior de la ventana de transmisión vale como máximo $j+n-1$. De no ser así, el extremo inferior de la ventana de emisión valdría al menos $j+1$, lo que querría decir que el receptor habría confirmado la recepción correcta de la trama j. Pero esto es imposible si el receptor espera precisamente la trama j.

Resulta, de estas dos observaciones, que al receptor sólo llegarán, en el peor de los casos, tramas de datos con número de secuencia en el rango $[j-n, j+n-1]$, dependiendo de cuál sea el último asentimiento correctamente recibido en el emisor. Y para distinguir esas $2n$ tramas bastan un mínimo de $2n$ números de secuencia diferentes. Al igual que en la estrategia de envío continuo con rechazo simple, tampoco aquí es preciso que los $m \geq 2n$ números de secuencia sean consecutivos. Cualquier secuencia periódica de periodo al menos $2n$ serviría.

Ejemplo 8.2. Un contraejemplo puede aclarar mejor este razonamiento. Imagine una estrategia de envío continuo con rechazo selectivo, tamaño de ventana n y numeración módulo-m, siendo $m < 2n$. Y considérese la sucesión ordenada de eventos:

1. El emisor envía las tramas de datos $0, \ldots, n-1$.
2. Las tramas de datos se reciben sin error, se aceptan y son confirmadas con los asentimientos ACK-1, ..., ACK-N. La ventana de recepción se desplaza por lo tanto a $[n, \ldots, 0, \ldots, 2n-m-1]$.
3. Todos los asentimientos se pierden.
4. Una vez agotado el plazo de espera de los asentimientos, el emisor retransmite las tramas de datos con números de secuencia $0, 1, \ldots, n-1$.
5. Las retransmisiones se reciben sin error y, puesto que $m < 2n$, las tramas con número de secuencia $0, \ldots, 2n-m-1$ son erróneamente aceptadas como válidas por el receptor.

Resulta patente que, para evitar el fallo, es condición necesaria y suficiente asegurar que los números de secuencia contenidos en la ventana de recepción desplazada no se solapen con los números de secuencia de la ventana de emisión, es decir, $m \geq 2n$. ■

Las reglas formales de operación de emisor y receptor en la estrategia de rechazo selectivo con tamaño de ventana $n \geq 1$ y numeración módulo $m \geq 2n$ son las siguientes. Sea $\mathrm{NS_{inf}}$, en el nodo emisor, el menor número de secuencia de las tramas que no han sido aún asentidas; sea $\mathrm{NS_{sup}}$ el número de secuencia de la próxima trama de datos nueva. Y, en el receptor, sea NA el número de secuencia del próximo asentimiento, $\mathrm{NA_{inf}}$ el extremo inferior de la ventana de recepción y sea Map un vector de n elementos.

Para el emisor:

1. Inicialmente, las variables $\mathrm{NS_{inf}}$ y $\mathrm{NS_{sup}}$ valen 0.
2. Si la ventana no se ha agotado, $(\mathrm{NS_{sup}} - \mathrm{NS_{inf}}) \mod m < n$, y hay una trama nueva pendiente de transmisión, asignarle el número $\mathrm{NS_{sup}}$ e incrementar $\mathrm{NS_{sup}}$ en una unidad, $(\mathrm{NS_{sup}} + 1) \mod m$. Transmitir la trama.
3. Si se recibe un asentimiento libre de errores con número de secuencia NA dentro de la ventana,

$$(\mathrm{NA} - \mathrm{NS_{inf}}) \mod m \leq (\mathrm{NS_{sup}} - \mathrm{NS_{inf}}) \mod m,$$

entonces actualizar el extremo inferior de la ventana, $\mathrm{NS_{inf}} = \mathrm{NA}$.

4. Si hay tramas pendientes de confirmación, $\mathrm{NS_{inf}} \neq \mathrm{NS_{sup}}$, retransmitirlas, no importa en qué orden.

Los pasos 2, 3 y 4 pueden ejecutarse en cualquier orden relativo.

Para el receptor:

1. Inicialmente, las variables NA y $\mathrm{NA_{inf}}$, y el vector MAP a 0.
2. Si se recibe una trama sin errores con número de secuencia igual a NA dentro de la ventana, $(\mathrm{NA} - \mathrm{NA_{inf}}) \mod m < n$, aceptarla y activar $\mathrm{MAP}[(\mathrm{NA} - \mathrm{NA_{inf}}) \mod m] = 1$.
3. Si $\mathrm{MAP}[0] = 1$, incrementar $\mathrm{NA_{inf}}$ a $(\mathrm{NA_{inf}} + 1) \mod m$ y desplazar el vector MAP una posición hacia la izquierda.
4. Si $\mathrm{MAP}[0] = 0$, enviar un asentimiento con número de secuencia $\mathrm{NA_{inf}}$.

Ha de advertirse que en rechazo selectivo tanto el emisor como el receptor precisan de n búferes para las tramas, y que el receptor además debe ser capaz de reordenarlas.

8.2. Análisis de la cadencia eficaz

Estudiaremos a continuación la tasa efectiva de transferencia de bits de información en presencia de retransmisiones, cuantificando el impacto de la estrategia ARQ en el uso de la capacidad del canal. Los resultados que desarrollemos confirmarán la intuición de que rechazo simple debe ser mejor que parada y espera, y que rechazo selectivo es, a su vez, mejor que ambos.

El parámetro objetivo de prestaciones será la *cadencia eficaz*, que definiremos como el cociente entre el número medio de bits de datos por trama, n, y el tiempo medio $\overline{t}_{\mathrm{oc}}$ que el canal permanece ocupado transmitiendo dicha trama:

$$C_e = \frac{\mathrm{E}[n]}{\mathrm{E}[t_{\mathrm{oc}}]} \; \frac{\text{bits}}{\text{unidad de tiempo}}.$$

En el apéndice 8.A se elabora con más cuidado esta definición. Por el momento, obsérvese que en el caso particular de que todas las tramas fuesen de la misma longitud, $n_i = n \quad \forall i$, el régimen efectivo de transmisión sería, simplemente,

$$C_e = \frac{n}{\mathrm{E}[t_{\mathrm{oc}}]}.$$

Con objeto de simplificar los cálculos se harán las siguientes hipótesis:

1. El emisor siempre dispone de tramas que transmitir, y éstas son de longitud constante $l = m + n$ bits, donde m es el número de bits de control de la trama, esto es, los bits que no llevan información de la fuente.

2. El receptor asiente cada trama de datos inmediatamente.

3. No se pierden tramas, y el receptor detecta todas las tramas con error.

4. No hay errores en los asentimientos, y la probabilidad de error en una trama de datos es proporcional a su longitud l.

5. El tiempo de asentimiento, T_{as}, es constante. Se define el tiempo de asentimiento como el tiempo que transcurre desde que se termina de transmitir una trama de datos hasta que su asentimiento se recibe completo y libre de errores en el emisor.

6. El tamaño de ventana es suficientemente grande para que no tenga que interrumpirse la transmisión en caso de envío continuo.

Discusión de las hipótesis

La suposición de que el emisor siempre dispone de tramas pendientes modela un enlace de datos saturado, condición con la que se obtiene la utilización máxima del canal. No es imprescindible suponer tramas con longitud constante; el análisis sería el mismo si se considerase en cada paso la longitud media de éstas. Pero suponer que no se pierden tramas es, por el contrario, una simplificación que evita tener que considerar no ya el efecto sino siquiera el uso de los temporizadores. En una situación normal, el valor de temporización suele ser mayor que el tiempo de asentimiento, de manera que, al obviarlo en el análisis, se calcula una cota superior a la cadencia eficaz real. Además, en envío continuo se toma un tamaño de ventana infinito, de nuevo para tratar con un enlace saturado.

En lo que respecta al tiempo de asentimiento, T_{as}, éste incluye el tiempo de procesamiento de una trama de datos en el receptor, el retardo de propagación de la señal de ida y vuelta y el tiempo de transmisión del asentimiento. No es constante, porque el tiempo de procesamiento no lo es, así como por el efecto de posibles pérdidas en los asentimientos, aunque se pueden argumentar las mismas consideraciones que para la longitud de las tramas: T_{as} se puede sustituir, en sentido estadístico, por el tiempo medio de asentimiento, y el análisis se mantiene correcto.

Las expresiones aproximadas de la probabilidad de error se justifican con el siguiente razonamiento: en un canal binario simétrico sin memoria con probabilidad de transición κ, la probabilidad de error en un bloque de

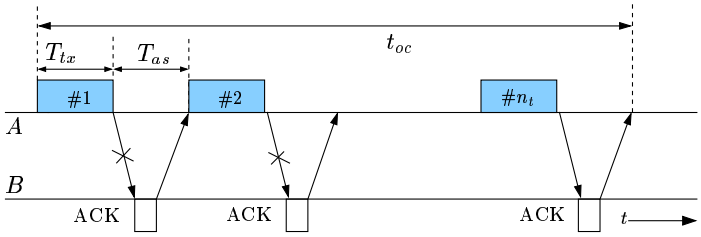


FIGURA 8.6. Cálculo del caudal eficaz en la técnica de parada y espera.

n bits es $1-(1-\kappa)^n = 1-\sum_{i=0}^{n}\binom{n}{i}(-\kappa)^i$, que podemos escribir como $1-(1-n\kappa+o(n\kappa))$. Cuando $n\kappa \ll 1$, se puede despreciar el término[6] $o(n\kappa)$, y entonces $1-(1-\kappa)^n \approx n\kappa$. También en canales con memoria se verifica esta misma condición, no siendo κ ahora la probabilidad de error en la transmisión de un símbolo, sino un promedio denominado típicamente tasa de error de bit. Esto explica que se trate la probabilidad de error como proporcional a la longitud de trama, y que no se tengan en cuenta posibles errores en los asentimientos, por ser mucho más cortos que las tramas de datos.

8.2.1. Parada y espera

En parada y espera no es posible comenzar a transmitir una trama hasta que la anterior ha sido confirmada. El tiempo de ocupación del canal debido a la transmisión de una trama es, por tanto, (ver la figura 8.6)

$$t_{\text{oc}} = \left(\frac{l}{C} + T_{\text{as}}\right) n_t$$

si C es el régimen nominal de transmisión del canal (bits/segundo) y n_t el número total de transmisiones necesarias. Pero n_t es una variable aleatoria discreta con distribución de probabilidad geométrica, $\mathrm{P}\{n_t = k\} = p^{k-1}(1-p), \quad k \geq 1$, siendo p la probabilidad de error en la transmisión de una trama. Tomando esperanzas se obtiene:

$$\overline{t}_{\text{oc}} = \left(\frac{l}{C} + T_{\text{as}}\right) \overline{n}_t = \frac{1}{1-p}\left(\frac{l}{C} + T_{\text{as}}\right)$$

que, sustituido en la definición de la cadencia eficaz, da:

$$C_e = (1-p)\frac{n}{n+m+CT_{\text{as}}}C$$

[6] Una función $f(p)$ es $o(p)$ si $\lim_{p\to 0}\frac{f(p)}{p} = 0$.

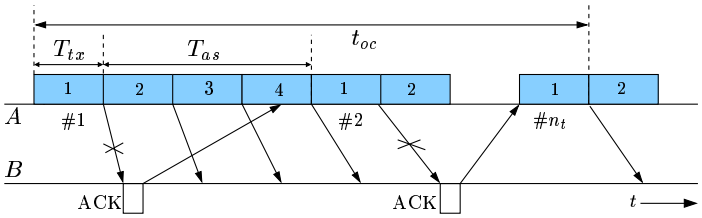


FIGURA 8.7. Cálculo del caudal eficaz en la técnica de envío continuo con rechazo simple.

El término CT_{as} que aparece en el denominador representa la cantidad de bits que el canal podría transmitir durante un periodo de tiempo igual a T_{as}. Cuando el factor dominante en el tiempo de asentimiento es el retardo de propagación de la señal eléctrica, CT_{as} es casi directamente proporcional al producto ancho de banda $\times$ distancia del canal. Si $n \gg m$ y se define $a(n) = \dfrac{CT_{\text{as}}}{n}$, se deduce la aproximación

$$\frac{C_e}{C} \approx (1-p)\frac{1}{1+a(n)}.$$

La cadencia eficaz relativa, C_e/C, depende entonces de p y del número de tramas de n bits de datos, $a(n)$, que se podrían enviar durante el intervalo T_{as}. Esto explica por qué parada y espera resulta poco adecuada cuando el retardo de propagación es grande en relación al tiempo de transmisión de una trama, bien porque se utiliza un canal de alta capacidad, bien porque el tiempo de propagación es de magnitud apreciable.

8.2.2. Rechazo simple

En rechazo simple todos los asentimientos negativos hacen retroceder al emisor, mientras que la última (re)transmisión ocupa solamente el tiempo de envío de la trama. Se puede escribir, por tanto, (ver la figura 8.7)

$$t_{oc} = \left(\frac{l}{C} + T_{\text{as}}\right)(n_t - 1) + \frac{l}{C}.$$

El tiempo medio de ocupación vale, en este caso,

$$\overline{t}_{\text{oc}} = \frac{p}{1-p}\left(\frac{l}{C} + T_{\text{as}}\right) + \frac{l}{C}$$

y la cadencia eficaz

$$C_e = (1-p)\frac{n}{n+m+pCT_{\text{as}}}C.$$

La diferencia con la fórmula de parada y espera es el término pCT_{as} del denominador: expresa que en rechazo simple sólo se desaprovecha el intervalo de asentimiento para la fracción p de tramas que deben ser retransmitidas. Al igual que antes, si $n \gg m$ y $a(n) = \frac{CT_{\text{as}}}{n}$, entonces

$$\frac{C_e}{C} \approx (1-p)\frac{1}{1+pa(n)}.$$

8.2.3. Rechazo selectivo

Con un tamaño de ventana infinito, cada trama ocupa nada más que el tiempo de sus n_t transmisiones. Entonces

$$\begin{aligned} t_{\text{oc}} &= n_t \frac{l}{C} \\ \overline{t}_{\text{oc}} &= \frac{1}{1-p}\frac{l}{C} \\ C_e &= (1-p)\frac{n}{n+m}C. \end{aligned}$$

La expresión para la cadencia eficaz es óptima en el sentido de que ninguna estrategia de retransmisión puede conseguir que una fracción $1-p$ de sus tramas lleguen correctamente a su destino en un número de intentos de transmisión menor que el rechazo selectivo ideal.

Si la comparamos con la cadencia eficaz de la estrategia de rechazo simple, podemos observar que la diferencia entre ambas será pequeña cuando lo sea el factor $pT_{\text{as}}C$ frente a n, es decir, si el tiempo de asentimiento es pequeño en comparación con el tiempo de transmisión de las tramas, o si sólo una muy pequeña fracción p de tramas se reciben con errores. En el primer caso el tiempo desperdiciado cada vez que el emisor debe retroceder es de poca magnitud, mientras que en el segundo caso la frecuencia con la que el receptor se ve obligado a retransmitir es baja.

8.2.4. Tamaño óptimo de trama

Las fórmulas a las que hemos llegado para la cadencia eficaz en los tres casos son funciones convexas de n, la longitud del campo de datos de la trama. Considerando a n como una variable continua, el valor óptimo n^* que hace máxima la cadencia eficaz es la solución única de la ecuación

$$\left.\frac{\mathrm{d}C_e(n)}{\mathrm{d}n}\right|_{n=n^*} = 0.$$

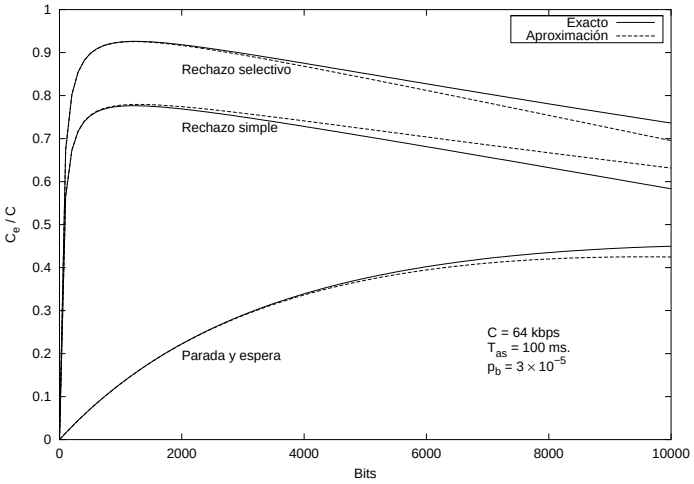


Figura 8.8. Curvas de cadencia eficaz, según estrategia ARQ.

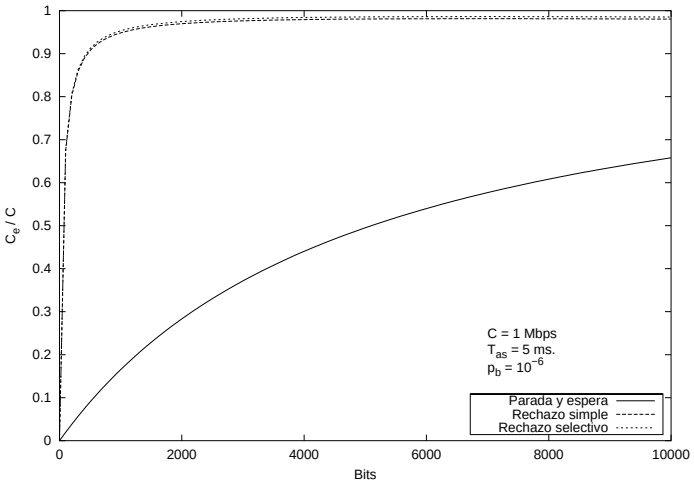


Figura 8.9. Curvas de cadencia eficaz, según estrategia ARQ.

La expresión más simple de la longitud óptima del campo de datos se obtiene para la cadencia eficaz en rechazo selectivo. Tras algunos cálculos sencillos se obtiene que

$$n^*|_{\text{R. Selectivo}} = \sqrt{\frac{m}{p_b}} - m$$

donde p_b es la probabilidad de error de bit.

8.2.5. Número medio de transmisiones por trama

En el análisis de la cadencia eficaz de las estrategias ARQ aparece continuamente como una variable el número n_t de transmisiones necesarias para hacer llegar una trama sin error al destino. La variable n_t es una variable aleatoria que, suponiendo errores de transmisión estadísticamente independientes en cada intento con probabilidad p, tiene como función de masas de probabilidad a

$$\mathrm{P}(n_t = k) = p^{k-1}(1-p), \quad \forall\, k \geq 1.$$

La esperanza matemática de n_t representa el número medio de intentos requeridos por trama, y puesto que n_t es una variable no negativa, vale

$$\mathrm{E}[n_t] = \sum_{k=1}^{\infty} \mathrm{P}\left(n_t \geq k\right) = \sum_{k=1}^{\infty} p^{k-1} = \frac{1}{1-p}.$$

El número medio de retransmisiones por cada trama es simplemente

$$\mathrm{E}[n_t] - 1 = \frac{p}{1-p}.$$

Notas bibliográficas

Son pocos los libros que tratan simultáneamente la teoría de la codificación y las estrategias de retransmisión automática. Entre ellos cabe citar [39] y [75]. Bien al contrario, prácticamente cualquier libro sobre redes de ordenadores incluye una exposición de las técnicas ARQ, y la mayoría realiza análisis de prestaciones similares al de este capítulo. Dentro de este grupo es particularmente recomendable el texto de Bertsekas y Gallager [5]. Por otra parte, las estrategias de retransmisión son sólo un aspecto de los protocolos de enlace de datos. Se pueden consultar detalles prácticos de especificación e implementación de estos protocolos en [6].

La verificación y validación formal de protocolos es una disciplina independiente, bien desarrollada y que requiere otro tipo de conceptos. Dos buenas obras especializadas en este campo son [29] y [33].

La combinación de códigos correctores de errores y de técnicas de retransmisión (ARQ híbrida) es una solución que este capítulo no explora, pero a la que sin embargo otros autores prestan una atención específica. Véase [75] para una exposición rigurosa del tema.

8.A. Sobre la cadencia eficaz

En este apéndice se analiza más detenidamente la definición del concepto de cadencia eficaz.

Considérese una sucesión arbitrariamente larga de tramas, de longitudes $n_1, n_2, \ldots$, y sea t_{oc_i} el tiempo efectivo de transmisión sin error de la i-ésima trama. Si todas estas cantidades fuesen deterministas, entonces el límite

$$\lim_{k\to\infty} \frac{n_1 + \cdots + n_k}{t_{\mathrm{oc}_1} + \cdots + t_{\mathrm{oc}_k}}$$

representaría el régimen efectivo de transmisión por el canal, promediado tras un tiempo de observación arbitrariamente largo. Ahora bien, en general, las secuencias $\mathcal{N} = (n_1, n_2, \ldots)$ y $\mathcal{T}_{\mathrm{oc}} = (t_{\mathrm{oc}_1}, t_{\mathrm{oc}_2}, \ldots)$ son dos procesos estocásticos, pues ni hay motivo para suponer que las longitudes de las tramas son conocidas de antemano ni los tiempos de ocupación son constantes incluso si las tramas tienen una longitud fija. Si se suponen las longitudes de las distintas tramas y los tiempos de ocupación de las mismas estadísticamente independientes e idénticamente distribuidos (iid), entonces tanto $\mathcal{N}$ como $\mathcal{T}_{\mathrm{oc}}$ son dos procesos estacionarios en sentido estricto, lo que significa que la función de distribución conjunta de $n_{j_1}, \ldots, n_{j_k}$ (análogamente la de $t_{\mathrm{oc}_{j_1}}, \ldots, t_{\mathrm{oc}_{j_k}}$), para cualesquiera $j_1 < j_2 < \cdots < j_k$ y cualquier valor de k, no depende ni del conjunto de subíndices temporales ni del orden en que éstos sean dados, sino sólo de k. Observe que t_{oc_i} no será, en general, independiente de n_i, ya que las tramas más largas tienden a ocupar por más tiempo el canal.

En estas condiciones, el teorema ergódico[7] para procesos estacionarios en sentido estricto permite escribir

$$\lim_{k\to\infty} \frac{n_1 + n_2 + \cdots + n_k}{k} = \mathrm{E}[n] \quad \text{con prob. 1}$$
$$\lim_{k\to\infty} \frac{t_{\mathrm{oc}_1} + t_{\mathrm{oc}_2} + \cdots + t_{\mathrm{oc}_k}}{k} = \mathrm{E}\left[t_{\mathrm{oc}}\right] \quad \text{con prob. 1.}$$

Los límites anteriores significan que los promedios temporales de las longitudes de las tramas y de los tiempos de ocupación convergen, para cualquier

[7] El teorema ergódico es una importante generalización de la ley de los grandes números y afirma que, en ciertas condiciones, la media temporal de un proceso estocástico estacionario (en sentido estricto o estacionario de segundo orden) converge, en un sentido matemático bien definido, a una variable aleatoria. Según el tipo de convergencia se tienen distintas versiones del teorema ergódico: J. von Neumann dio la prueba para la convergencia en media entre variables aleatorias, mientras que para el caso, notoriamente más difícil, de la convergencia con probabilidad 1, la demostración es un logro de D. Birkhoff. Veáse el capítulo 9 de G. Grimmet, D. Stirzaker, *Probability and random processes*, Oxford University Press, 3ª edición, 2001, para una formulación y una demostración rigurosas del teorema ergódico.

posible realización excepto un conjunto de trayectorias de probabilidad nula, al valor medio estadístico de longitud de una trama y del tiempo de ocupación, respectivamente. Por tanto,

$$\lim_{k\to\infty} \frac{n_1 + \cdots + n_k}{t_{\mathrm{oc}_1} + \cdots + t_{\mathrm{oc}_k}} = \lim_{k\to\infty} \frac{k^{-1}(n_1 + \cdots + n_k)}{k^{-1}(t_{\mathrm{oc}_1} + \cdots + t_{\mathrm{oc}_k})} = \frac{\mathrm{E}[n]}{\mathrm{E}[t_{\mathrm{oc}}]}$$

con probabilidad 1, puesto que los límites del numerador y del denominador existen con probabilidad 1 y son constantes.

PARTE III

Códigos BCH, códigos Reed–Solomon y códigos convolucionales

CAPÍTULO 9

Cuerpos finitos

Los cuerpos finitos son uno de los entes matemáticos en los que se sustentan la construcción y los procedimientos de decodificación de los códigos algebraicos. Este capítulo se ocupa de estudiar qué estructura tiene esta clase de objetos, cómo se representan sus elementos y qué propiedades poseen.

9.1. Definición

Un cuerpo es cualquier conjunto abstracto en el que hay una aritmética bien definida entre sus elementos. Siendo más precisos:

DEFINICIÓN 9.1. *Un cuerpo es una estructura algebraica formada por un conjunto de elementos $\mathcal{C}$ y dos operaciones binarias internas, que se simbolizan aquí con los signos $+$ y $\cdot$, tales que*

a) $(\mathcal{C}, +)$ *es un grupo conmutativo (la operación $+$ es asociativa, conmutativa, existe para ella un elemento neutro, que se representa por* 0*, y todo elemento de $\mathcal{C}$ tiene simétrico).*

b) $(\mathcal{C}, \cdot)$ *es un grupo; el elemento neutro o* unidad *se suele representar en este caso como* 1*.*

c) La operación $\cdot$ es distributiva por la izquierda y por la derecha respecto de la operación $+$.

Un cuerpo es conmutativo (o abeliano) si lo es su operación $\cdot$, o sea, si $(\mathcal{C}, \cdot)$ es un grupo conmutativo.

Se utilizan los símbolos + y · para las dos operaciones, y se suelen designar, respectivamente, como *suma* y *producto*, porque son una abstracción de las operaciones de suma y producto de los conjuntos numéricos. La estructura algebraica de cuerpo no es sino la formalización de un conjunto en el que las operaciones de suma y producto y sus inversas, resta y división, están bien definidas, salvo la división por cero.

Como consecuencia inmediata de la definición se tienen, para dos elementos cualesquiera a y b de un cuerpo, las siguientes propiedades, que son sencillamente las reglas aritméticas usuales:

a) $a \cdot b = 0 \Leftrightarrow a = 0$ o $b = 0$.

b) Regla de los signos: $-(ab) = (-a)b = a(-b)$; $(-a)(-b) = ab$.

c) Todo elemento no nulo es simplificable para el producto: si $a \neq 0$, $ab = ac \Rightarrow b = c$. (Consecuencia de la propiedad 1.)

Son ejemplos bien conocidos de cuerpos el conjunto de los números racionales, el conjunto de los números reales y el conjunto de los números complejos, todos ellos con un número infinito de elementos. Pero en la construcción de códigos se manejan conjuntos (alfabetos) finitos, y por ello las estructuras algebraicas que surgen constan siempre de un número finito de elementos.

Definición 9.2. *Un cuerpo con un número finito de elementos se llama cuerpo finito o cuerpo de Galois.*

Definición 9.3. *El número de elementos de un cuerpo finito es el cardinal del cuerpo.*

También es habitual el término *orden* para referirse al número de elementos de un cuerpo.

Los cuerpos no conmutativos poseen una estructura más complicada que los conmutativos. Por fortuna, todo cuerpo finito es conmutativo, proposición que constituye uno de los llamados teoremas de estructura de Wedderburn.

Ejemplo 9.1 (El cuerpo GF(p)). En el conjunto $\mathbb{Z}$ de los números enteros, la relación de congruencia módulo $p \in \mathbb{N}$, $p > 1$, según la cual dos números enteros a y b se dicen congruentes si ambos producen el mismo resto al ser divididos entre p (o lo que es igual, si su diferencia $a - b$ es un múltiplo de p), es una relación de equivalencia. El conjunto cociente

que induce esta relación se simboliza por $\mathbb{Z}/p\mathbb{Z}$, o también por $\mathbb{Z}_p$, y está compuesto por las p clases de números enteros

$$[r] = r + p\mathbb{Z} = \{r + pn : n \in \mathbb{Z}\}, \quad 0 \leq r \leq p-1$$

a las que se conoce por clases residuales (o de residuos) módulo p. El signo $[r]$ indica la clase de representante r.

En el conjunto de representantes de $\mathbb{Z}/p\mathbb{Z}$,

$$\{0, 1, \ldots, p-1\}$$

considérense las operaciones de suma y producto definidas por las siguientes identidades

$$\begin{aligned} &a + b \equiv (a + b) \mod p \\ &ab \equiv (ab) \mod p, \quad \forall a, b \in \{0, 1, \ldots, p-1\} \end{aligned}$$

en donde la notación $a \equiv b \mod p$ se lee «a es congruente con b módulo p». Es decir, suma y producto en $\mathbb{Z}/p\mathbb{Z}$ se llevan a cabo considerando a los operandos números enteros ordinarios y reduciendo el resultado módulo p.

No es difícil probar que, si p es un número primo, entonces el conjunto $(\mathbb{Z}/p\mathbb{Z}, +, \cdot)$ de residuos módulo p es efectivamente un cuerpo. La única propiedad que no es trivial verificar es la de existencia de inverso multiplicativo de cualquier elemento no nulo. Pero sea $a \neq 0$ (y también $a \neq 1$, puesto que 1 es obviamente el inverso de sí mismo) uno de tales elementos, primo relativo de p, por tanto. Existen, entonces, en virtud del algoritmo de división euclídeo, dos números enteros b y q no nulos tales que $ab + pq = 1$. Y, en consecuencia, $ab + pq \equiv ab \equiv 1 \mod p$, lo que muestra que $a^{-1} \equiv b \mod p$ es el inverso de a. En el resto del capítulo se utilizará el símbolo $\mathrm{GF}(p)$ para referirse a este cuerpo de p elementos (p primo).

Por el contrario, si p no es primo, $(\mathbb{Z}/p\mathbb{Z}, +, \cdot)$ no es un cuerpo, pues $p = rs \equiv 0 \mod p$, con $r \neq 0$ y $s \neq 0$ por hipótesis.

Los cuerpos de cardinal pequeño se pueden definir o presentar de forma explícita indicando en un par de tablas, llamadas *tablas de Cayley*, el resultado de operar aditiva o multiplicativamente con dos elementos cualesquiera suyos. Por ejemplo, $\mathrm{GF}(2) = \{0, 1\}$ con

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

y $\mathrm{GF}(3) = \{0, 1, 2\}$ con

$+$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

$\cdot$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

En breve se verán formas mucho más compactas de describir los cuerpos finitos. ■

A la vista del ejemplo anterior, se presenta de manera natural la siguiente cuestión: ¿existen cuerpos finitos cuyo cardinal no sea un número primo? Y, si existen, ¿cuáles son y cómo se construyen?

9.2. Construcción de cuerpos finitos

Se verá más adelante en este mismo capítulo que sólo existen cuerpos finitos de cardinal p^m, con p un número primo y m un número natural, y además, que hay, salvo isomorfismos, un solo cuerpo de cardinal p^m, por lo que tiene perfecto sentido escribirlo como $\mathrm{GF}(p^m)$.

Entre tanto, mostraremos una manera de construir $\mathrm{GF}(p^m)$ cuando $m > 1$. La idea central del método que se expone en el teorema 9.1 es la de construir este cuerpo tomando como elementos suyos las clases de equivalencia de un cierto dominio euclídeo,[1] en este caso el dominio formado por el conjunto de polinomios de una variable con coeficientes en $\mathrm{GF}(p)$.

Definición 9.4. *En el conjunto $F_p[x]$ de polinomios con coeficientes del cuerpo $\mathrm{GF}(p)$, dos polinomios se dicen congruentes módulo $\pi(x) \in F_p[x]$ si tienen el mismo resto al ser divididos por $\pi(x)$.*

La relación de congruencia módulo $\pi(x)$ es una equivalencia compatible con la suma y el producto ordinario de polinomios en $F_p[x]$. Su conjunto cociente se representa por $F_p[x]/\pi(x)$, y es el conjunto de polinomios de grado menor que el de $\pi(x)$, con coeficientes tomados del cuerpo $\mathrm{GF}(p)$, en el que las operaciones de suma y producto de polinomios se realizan módulo $\pi(x)$.

[1] Formalmente, $(E, +, \cdot)$ es un dominio euclídeo si

a) $(E, +, \cdot)$ es un dominio de integridad (un anillo conmutativo sin divisores de cero).

b) Existe una función de orden $g(\cdot)$, con valores enteros no negativos, tal que $g(a) \leq g(ab)$ si $b \neq 0$; y tal que, para cualesquiera $a, b \neq 0$, existen $q, r \in E$ de modo que o bien $a = qb$, o bien $a = qb + r$ con $g(r) < g(b)$.

Son ejemplos inmediatos de dominios euclídeos el conjunto $\mathbb{Z}$ con $g(n) = |n|$, o bien el conjunto de los polinomios con coeficientes en un cuerpo y con $g(f(x)) = \mathrm{grado}(f(x))$.

Teorema 9.1. *$F_p[x]/\pi(x)$ es un cuerpo de cardinal p^m si y solamente si $\pi(x) \in F_p[x]$ es un polinomio irreducible de grado m.*

Demostración. Un polinomio con coeficientes de GF(p) se dice *irreducible* en $F_p[x]$ cuando no es posible escribirlo como producto de otros dos polinomios de $F_p[x]$ de menor grado.[2] En el teorema 9.23 se prueba que, para todo número primo p y todo número natural m, existen en $F_p[x]$ polinomios irreducibles de grado m. Por el momento, dése por cierta esta proposición.

Una vez precisado este punto, la demostración del teorema se reduce a verificar cada uno de los axiomas estipulados en la definición de cuerpo.

Así, la suma de dos polinomios de $F_p[x]$ de grado menor que m es una operación interna asociativa, conmutativa, con elemento neutro y tal que todo polinomio posee opuesto. Si ahora, como se indica en el enunciado, se define el producto de dos polinomios como

$$a(x)b(x) \equiv (a(x)b(x)) \mod \pi(x)$$

se tiene en $F_p[x]/\pi(x)$ una operación interna asociativa, conmutativa y con elemento neutro $e(x) = 1$. Además, este producto es distributivo respecto de la suma de polinomios, es decir, $F_p[x]/\pi(x)$ es un anillo unitario y conmutativo. De manera que, para probar que el conjunto así formado es en efecto un cuerpo con las dos operaciones consideradas, sólo hay que asegurarse de que todo polinomio no nulo $a(x)$ de grado $m-1$ o menor posee inverso para la operación producto. Pero, por el algoritmo de división euclídeo, y como $\pi(x)$ es irreducible, y por lo tanto $a(x) \neq 1$ es primo relativo suyo, se tiene garantizada la existencia de dos polinomios $b(x)$ y $q(x)$ no nulos tales que

$$a(x)b(x) + \pi(x)q(x) = 1.$$

De tal forma que $a(x)b(x) \equiv 1 \mod \pi(x)$ y $a^{-1}(x) \equiv b(x) \mod \pi(x)$ es el inverso de $a(x)$. Adviértase que es esencial para la existencia de inverso multiplicativo que $\pi(x)$ sea irreducible.

El número de elementos del cuerpo es igual al número de polinomios de grado menor que m con coeficientes de GF(p), es decir, p^m. ▶

Más en general, el razonamiento seguido en la demostración podría haberse aplicado igualmente a cualquier dominio euclídeo abstracto D y a cualquier elemento primo π en él, de suerte que D/π sería un cuerpo.

Para referirse a un elemento $a(x) \in F_p[x]/\pi(x)$ basta entonces con conocer la secuencia ordenada de m coeficientes del polinomio $a(x)$. Ahora bien, el conjunto de secuencias de m elementos de GF(p) es un espacio vectorial de dimensión m sobre el cuerpo GF(p) y la aplicación que a cada

[2] Un polinomio irreducible en $F_p[x]$ puede no serlo en $F_q[x]$, $p \neq q$.

$a(x) \in F_p[x]/\pi(x)$ le asigna la m-tupla de coeficientes de $a(x)$ es un isomorfismo entre $F_p[x]/\pi(x)$ y $\mathrm{GF}(p)^m$. Por tanto, se infiere que $F_p[x]/\pi(x)$ es, para la operación de suma, un espacio vectorial m-dimensional sobre el cuerpo de escalares $\mathrm{GF}(p)$. Más adelante resultará conveniente recurrir a la caracterización de un cuerpo finito como espacio vectorial para deducir ciertas propiedades con mayor comodidad.

EJEMPLO 9.2. El polinomio $\pi(x) = x^4 + x + 1$ es irreducible en $F_2[x]$ (compruébese como ejercicio). En virtud del teorema 9.1, el cuerpo $\mathrm{GF}(2^4)$ es, por tanto, isomorfo a $F_2[x]/\pi(x)$, el conjunto de polinomios binarios de grado menor que 4 con las operaciones de suma y producto módulo $x^4 + x + 1$.

Si se representa por α el elemento $x \in F_2[x]/\pi(x)$, entonces, por definición, $\pi(\alpha) = \alpha^4 + \alpha + 1 \mod \pi(x) = 0$, o lo que es igual, $\alpha^4 \equiv \alpha + 1 \mod \pi(x)$. Pues bien, se da el caso de que es posible representar cualquier elemento de $F_2[x]/\pi(x)$ como una potencia de α. Por ejemplo,

$$x^3 + x \equiv x^9 \mod x^4 + x + 1 = \alpha^9.$$

Aplicando reiteradamente la identidad $\alpha^4 = \alpha + 1$ (módulo $\pi(x)$), y calculando las sucesivas potencias de α, se obtiene el conjunto de elementos de $F_2[x]/\pi(x)$, recogidos en la tabla 9.1.

Existen, pues, dos maneras de representar los elementos del cuerpo, una aditiva y otra multiplicativa. En la tabla 9.1, la primera columna indica la secuencia de coeficientes de la representación polinómica, con el coeficiente del término de menor exponente situado más a la derecha, y la columna central es el polinomio correspondiente. Como se puede ver, para sumar dos elementos cualesquiera es suficiente sumar sus polinomios respectivos (¡siempre en $F_2[x]$!) o bien sumar elemento a elemento sus vectores de coeficientes asociados.

La última columna de la tabla contiene la representación de los elementos del cuerpo en términos de potencias de α. El método más directo para multiplicar dos de ellos consiste en representar ambos factores con esta notación y operar con los exponentes. Por ejemplo, es así que

$$(x^2 + x)(x^3 + 1) \equiv \alpha^5\alpha^{14} = \alpha^{15}\alpha^4 \equiv x + 1 \mod \pi(x)$$

ya que $\alpha^{15} \equiv 1 \mod \pi(x)$.

En el apartado siguiente se analizarán las estructuras aditiva y multiplicativa de un cuerpo finito. Se mostrará, en particular, que en todo cuerpo finito existe al menos un elemento, llamado primitivo o generador, tal que todos los demás excepto el cero son una potencia suya. Para $F_2[x]/x^4 + x + 1$, el lector debe convencerse, examinando la tabla 9.1, de que α^2, α^4 y α^8 también son elementos primitivos, además de α. Aunque todos ellos han

Vector	Polinomio	Elemento
0000	0	0
0001	1	1
0010	x	α
0100	x^2	α^2
1000	x^3	α^3
0011	$x+1$	α^4
0110	x^2+x	α^5
1100	x^3+x^2	α^6
1011	x^3+x+1	α^7
0101	x^2+1	α^8
1010	x^3+x	α^9
0111	x^2+x+1	α^{10}
1110	x^3+x^2+x	α^{11}
1111	x^3+x^2+x+1	α^{12}
1101	x^3+x^2+1	α^{13}
1001	x^3+1	α^{14}

Tabla 9.1. El cuerpo $F_2[x]/\pi(x)$, con $\pi(x) = x^4 + x + 1$.

resultado ser, en este caso, raíces del polinomio $\pi(x)$ que se ha utilizado para construir el cuerpo, no es ésta una propiedad general. ■

9.3. Estructura

Los cuerpos finitos poseen realmente una estructura bastante simple. Para establecer cuáles son sus propiedades más básicas, comencemos por identificar qué cuerpos contienen en su seno a otros más pequeños.

Definición 9.5. *Se dice de un cuerpo $\mathcal{E}$ que contiene a otro $\mathcal{C}$ que es una extensión de $\mathcal{C}$.*

Por ejemplo, el cuerpo complejo es una extensión del cuerpo real, el cual, a su vez, es una extensión del cuerpo racional. Y todo cuerpo de cardinal p^m es una extensión de GF(p), como se verá a continuación.

Considérese un cuerpo finito arbitrario, $\mathcal{C}$, de cardinal q. $\mathcal{C}$ contiene al elemento unidad 1. Contiene también a la sucesión de elementos $1+1=2$, $1+1+1=3$, ...; pero, por ser el cuerpo finito, éstos no podrán ser todos distintos entre sí. En consecuencia, debe existir el menor entero $p \leq q$ tal que $p = 1+1+\cdots+1(p \text{ veces}) = 1_p = 0$, y además este número p debe ser

primo. Pues si p no fuese primo, entonces $p = rs = 0$ y, por la definición de cuerpo, o bien $r = 0$ o bien $s = 0$, en contradicción con la premisa de ser p mínimo. El número p es la *característica* del cuerpo y es evidente que si $\mathcal{C}$ tiene característica p entonces $p\alpha = 0,\ \forall \alpha \in \mathcal{C}$.[3]

Pero, además, el producto o la suma de dos de los elementos $1,\ 1+1,\ \ldots,\ 1_p = 0$ es un elemento de la misma forma, ya que

$$1_j \cdot 1_k = 1_{(jk)} \mod p$$
$$1_j + 1_k = 1_{(j+k)} \mod p$$

lo que significa que los p elementos

$$0, 1, 1+1 = 2, \ldots, 1+\cdots+1 = p-1$$

son un subcuerpo de $\mathcal{C}$; esto es, se trata de un conjunto de elementos cerrado para la suma y el producto. Se concluye, por tanto, que todo cuerpo finito es una extensión de un cuerpo de cardinal igual a su característica, siendo éste el menor cuerpo contenido en él. Si $q = p$, entonces $\mathcal{C}$ contiene p elementos y $\mathcal{C} = \mathrm{GF}(p)$ es un *cuerpo primo*, en el sentido de que no contiene subcuerpos. Es obvio que el cuerpo primo de un cuerpo de característica p es isomorfo a $\mathrm{GF}(p)$.

Si $q > p$, sea m el número máximo de elementos $\alpha_0, \alpha_1, \ldots, \alpha_{m-1}$ del cuerpo linealmente independientes sobre los escalares de $\mathrm{GF}(p)$. Entonces $\mathcal{C}$ contiene todos los elementos de la forma

$$e_0\alpha_0 + e_1\alpha_1 + \cdots + e_{m-1}\alpha_{m-1}, \quad e_j \in \mathrm{GF}(p)$$

y, recíprocamente, todo elemento de $\mathcal{C}$ se puede escribir de forma única como combinación lineal de $\alpha_0, \ldots, \alpha_{m-1}$; porque si se pudiese expresar algún elemento η de dos formas diferentes

$$\begin{aligned}\eta &= e_0\alpha_0 + e_1\alpha_1 + \cdots + e_{m-1}\alpha_{m-1} \\ &= f_0\alpha_0 + f_1\alpha_1 + \cdots + f_{m-1}\alpha_{m-1}, \quad e_i, f_i \in \mathrm{GF}(p)\end{aligned}$$

entonces se tendría que

$$0 = (e_0 - f_0)\alpha_0 + (e_1 - f_1)\alpha_1 + \cdots + (e_{m-1} - f_{m-1})\alpha_{m-1}$$

y los elementos $\alpha_0,\ \ldots,\ \alpha_{m-1}$ no serían linealmente independientes sobre $\mathrm{GF}(p)$, en contra de lo supuesto. Por consiguiente, $\mathcal{C}$ es isomorfo a un espacio vectorial de dimensión m sobre $\mathrm{GF}(p)$ y contiene p^m elementos, para algún $m > 1$.

[3] En un cuerpo infinito, si $n \cdot 1 \neq 0$ para todo $n \neq 0$, se dice que el cuerpo tiene característica infinita o, también, característica cero (pues $n \cdot 1 = 0 \Leftrightarrow n = 0$). Conviene señalar que existen cuerpos infinitos con característica finita, por ejemplo $\mathrm{GF}(p)^\infty$, las sucesiones en $\mathrm{GF}(p)$.

Teorema 9.2. *El cardinal de un cuerpo finito es p^m, para algún p primo y algún $m \geq 1$.*

A veces puede ser útil pensar en este teorema como en un aserto elemental de inexistencia; nos permite afirmar, por ejemplo, que no es posible tener un cuerpo con 6 elementos.

Puesto que $\mathrm{GF}(p^m)$ es un espacio vectorial, se dispone de inmediato de la siguiente caracterización de su grupo aditivo, que muestra que la estructura aditiva de un cuerpo finito es esencialmente la de su subcuerpo primo.

Teorema 9.3 (Estructura aditiva de un cuerpo finito). *$(\mathcal{C}, +)$, el grupo aditivo de un cuerpo $\mathcal{C}$ de cardinal p^m (p primo), es isomorfo al grupo $(\mathbb{Z}/p\mathbb{Z}, +) \times \cdots \times (\mathbb{Z}/p\mathbb{Z}, +)$, el producto directo de orden m del grupo $(\mathbb{Z}/p\mathbb{Z}, +)$.*

Demostración. Sea $B = \{\alpha_1, \alpha_2, \ldots, \alpha_m\}$ un conjunto máximo de elementos de $\mathcal{C}$ linealmente independientes sobre $\mathrm{GF}(p)$. La aplicación

$$\phi : \ (\mathcal{C}, +) \to (\mathbb{Z}/p\mathbb{Z}, +) \times \cdots \times (\mathbb{Z}/p\mathbb{Z}, +)$$

que a cada $\beta \in \mathcal{C}$ le hace corresponder sus coordenadas en la base B es claramente un isomorfismo de grupos. ▶

Analicemos ahora la estructura de un cuerpo finito en relación con la operación producto. Sea α un elemento no nulo de $\mathcal{C}$. Los elementos α, $\alpha^2 = \alpha \cdot \alpha$, $\alpha^3 = \alpha \cdot \alpha \cdot \alpha$, ... son todos no nulos y, de nuevo, como $\mathcal{C}$ consta de un número finito de elementos, debe existir un entero $r < q$ para el que $\alpha^r = 1$, de tal forma que la secuencia $1, \alpha, \alpha^2, \ldots$ es periódica de periodo r. Es fácil comprobar que el primer elemento que se repite en la sucesión $1, \alpha, \alpha^2, \ldots$ es en verdad $\alpha^r = 1$, ya que si para algún $j \neq 0$ fuese $\alpha^i = \alpha^j$ entonces $\alpha^{i-j} = 1$, e $i - j < i$.

El menor índice r tal que $\alpha^r = 1$ se llama *orden* del elemento α y es obvio que el conjunto

$$\{1, \alpha, \alpha^2, \ldots, \alpha^{r-1}\}$$

compuesto por todas las potencias distintas de α es un subgrupo multiplicativo de $\mathcal{C}^*$. Si se conoce el orden de un elemento α de un grupo multiplicativo, se puede obtener de inmediato el orden de cualquiera de sus potencias.

Lema 9.4. *Si α es un elemento de orden r de un grupo multiplicativo, el orden de α^i es*

$$\frac{r}{(i, r)}$$

en donde (i, r) representa el máximo común divisor de i, r.

DEMOSTRACIÓN. Es fácil ver que

$$\left(\alpha^i\right)^{\frac{r}{(i,r)}} = (\alpha^r)^{\frac{i}{(i,r)}} = 1.$$

Sea x el orden de α^i, $(\alpha^i)^x = 1$. Entonces r divide a ix, lo que implica que $r/(i,r)$ divide a x. Pero, por la identidad anterior, x divide a $r/(i,r)$. Por lo tanto, el orden de α es $r/(i,r)$. ▶

TEOREMA 9.5. *Si β es un elemento no nulo de un cuerpo de cardinal q, entonces $\beta^{q-1} = 1$.*

DEMOSTRACIÓN. Sean $\alpha_1, \alpha_2, \ldots, \alpha_{q-1}$ todos los elementos del cuerpo distintos de 0. Entonces

$$(\beta\alpha_1)(\beta\alpha_2)\cdots(\beta\alpha_{q-1}) = \beta^{q-1}\alpha_1\alpha_2\cdots\alpha_{q-1} = \alpha_1\alpha_2\cdots\alpha_{q-1}$$

puesto que $\{\beta\alpha_1, \ldots, \beta\alpha_{q-1}\}$ es una permutación de $\{\alpha_1, \ldots, \alpha_{q-1}\}$. Por lo tanto, $\beta^{q-1} = 1$ o, lo que es igual, β es raíz del polinomio $x^{q-1} - 1$. ▶

TEOREMA 9.6. *El orden de un elemento no nulo de* GF(q) *es un divisor de* $q - 1$.

DEMOSTRACIÓN. Este resultado es elemental: vea que, por definición, si $\beta \in \mathrm{GF}(q)$ tiene orden r, entonces $\beta^n = 1$ si y solamente si r divide a n; ya que de no cumplirse esto, se podría escribir que $n = rs + t$ con $t \neq 0$, y entonces $\beta^n = \beta^t\beta^{sr} = \beta^t = 1$ con $t < r$, lo que no es posible. En particular, como según el teorema anterior $\beta^{q-1} = 1$, resulta que r es un divisor de $q-1$.

Por otro lado, si $\beta \in \mathrm{GF}(q)$ es un elemento de orden r, el conjunto $\{1, \beta, \ldots, \beta^{r-1}\}$ es un subgrupo multiplicativo de r elementos del grupo $\mathrm{GF}(q) \setminus \{0\}$. En virtud del teorema de Lagrange,[4] r divide a $q - 1$. ▶

Por tanto, como consecuencia de este teorema y del lema 9.4, se tiene que, en un cuerpo de característica q, los elementos γ y γ^q tienen igual orden. Si en un cuerpo de cardinal q existiese un elemento α de orden $q-1$, entonces al subgrupo multiplicativo

$$F_q^* = \{\alpha, \alpha^2, \ldots, \alpha^{q-1} = 1\}$$

pertenecerían todos los elementos no nulos del cuerpo, y se podría escribir que $\mathrm{GF}(q) = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\}$. Cuando en un grupo finito existe algún

[4] El teorema de Lagrange establece que cualquier subgrupo $(H, *)$ de un grupo $(G, *)$ posee un número de elementos que divide al número de elementos de G. Para probar tal cosa, basta con definir los cogrupos $a * H$, $a \in G$, que tienen cada uno tantos elementos como H y forman una partición de G.

elemento β tal que todos los demás son, con notación multiplicativa, una potencia suya, se llama a β elemento generador del grupo, y se dice de éste que es un grupo cíclico. Un elemento de un cuerpo con la propiedad de que sus potencias engendran todos los demás elementos distintos de cero recibe el nombre de *elemento primitivo* (véase de nuevo el ejemplo 9.2). Pues bien, en todo cuerpo finito hay al menos un elemento primitivo; o, por expresarlo de otra forma, $(\mathcal{C}^*, \cdot)$, el conjunto de elementos del cuerpo distintos de cero, es siempre un grupo multiplicativo cíclico.

Teorema 9.7 (Estructura multiplicativa de los cuerpos finitos). *En todo cuerpo finito existe un elemento primitivo.*

Demostración. Sea α un elemento de orden r máximo. Por supuesto, según se desprende del teorema 9.5, es $r \leq q-1$. Sea β otro elemento de orden $s < r$, y, para cualquier número primo p, supóngase que $r = p^a r'$ y que $s = p^b s'$, con r' y s' no divisibles por p. Así, α^{p^a} es un elemento de orden r', $\beta^{s'}$ es un elemento de orden p^b y, por ser r' y p^b primos entre sí, $\alpha^{p^a}\beta^{s'}$ es un elemento de orden $p^b r'$. Luego $b \leq a$, o de otro modo r no sería máximo. En suma, cualquier factor primo que divida a s también debe dividir a r, y en consecuencia s es un divisor de r, de donde se sigue que el orden de cualquier elemento no nulo de un cuerpo divide siempre al mayor de los órdenes de todos los elementos.

Como cualquier elemento no nulo es, por todo lo anterior, raíz del polinomio $x^r - 1$, este polinomio es divisible por

$$\prod_{\beta \in \mathrm{GF}(q)\setminus\{0\}} (x - \beta).$$

Dado que $\mathrm{GF}(q)$ contiene $q-1$ elementos no nulos, necesariamente $r \geq q-1$. Combinando esta desigualdad con la que afirma que $r \leq q-1$, se concluye finalmente que $r = q-1$, lo que significa que α es un elemento primitivo. ▶

Aplicando el lema 9.4, es un corolario inmediato verificar que si α es un elemento primitivo de un cuerpo de característica p, también son primitivos α^p, $\alpha^{p^2}, \ldots, \alpha^{p^{m-1}}$.

Así pues, el producto de dos elementos no nulos de un cuerpo de cardinal q se reduce al producto módulo-$(q-1)$ de sus exponentes.

Teorema 9.8 (Estructura multiplicativa de los cuerpos finitos). *$(F_q^*, \cdot)$, el grupo multiplicativo de los elementos no nulos del cuerpo* $\mathrm{GF}(q)$*, es isomorfo a* $(\mathbb{Z}_{q-1}, \cdot)$.

Demostración. Obviamente, la aplicación $\phi(\alpha^i) = i$, con α un elemento primitivo de $\mathrm{GF}(q)$, es un isomorfismo de grupos entre $(F_q^*, \cdot)$ y $(\mathbb{Z}_{q-1}, \cdot)$. ▶

Combinando el lema 9.4 con el teorema 9.7, se obtiene directamente el número de elementos de un orden dado en $\mathrm{GF}(q)$. Para un entero positivo t, sea $\varphi(t)$ la *función de Euler*, que se define como el número de enteros i en $\{1, 2, \ldots, t-1\}$ tales que $\mathrm{mcd}(i, t) = 1$.

TEOREMA 9.9.

a) *El número de elementos de orden t en $\mathrm{GF}(q)$ es 0 si t no es un divisor de $q-1$; o es $\varphi(t)$ si t divide a $q-1$.*

b) $\displaystyle\sum_{t|q-1} \varphi(t) = q - 1.$

DEMOSTRACIÓN.

a) Sea α un elemento de $\mathrm{GF}(q)$ de orden t. El orden de α^i es igual a t si y sólo si $(i, t) = 1$. El número de estos elementos es, por definición, $\varphi(t)$.

b) El número de elementos no nulos de $\mathrm{GF}(q)$ es precisamente

$$\sum_{t|q-1} \varphi(t).$$

La identidad $\sum_{t|q-1} \varphi(t) = q - 1$ es un caso particular de un teorema debido a Gauss que afirma que para cualquier entero m se cumple que $\sum_{t|m} \varphi(t) = m$. ▶

El teorema 9.7 también trae como consecuencia un hecho notable: las q raíces de la ecuación algebraica $x^q - x = 0$ son todas distintas y poseen estructura de cuerpo cuando q es potencia de un número primo.

COROLARIO 9.10 (TEOREMA DE FERMAT). *Todo elemento de un cuerpo de cardinal q es raíz de la ecuación $x^q = x$, y entonces*

$$x^q - x = \prod_{\alpha \in \mathrm{GF}(q)} (x - \alpha).$$

La próxima proposición indica cuáles son los posibles subcuerpos de un cuerpo dado. Antes de analizarla se precisa un resultado preparatorio.

LEMA 9.11. *El polinomio $x^a - 1$ divide a $x^b - 1$ si y solamente si a es un divisor de b.*

Demostración. La prueba de la condición suficiente es inmediata, ya que si $b = ac$,

$$x^b - 1 = x^{ac} - 1 = (x^a - 1)(x^{a(c-1)} + x^{a(c-2)} + \cdots + x^{2a} + x^a + 1).$$

Para probar que también es condición necesaria, supóngase que b no fuese múltiplo de a

$$b = qa + r, \qquad 0 < r < a.$$

Entonces

$$\frac{x^b - 1}{x^a - 1} = x^r \frac{x^{qa} - 1}{x^a - 1} + \frac{x^r - 1}{x^a - 1}.$$

Pero $x^a - 1$ siempre es factor de $x^{qa} - 1$, mientras que, en cambio, $x^a - 1$ nunca divide a $x^r - 1$ porque $r < a$. Y como, por hipótesis, $x^a - 1$ divide a $x^b - 1$, necesariamente ha de ocurrir que $r = 0$, es decir, a divide a b. ▶

Teorema 9.12 (Caracterización de subcuerpos).

a) El cardinal de un subcuerpo de $\mathrm{GF}(p^m)$ *es* p^s*, siendo* s *un divisor positivo de* m*. Recíprocamente, si* s *divide a* m*, entonces* $\mathrm{GF}(p^s) \subseteq \mathrm{GF}(p^m)$.

b) Además, si α *es un elemento primitivo de* $\mathrm{GF}(p^m)$*, entonces* $\gamma = \alpha^t$*,* $t = (p^m - 1)/(p^s - 1)$*, es un elemento primitivo de* $\mathrm{GF}(p^s)$*, y*

$$\mathrm{GF}(p^s) = \left\{ \beta \in \mathrm{GF}(p^m) \ : \ \beta^{p^s} = \beta \right\}.$$

Demostración.

a) Sea F un subcuerpo de $\mathrm{GF}(p^m)$. Es claro que F tiene característica p y debe tener cardinal p^s para algún $s \leq m$. Sin embargo, $\mathrm{GF}(p^m)$ es también un espacio vectorial sobre F, por lo que, si su dimensión es r, $p^m = (p^s)^r$ y s divide a m.

Si s es un divisor de m, entonces $p^s - 1$ divide a $p^m - 1$ y $x^{p^s-1} - 1$ divide a $x^{p^m-1} - 1$. Por consiguiente, $x^{p^s} - x$ es un divisor de $x^{p^m} - x$, y las raíces del primer polinomio lo son también del segundo. Pero, por el teorema de Fermat, las raíces de $x^{p^s} - x$ son un cuerpo isomorfo a $\mathrm{GF}(p^s)$ y las de $x^{p^m} - x$ son un cuerpo isomorfo a $\mathrm{GF}(p^m)$. Se concluye, en consecuencia, que $\mathrm{GF}(p^s) \subseteq \mathrm{GF}(p^m)$.

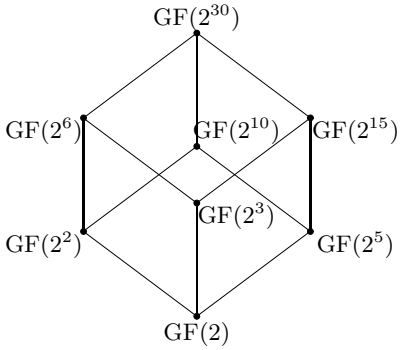
b) Por último, si α es un elemento primitivo de $\mathrm{GF}(p^m)$ y

$$t = \frac{p^m - 1}{p^s - 1}$$

es claro que $\gamma = \alpha^t$ es un elemento de orden $p^s - 1$ y que $\mathrm{GF}(p^s) = \{0, \gamma, \gamma^2, \ldots, \gamma^{p^s-2}\}$. ▶

Así, los subcuerpos contenidos en $\mathrm{GF}(p^m)$ podrán obtenerse sin más que enumerar todos los enteros positivos divisores de m. Y, en particular, el subcuerpo primo de un cuerpo finito de característica p es el subconjunto de elementos que verifican $\beta^p = \beta$.

EJEMPLO 9.3. Con frecuencia se representan esquemáticamente los subcuerpos de un cuerpo finito, por ejemplo de $\mathrm{GF}(2^{30})$, en un diagrama reticular como el siguiente, en el que los arcos y caminos del grafo simbolizan la inclusión. Las relaciones de inclusión mutuas entre estos cuerpos son las mismas que las relaciones de divisibilidad entre los factores enteros de 30.



■

Por último, he aquí un par de identidades que simplifican un sinnúmero de expresiones aritméticas en un cuerpo de característica finita. Ambas son una consecuencia de la estructura aditiva del mismo.

TEOREMA 9.13. *En un cuerpo conmutativo de característica p,*

$$(a+b)^p = a^p + b^p.$$

DEMOSTRACIÓN. En cualquier cuerpo conmutativo (finito o infinito),

$$(a+b)^p = \sum_{i=0}^{p} \binom{p}{i} a^i b^{p-i}$$

donde, por definición,

$$\binom{p}{0} = \binom{p}{p} = 1$$

y

$$\binom{p}{i} = \frac{p(p-1)\cdots(p-i+1)}{i(i-1)\cdots 1}, \quad i = 1, \ldots, p-1.$$

Pero $\binom{p}{i} \equiv 0 \mod p$ para $i = 1, \ldots, p-1$, porque siendo el numerador múltiplo de p y el denominador no, $\binom{p}{i}$ es múltiplo de p. ▶

Corolario 9.14. *Si $\alpha_1, \alpha_2, \ldots, \alpha_n$ son elementos del cuerpo* $\mathrm{GF}(p^m)$*, entonces, para cualquier $r \geq 0$,*

$$(\alpha_1 + \alpha_2 + \cdots + \alpha_n)^{p^r} = \alpha_1^{p^r} + \alpha_2^{p^r} + \cdots + \alpha_n^{p^r}.$$

Demostración. Por inducción en r y en n. ▶

La aplicación

$$\begin{aligned} \sigma_p : \mathrm{GF}(p^m) &\longrightarrow \mathrm{GF}(p^m) \\ \alpha &\longrightarrow \alpha^p \end{aligned}$$

es un automorfismo de $\mathrm{GF}(p^m)$, conocido como *automorfismo de Galois*. σ_p tiene orden m (σ_p^m es la aplicación identidad), y sus puntos fijos son los del subcuerpo primo $\mathrm{GF}(p)$. Tomando como producto la composición de aplicaciones, el conjunto de todos los automorfismos de un cuerpo de característica p forma un grupo cíclico generado por el automorfismo de Galois, ya que cualquier automorfismo transforma elementos primitivos en elementos primitivos. La acción del automorfismo de Galois se extiende de forma natural al anillo de los polinomios de $F_p[x]$, como se expone seguidamente.

Corolario 9.15. *Si $Q(x)$ es un polinomio con coeficientes en* $\mathrm{GF}(p)$*, con p primo, entonces*

$$\sigma_p\left(Q(x)\right) = \left(Q(x)\right)^p = Q(x^p)$$

en $\mathrm{GF}(p^m)$.

Demostración. Si

$$Q(x) = \sum_{i=0}^{n} q_i x^i, \qquad q_i \in \mathrm{GF}(p)$$

entonces, por el teorema 9.13,

$$Q^p(x) = (q_0 + \cdots + q_n x^n)^p = (q_0^p + q_1^p x^p + \cdots + q_n^p x^{np}).$$

Y como, según el teorema de Fermat, $q_i \in \mathrm{GF}(p) \Leftrightarrow q_i^p = q_i$, resulta

$$(Q(x))^p = q_0 + q_1 x^p + \cdots + q_n x^{np} = Q(x^p).$$ ▶

9.4. Polinomios mínimos

El teorema de Fermat afirma que cualquier elemento γ de un cuerpo de cardinal q y característica p es una raíz del polinomio $x^q - x \in F_p[x]$. Pero bien pudiera ocurrir que existiesen otros polinomios con coeficientes de GF(p) que no fuesen múltiplos de $x^q - x$ y que tuviesen también a γ como un cero.

Siendo $P(x)$ uno de ellos, como es evidente que si $P(\gamma) = 0$ entonces para cualquier escalar β distinto de cero es $\beta P(\gamma) = 0$, se puede suponer sin pérdida de generalidad que $P(x)$ es *mónico*, esto es, que el coeficiente de su término de mayor exponente vale 1.

Definición 9.6. *Sea $\mathcal{F}$ un subcuerpo del cuerpo $\mathcal{C}$. El polinomio mínimo $M_\gamma(x)$ de $\gamma \in \mathcal{C}$ sobre $\mathcal{F}$ es el polinomio mónico de menor grado con coeficientes de $\mathcal{F}$ que admite a γ como raíz, $M_\gamma(\gamma) = 0$.*

Ejemplo 9.4. La tabla siguiente da la lista de los polinomios mínimos de los elementos de $F_2[x]/x^4 + x + 1$ sobre $F_2[x]$; α es el elemento primitivo x.

Elemento	Polinomio mínimo	Elemento	Polinomio mínimo
0	x	α^7	$x^4 + x^3 + 1$
1	$x + 1$	α^8	$x^4 + x + 1$
α	$x^4 + x + 1$	α^9	$x^4 + x^3 + x^2 + x + 1$
α^2	$x^4 + x + 1$	α^{10}	$x^2 + x + 1$
α^3	$x^4 + x^3 + x^2 + x + 1$	α^{11}	$x^4 + x^3 + 1$
α^4	$x^4 + x + 1$	α^{12}	$x^4 + x^3 + x^2 + x + 1$
α^5	$x^2 + x + 1$	α^{13}	$x^4 + x^3 + 1$
α^6	$x^4 + x^3 + x^2 + x + 1$	α^{14}	$x^4 + x^3 + 1$

Observe que los elementos $\{\alpha^1, \alpha^2, \alpha^4, \alpha^8\}$ poseen el mismo polinomio mínimo; y que lo mismo les sucede a $\{\alpha^3, \alpha^6, \alpha^9, \alpha^{12}\}$, a $\{\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}\}$ y a $\{\alpha^5, \alpha^{10}\}$. ■

Dos elementos de un cuerpo con idéntico polinomio mínimo sobre el mismo subcuerpo $\mathcal{F}$ son *conjugados*.[5] De la definición se desprenden varias propiedades sencillas e importantes acerca de los polinomios mínimos.

Teorema 9.16. *Sea* GF(p^m) *una extensión de* GF(p)*, y sea γ un elemento de* GF(p^m) *con polinomio mínimo $M_\gamma(x)$ en $F_p[x]$.*

a) $M_\gamma(x)$ es irreducible en $F_p[x]$ y único.

[5]Esta es la razón de que en el cuerpo de los números complejos los números i y $-i$ se digan conjugados: ambos tienen al polinomio real $x^2 + 1$ como polinomio mínimo.

b) Si $g(x)$ es un polinomio de $F_p[x]$ y $g(\gamma) = 0$, $M_\gamma(x)$ divide a $g(x)$.

c) $M_\gamma(x)$ divide a $x^{p^m} - x$.

d) El grado de $M_\gamma(x)$ es un divisor de m.

e) El polinomio mínimo de un elemento primitivo de $\mathrm{GF}(p^m)$ *es de grado m.*

Demostración.

a) La prueba procede por simple contradicción. Si $M_\gamma(x)$ no fuese irreducible en $F_p[x]$, sería el producto de otros dos de menor grado, $M_\gamma(x) = Q_1(x)Q_2(x)$. Pero $M_\gamma(\gamma) = 0$ implica $Q_1(\gamma) = 0$ o $Q_2(\gamma) = 0$, y $M_\gamma(x)$ no sería mínimo. Para probar la unicidad, supóngase que existiesen en $F_p[x]$ dos polinomios mónicos distintos de grado mínimo con $P(\gamma) = Q(\gamma) = 0$. $P(x) - Q(x)$ sería entonces un polinomio de grado inferior a ambos, y γ uno de sus ceros.

b) Llamemos $r(x) \neq 0$ al resto de dividir $g(x)$ entre $M_\gamma(x)$: $g(x) = c(x)M_\gamma(x) + r(x)$. Pero $r(\gamma) = g(\gamma) - c(\gamma)M_\gamma(\gamma) = 0$, con $r(x)$ de grado inferior a $M_\gamma(x)$, lo que no es posible. Por tanto, $r(x) = 0$.

c) Se trata de una consecuencia del teorema de Fermat y de un caso particular del punto anterior: $\gamma^{p^m} = \gamma$ para todo $\gamma \in \mathrm{GF}(p^m)$.

d) Si $M_\gamma(x)$ tiene grado s, como es irreducible, se puede utilizar para construir el cuerpo $\mathrm{GF}(p^s) \subseteq \mathrm{GF}(p^m)$. Pero entonces s es un divisor de m.

e) Sea α un elemento primitivo. Si $M_\alpha(x)$ es de grado n, entonces con él se puede construir $\mathrm{GF}(p^n)$. Pero, por ser α primitivo, $\mathrm{GF}(p^m) \subseteq \mathrm{GF}(p^n)$. Y como n es un divisor de m, solamente puede ser $m = n$. ▶

Definición 9.7. *El polinomio mínimo de un elemento primitivo es un polinomio primitivo.*

Gracias a estas propiedades se probarán ahora los dos resultados más importantes de este capítulo. En primer lugar, la unicidad de todos los cuerpos de igual cardinal.

Teorema 9.17. *Todos los cuerpos de cardinal p^m son isomorfos.*

Vector	Polinomio	$F_2[x]/x^4+x^3+1$	$F_2[x]/x^4+x+1$
0000	0	0	0
0001	1	1	1
0010	x	η	α
0100	x^2	η^2	α^2
1000	x^3	η^3	α^3
1001	x^3+1	η^4	α^{14}
1011	x^3+x+1	η^5	α^7
1111	x^3+x^2+x+1	η^6	α^{12}
0111	x^2+x+1	η^7	α^{10}
1110	x^3+x^2+x	η^8	α^{11}
0101	x^2+1	η^9	α^8
1010	x^3+x	η^{10}	α^9
1101	x^3+x^2+1	η^{11}	α^{13}
0011	$x+1$	η^{12}	α^4
0110	x^2+x	η^{13}	α^5
1100	x^3+x^2	η^{14}	α^6

Tabla 9.2. Dos versiones del cuerpo de cardinal 16.

Demostración. Tomemos dos cuerpos, $\mathcal{C}_1$ y $\mathcal{C}_2$, de cardinal p^m. Sea α un elemento primitivo del primero, y $M_\alpha(x)$ su polinomio mínimo, que divide a $x^{p^m}-x$ y es de grado m. En este supuesto, el teorema de Fermat implica la existencia en $\mathcal{C}_2$ de un elemento γ cuyo polinomio mínimo es precisamente $M_\alpha(x)$. Para comprender por qué, es suficiente con observar que si $M_1(x),\ldots,M_r(x)$ son todos los polinomios mínimos distintos de los elementos de $\mathcal{C}_1$, dado que son irreducibles y divisores de $x^{p^m}-x$, entonces

$$x^{p^m}-x=\prod_{j=1}^{r}M_j(x)$$

es la única descomposición de $x^{p^m}-x$ en factores irreducibles sobre $F_p[x]$; acontece siempre, en suma, que dos cuerpos del mismo cardinal tienen el mismo conjunto de polinomios mínimos.

Pero $\mathcal{C}_1$ puede interpretarse como el conjunto de polinomios de grado menor que m en la indeterminada α, o como el conjunto de polinomios de $F_p[x]$ módulo $M_\alpha(x)$. De manera análoga, $\mathcal{C}_2$ incluye (y, por lo tanto, es) el conjunto de polinomios de grado menor que m en la indeterminada γ. La transformación $\alpha\to\gamma$ es, obviamente, un isomorfismo entre $\mathcal{C}_1$ y $\mathcal{C}_2$. ▶

Ejemplo 9.5. El polinomio x^4+x^3+1 es irreducible en $F_2[x]$, y genera el cuerpo $F_2[x]/x^4+x^3+1$, que se relaciona en la tabla 9.2, siendo $\eta=x$ un elemento primitivo suyo. El polinomio mínimo del elemento η^7 es $M_{\eta^7}(x)=$

$x^4 + x + 1$, lo que hace de la transformación $\alpha \to \eta^7$ una biyección entre los cuerpos $F_2[x]/x^4 + x + 1$ y $F_2[x]/x^4 + x^3 + 1$. En la tabla 9.2 se han representado las dos versiones de estos cuerpos de cardinal 16. Así, por ejemplo, la expresión $\alpha + \alpha^2 = \alpha^5$ en el primero se convierte en $\eta^7 + (\eta^7)^2 = (\eta^7)^5$ en el segundo. Note que η^7 es un elemento primitivo de $F_2[x]/x^4 + x^3 + 1$, pues un isomorfismo entre dos cuerpos necesariamente convierte elementos primitivos en elementos primitivos.

Un isomorfismo entre dos cuerpos es simplemente una forma de reescribir los elementos del primero sin cambiar la aritmética, es decir, preservando formalmente la suma y la multiplicación. ■

Y en segundo lugar, que el cuerpo de Galois GF(p^m) existe siempre o, por expresarlo de otro modo, que existen polinomios irreducibles en $F_p[x]$, para p primo, de cualquier grado $m \geq 1$.

Teorema 9.18. *Dados p primo y $m \geq 1$, existe un cuerpo de orden p^m.*

Demostración. Si $m = 1$ el cuerpo es simplemente GF(p), construido como en el ejemplo 9.1. Para $m > 1$, la técnica de la demostración será construir una sucesión de cuerpos finitos hasta obtener uno que contenga a todas las raíces de $x^{p^m} - x$ y conste exactamente de p^m elementos.

Sea, entonces, $p_1(x)$ un polinomio irreducible de grado al menos 2 sobre $\mathcal{C}_1 = \text{GF}(p)$, factor de $x^{p^m} - x$ (vid. el teorema 9.23 que afirma que existen polinomios irreducibles de grado cualquiera). A partir de él construyamos el cuerpo $\mathcal{C}_2$ tal como se hizo en el apartado 9.2: $\mathcal{C}_2$ será el conjunto de polinomios de grado menor que $p_1(x)$ con coeficientes de $\mathcal{C}_1$. Si $\mathcal{C}_2$ es de cardinal menor que p^m, seleccionemos un factor irreducible de $x^{p^m} - x$ sobre C_2, y apliquemos de nuevo la construcción del apartado 9.2 para obtener una extensión $\mathcal{C}_3$ del cuerpo $\mathcal{C}_2$. Procediendo de esta forma, tras un número finito de etapas habremos obtenido un cuerpo con todas las raíces de $x^{p^m} - x$.

Observación: el teorema 9.22 garantiza la existencia de al menos un factor irreducible de $x^{p^m} - x$ sobre $F_p[x]$ con grado s divisor de m, por lo que la construcción de un cuerpo finito a partir de un subcuerpo suyo siempre podrá realizarse en un único paso. ▶

Ejemplo 9.6. Podemos construir GF(2^4) de dos formas distintas extendiendo GF(2). El polinomio $x^2 + x + 1$ es un factor irreducible de $x^{2^4} - x$ sobre $\mathcal{C}_1 = \text{GF}(2)$, razón por la que se puede afirmar que GF(4) es isomorfo al conjunto $\mathcal{C}_2 = \{0, 1, x, x+1\} = \{0, 1, \alpha, \alpha^2\}$ de residuos polinómicos módulo $x^2 + x + 1$. Y, sobre $\mathcal{C}_2$, $x^2 + \alpha x + 1$ es un divisor irreducible de $x^{2^4} - x$, de manera que podemos utilizar $x^2 + \alpha x + 1$ para construir un cuerpo isomorfo a GF(2^4). Por otro lado, $\pi(x) = x^4 + x + 1$ es un factor irreducible de $x^{2^4} - x$

sobre el cuerpo binario; el conjunto de residuos polinómicos módulo $\pi(x)$ engendra, pues, (otro cuerpo isomorfo a) $\mathrm{GF}(2^4)$. ■

9.5. Factorización de $x^n - 1$

El problema de la factorización del polinomio $x^n - 1$ sobre un cuerpo $\mathrm{GF}(q)$ es fundamental en el estudio de los códigos cíclicos. Además, cuando $n = q^m - 1$, tal factorización da los polinomios mínimos de los elementos de $\mathrm{GF}(q^m)$ sobre $\mathrm{GF}(q)$. Se verá en este apartado que la estructura básica de la descomposición de $x^n - 1$ en factores irreducibles depende tan sólo de q y n.

Para ello, basta suponer el caso en que q y n son primos entre sí. De lo contrario, si $n = rq$, sería $x^n - 1 = (x^r - 1)^q$, y $x^n - 1$ tendría raíces con multiplicidad q. Por tanto, si $\mathrm{mcd}(n, q) = 1$, todas las raíces de $x^n - 1$ son simples y distintas, y se conocen como *raíces n-ésimas de la unidad*.

Un importante resultado de la teoría de números, el teorema de Fermat–Euler, establece que, para q y n primos relativos, existe un entero positivo m mínimo tal que $q^m - 1$ es divisible por n. El exponente m se conoce como *orden multiplicativo de q módulo n*, y es posible probar que, como consecuencia, el polinomio $x^n - 1$ divide a $x^{q^m-1} - 1$ pero no divide a $x^{q^s-1} - 1$ para ningún $s < m$. Quiere esto decir que las raíces de $x^n - 1$ pertenecen al cuerpo finito $\mathrm{GF}(q^m)$, y que no hay ningún otro cuerpo menor de característica q que las contenga. Las raíces n-ésimas de la unidad son un subgrupo cíclico de $\mathrm{GF}(q^m)$ cuyo elemento generador recibe el nombre de *raíz primitiva n-ésima de la unidad*. Cuando $n = p^m - 1$, el menor cuerpo de característica p al que pertenecen las raíces n-ésimas de la unidad es precisamente $\mathrm{GF}(p^m)$ y una raíz primitiva es un elemento primitivo del cuerpo.

El siguiente resultado da una caracterización de los polinomios que tienen entre sus ceros a alguna raíz n-ésima de la unidad.

TEOREMA 9.19. *Sean $f(x)$ un polinomio con coeficientes en $\mathrm{GF}(q)$ y γ una raíz de $f(x)$ en la extensión $\mathrm{GF}(q^s)$.*

a) γ^q es una raíz de $f(x)$ en $\mathrm{GF}(q^s)$.

b) Si γ tiene orden n y t es el orden multiplicativo de q módulo n, entonces γ, γ^q, ..., $\gamma^{q^{t-1}}$ son raíces distintas de $f(x)$ en $\mathrm{GF}(q^s)$.

DEMOSTRACIÓN. Ambas partes son una consecuencia inmediata del corolario 9.15. En el caso particular de que $f(x)$ sea el polinomio mínimo de γ,

ello implica que γ, γ^q, ..., $\gamma^{q^{t-1}}$ son elementos conjugados. ▶

De acuerdo con el resultado precedente, si α es una raíz primitiva de la unidad y $0 \leq s < n$, los elementos α^s, α^{sq}, ..., $\alpha^{sq^{k-1}}$ son conjugados, siendo k el menor entero tal que $sq^k \equiv s \mod n$. Por consiguiente, la operación de multiplicar por q (módulo n) induce una partición de los enteros $0 \leq s < n$ en los conjuntos

$$\mathcal{A}_s = \{s, sq, \ldots, sq^{k-1}\} \mod n.$$

$\mathcal{A}_s$ es el *conjunto q-ciclotómico* (módulo n) de representante s. Visto de manera equivalente, los conjuntos q-ciclotómicos definen una partición de las raíces n-ésimas de la unidad en las clases de elementos conjugados

$$\mathcal{C}_{\alpha^s} = \{\alpha^s, \alpha^{sq}, \ldots, \alpha^{sq^{k-1}}\}$$

que son clases de elementos del mismo orden.

Ejemplo 9.7. Los conjuntos 2-ciclotómicos módulo 15 son:

$$\begin{array}{ll} \mathcal{A}_0 = \{0\} & \mathcal{A}_5 = \{5, 10\} \\ \mathcal{A}_1 = \{1, 2, 4, 8\} & \mathcal{A}_7 = \{7, 14, 13, 11\} \\ \mathcal{A}_3 = \{3, 6, 12, 9\} & \end{array}$$

y $\{\alpha^0\}$, $\{\alpha^1, \alpha^2, \alpha^4, \alpha^8\}$, $\{\alpha^3, \alpha^6, \alpha^{12}, \alpha^9\}$, $\{\alpha^5, \alpha^{10}\}$, $\{\alpha^7, \alpha^{14}, \alpha^{13}, \alpha^{11}\}$ son las clases de elementos conjugados de GF(16) (α un elemento primitivo) con órdenes respectivos 1, 15, 5, 3 y 15. Los conjuntos ciclotómicos de 2 módulo 31 son:

$$\begin{aligned} \mathcal{B}_0 &= \{0\} \\ \mathcal{B}_1 &= \{1, 2, 4, 8, 16\} \\ \mathcal{B}_3 &= \{3, 6, 12, 24, 17\} \\ \mathcal{B}_5 &= \{5, 10, 20, 9, 18\} \\ \mathcal{B}_7 &= \{7, 14, 28, 25, 19\} \\ \mathcal{B}_{11} &= \{11, 22, 13, 26, 21\} \\ \mathcal{B}_{15} &= \{15, 30, 29, 27, 23\} \end{aligned}$$

Tenga presente que cualquier elemento no nulo de GF(32), a excepción de la unidad, tiene orden 31. Normalmente se elige como representante de un conjunto ciclotómico, por comodidad, al menor de sus elementos, aunque este convenio no tiene la menor trascendencia. ■

El teorema 9.19 apunta una relación directa entre polinomios mínimos y conjuntos ciclotómicos.

TEOREMA 9.20. *Sea* $\mathrm{GF}(q^s)$ *una extensión de* $\mathrm{GF}(q)$ *y sea* α *una raíz primitiva* n*-ésima en* $\mathrm{GF}(q^s)$.

a) El polinomio mínimo de α^s *sobre* $\mathrm{GF}(q)$ *es*

$$M_{\alpha^s}(x) = \prod_{i \in \mathcal{A}_s} (x - \alpha^i),$$

siendo $\mathcal{A}_s$ *el conjunto* q*-ciclotómico de representante* s.

b) $x^n - 1 = \prod_j M_{\alpha^j}(x)$, *en donde el índice* j *recorre todos los representantes distintos de los conjuntos* q*-ciclotómicos.*

DEMOSTRACIÓN.

a) Se probará en primer lugar que $\prod_{i \in \mathcal{A}_r}(x - \alpha^i)$ es un polinomio con coeficientes en $\mathrm{GF}(q)$.

Supóngase que el conjunto ciclotómico $\mathcal{A}_r = \{\alpha^r, \alpha^{rq}, \ldots, \alpha^{rq^{w-1}}\}$ tiene w elementos. Los coeficientes de $\prod_{i \in A_r}(x - \alpha^i) = \sum_{j=0}^{w} a_j x^j$ son las funciones elementales simétricas de los elementos α^r, $\ldots$, $\alpha^{rq^{w-1}}$. Explícitamente

$$\begin{aligned}
a_0 &= (-\alpha^r) \cdots (-\alpha^{rq^{w-1}}) \\
a_1 &= (-1) \sum_{k=0}^{w-1} \alpha^{rq^k} \\
&\vdots \\
a_j &= (-1)^j \sum_{i_1 \neq i_2 \neq \cdots \neq i_j} \alpha^{i_1} \cdots \alpha^{i_j}, \quad i_1, \ldots, i_j \in \{r, rq, \ldots, rq^{w-1}\} \\
&\vdots \\
a_w &= 1
\end{aligned}$$

Esto es, para $j < w$, el coeficiente a_j es la suma de productos de la forma $\alpha^{i_1}\alpha^{i_2} \cdots \alpha^{i_j}$, recorriendo todas las combinaciones de j exponentes $i_1, \ldots, i_j$ del conjunto ciclotómico $\mathcal{A}_r$, distintos entre sí. Ahora bien,

$$(\alpha^{i_1}\alpha^{i_2} \cdots \alpha^{i_j})^q = \alpha^{qi_1} \cdots \alpha^{qi_j} = \alpha^{i'_1} \cdots \alpha^{i'_j}$$

en donde $\{i'_1, \ldots, i'_j\}$ es una permutación de $\{i_1, \ldots, i_j\}$. Así pues, $a_j^q = a_j$ para $j = 0, \ldots, w$, lo que significa que $a_j \in \mathrm{GF}(q)$.

Para probar que $\prod_{i \in \mathcal{A}_r}(x - \alpha^i)$ es el polinomio mínimo de α^r, es suficiente observar que ningún otro polinomio mónico de menor grado se anula en α^r y en todos sus conjugados.

b) Es obvio que

$$x^n - 1 = \prod_{i \in \cup_j \mathcal{A}_j} (x - \alpha^i) = \prod_j \prod_{i \in \mathcal{A}_j} (x - \alpha^i) = \prod_j M_{\alpha^j}(x)$$

en donde j recorre el conjunto de representantes de los conjuntos ciclotómicos. ▶

En consecuencia, los grados de los factores irreducibles de $x^n - 1$ son los cardinales de los conjuntos q-ciclotómicos.

Aplicando el teorema 9.20 a un elemento primitivo tenemos que

Teorema 9.21. *Sea α un elemento primitivo de* $\mathrm{GF}(q^m)$

$$x^{q^m} - x = x \cdot \prod_{j \in \mathcal{A}} M_{\alpha^j}(x).$$

Habiendo observado que $x^{p^m} - x$ es el producto de todos los polinomios mínimos del cuerpo $\mathrm{GF}(p^m)$ sobre $F_p[x]$, y recordando que estos polinomios mínimos son siempre irreducibles y de grado menor o igual que m, cabría preguntarse ¿qué polinomios irreducibles sobre $F_p[x]$ dividen a $x^{p^m} - x$? La respuesta es que son *todos* los polinomios mónicos irreducibles cuyo grado es un divisor de m. Así que la descomposición de $x^{p^m} - x$ que se acaba de deducir sirve no sólo para calcular los polinomios mínimos, sino también para calcular de paso todos los posibles polinomios irreducibles de cierto grado.

Teorema 9.22. *Cualquier polinomio mónico irreducible en $F_q[x]$ cuyo grado sea un divisor de m es factor de $x^{q^m} - x$.*

Demostración. Sea $p(x)$ mónico e irreducible en $F_q[x]$, con grado r divisor de m. Como el caso $p(x) = x$ es trivial, supongamos que $p(x) \neq x$. Con $p(x)$ se puede construir el cuerpo $\mathrm{GF}(q^r) \subseteq \mathrm{GF}(q^m)$, en el que existirá, por tanto, un elemento que tenga a $p(x)$ como polinomio mínimo, y además $p(x)$ será divisor de $x^{q^r-1} - 1$. Como, por el lema 9.11, $x^{q^r-1} - 1$ divide a $x^{q^m-1} - 1$, se concluye que $p(x)$ es factor de $x^{q^m-1} - 1$ y también de $x^{q^m} - x$. ▶

Ejemplo 9.8. Hallemos todos los polinomios binarios irreducibles de grado 5 calculando la factorización de $x^{2^5} - x$ en polinomios mínimos. Los conjuntos ciclotómicos de 2 módulo 31 son:

$$\begin{array}{ll} \mathcal{A}_0 = \{0\} & \mathcal{A}_7 = \{7, 14, 28, 25, 19\} \\ \mathcal{A}_1 = \{1, 2, 4, 8, 16\} & \mathcal{A}_{11} = \{11, 22, 13, 26, 21\} \\ \mathcal{A}_3 = \{3, 6, 12, 24, 17\} & \mathcal{A}_{15} = \{15, 30, 29, 27, 23\} \\ \mathcal{A}_5 = \{5, 10, 20, 9, 18\} & \end{array}$$

Si para construir $\mathrm{GF}(2^5)$ se parte del polinomio irreducible y primitivo $x^5 + x^2 + 1$, los polinomios mínimos asociados a los anteriores conjuntos ciclotómicos valen, aplicando directamente el teorema 9.21:

$$\begin{array}{ll} M_{\alpha^0}(x) = x + 1 & M_{\alpha^7}(x) = x^5 + x^3 + x^2 + x + 1 \\ M_{\alpha^1}(x) = x^5 + x^2 + 1 & M_{\alpha^{11}}(x) = x^5 + x^4 + x^3 + x + 1 \\ M_{\alpha^3}(x) = x^5 + x^4 + x^3 + x^2 + 1 & M_{\alpha^{15}}(x) = x^5 + x^3 + 1 \\ M_{\alpha^5}(x) = x^5 + x^4 + x^2 + x + 1 & \end{array}$$

Entonces

$$x^{31} - 1 = M_{\alpha^0}(x)M_{\alpha^1}(x)M_{\alpha^3}(x)M_{\alpha^5}(x)M_{\alpha^7}(x)M_{\alpha^{11}}(x)M_{\alpha^{15}}(x).$$

Observe que, como 5 es un número primo, los factores de $x^{32} - x$ sólo pueden tener grado 1 o grado 5, y $\mathrm{GF}(2)$ es el único subcuerpo de $\mathrm{GF}(2^5)$. Observe también que $x^5 + x^2 + 1$ y $x^5 + x^3 + 1$ son recíprocos, en el sentido de que los coeficientes de uno son los del otro escritos en sentido inverso. La relación algebraica general entre un polinomio $p(x)$ y su recíproco $p_r(x)$ es

$$p_r(x) = x^{\text{grado de } p(x)} p(x^{-1}),$$

que expresa que las raíces del polinomio recíproco son, aparte del cero, los inversos de las raíces de $p(x)$. En el caso presente, también son recíprocos los factores $M_{\alpha^3}(x)$ y $M_{\alpha^7}(x)$, así como $M_{\alpha^5}(x)$ y $M_{\alpha^{11}}(x)$. ■

Finalmente, otra importante implicación del teorema 9.22 es que permite asegurar que en $F_p[x]$ existen polinomios irreducibles de cualquier grado.

TEOREMA 9.23. *Sea $I_p(m)$ el número de polinomios mónicos irreducibles de grado $m \geq 1$ con coeficientes en $\mathrm{GF}(p)$. Se verifica que*

$$I_p(m) \geq 1 \qquad \forall m, p.$$

DEMOSTRACIÓN. Elijamos p primo y $m \geq 1$ arbitrarios. En virtud del teorema 9.21, el polinomio $x^{p^m} - x$ es igual al producto de todos los polinomios mónicos irreducibles sobre $\mathrm{GF}(p)$ cuyo grado sea un divisor de m. Así pues, igualando los grados

$$p^m = \sum_{d|m} d I_p(d)$$

en donde $d \mid m$ indica que d es un divisor de m. A partir de esta identidad, se tiene en primer lugar que $p = I_p(1)$. Tomando sólo el primer y el último sumandos, y como $I_p(d) \geq 0$, se obtiene la cota

$$p^m \geq p + m I_p(m) \Rightarrow I_p(m) \leq \frac{p^m - p}{m}$$

en la que se alcanza la igualdad estricta sólo si m es un número primo. Esto significa, en particular, que $I_p(2) = (p^2 - p)/2 > 0$. Pero además, el número de polinomios irreducibles se puede acotar inferiormente viendo que

$$p^m \leq mI_p(m) + \sum_{i=1}^{m/2} p^i < mI_p(m) + p^{m/2+1},$$

es decir, que

$$I_p(m) > \frac{p^m - p^{m/2+1}}{m} = \frac{p^m}{m}\left(1 - p^{-m/2+1}\right)$$

y que $I_p(m) > 0$ para cualquier $m \geq 2$. Puesto que ya se dedujo que tanto $I_p(1)$ como $I_p(2)$ son no nulos, el teorema queda probado. ▶

Nótese que, como el polinomio mínimo de los elementos primitivos de $\mathrm{GF}(p^m)$ es irreducible y de grado m, e $I_p(m) \geq 1$, resulta que existen polinomios primitivos de cualquier grado $m \geq 1$ sobre $\mathrm{GF}(p)$. Es decir, para construir $\mathrm{GF}(p^m)$ (apartado 9.2) *siempre* es posible elegir un polinomio primitivo de grado m.

Por otro lado, es posible formalizar un argumento de cuenta distinto para obtener el número de polinomios irreducibles. Sea a_k el número de polinomios mónicos de grado k en $F_p[x]$ y definamos el siguiente polinomio enumerador de polinomios mónicos:

$$A(z) = \sum_{k=0}^{\infty} a_k z^k.$$

Puesto que $a_k = p^k$, es inmediato ver que $A(z) = (1 - pz)^{-1}$. Ahora, sea $p(x)$ un polinomio irreducible en $F_p[x]$ de grado r y considérese el conjunto formado por todas las potencias de $p(x)$. El enumerador de grados de los polinomios de este conjunto es el polinomio

$$1 + z^r + z^{2r} + \cdots = (1 - z^r)^{-1}.$$

Si un polinomio mónico de $F_p[x]$ es el producto de k factores irreducibles, de grados $m_1, m_2, \ldots, m_k$, cada uno de ellos elevado a las potencias $i_1, i_2 \ldots, i_k$, entonces podemos identificar este polinomio mónico con uno y sólo uno de los términos de grado $i_1 + i_2 + \cdots + i_k$ del producto

$$(1 - z^{m_1})^{-1}\,(1 - z^{m_2})^{-1} \cdots (1 - z^{m_k})^{-1}$$

y viceversa, cualquiera de los términos que aparece al desarrollar este producto representa un único polinomio mónico. Si existen $I_p(m)$ polinomios irreducibles de grado m en $F_p[x]$, podemos extender este argumento de la siguiente manera.

TEOREMA 9.24.

$$(1-pz)^{-1} = \prod_{m=1}^{\infty} (1-z^m)^{-I_p(m)}\,.$$

DEMOSTRACIÓN. El conjunto de todos los polinomios mónicos de $F_p[x]$ es igual al conjunto de todos los posibles productos entre los polinomios irreducibles de $F_p[x]$. ▶

Derivando con respecto a z ambos miembros de la anterior identidad, resulta

$$\frac{p}{(1-pz)^2} = \sum_{m=1}^{\infty} \frac{mI_p(m)z^{m-1}}{(1-pz)(1-z^m)}$$

o bien

$$\frac{pz}{1-pz} = \sum_{m=1}^{\infty} mI_p(m)\frac{z^m}{1-z^m}.$$

Sustituyendo las fracciones por su desarrollo en serie, esta igualdad se puede escribir como

$$\begin{aligned}\sum_{k=1}^{\infty}(pz)^k = \sum_{m=1}^{\infty} mI_p(m)\sum_{j=1}^{\infty} z^{mj} &= \sum_{m=1}^{\infty} mI_p(m)\sum_{k:m|k} z^k \\ &= \sum_{k=1}^{\infty} z^k \sum_{m|k} mI_p(m).\end{aligned}$$

E igualando los coeficientes asociados al mismo exponente queda finalmente

$$p^m = \sum_{d|m} dI_p(d)$$

tal y como fue utilizada en el teorema 9.23.

El cálculo explícito del número de polinomios irreducibles de un grado dado puede efectuarse apelando a la *fórmula de inversión de Moebius*

$$I_p(m) = \frac{1}{m}\sum_{d|m} \mu(d)p^{m/d}$$

en donde $\mu(d)$ es la función de Moebius

$$\mu(d) = \begin{cases} 1 & \text{si } d = 1 \\ (-1)^r & \text{si } d \text{ es el producto de } r \text{ primos distintos} \\ 0 & \text{en otro caso.} \end{cases}$$

9.6. Periodo de un polinomio

Un problema íntimamente relacionado con la factorización de $x^n - 1$ y, en cierto modo, inverso a aquél es el siguiente: dado un polinomio mónico $f(x) \in F_q[x]$, ¿cuál es el menor entero n tal que $f(x)$ divide a $x^n - 1$? O bien, en lenguaje de la teoría de códigos, ¿cuál es, en el alfabeto F_q, la longitud n del menor código cíclico engendrado por $f(x)$? Tal entero, que siempre existe, se llama *periodo* de $f(x)$.

Supongamos inicialmente que $f(x)$ es un polinomio irreducible en $F_q[x]$. En tal caso, su periodo n es un divisor del orden multiplicativo de sus raíces, es decir, n divide a $q^{\text{grado}(f(x))} - 1$. Si $f(x)$ fuese primitivo, su periodo sería exactamente $q^{\text{grado}(f(x))} - 1$.

El periodo de una potencia de $f(x)$ puede hallarse con el siguiente razonamiento. $f(x)$ solamente divide a $x^a - 1$ si a es un múltiplo de n. Supongamos que $a = p^i j$, con p la característica de F_q y j un entero que no tiene a p entre sus factores. Entonces

$$x^a - 1 = (x^j - 1)^{p^i}.$$

Pero $x^j - 1$ no contiene raíces repetidas en ninguna extensión de F_q, pues q y j son coprimos. Por lo tanto, $x^a - 1$ tiene raíces con multiplicidad p^i. En tal caso, el periodo de $f(x)^m$ será igual a n, el periodo de $f(x)$, multiplicado por la menor potencia de p que es mayor o igual que m. Generalizando este razonamiento, tendremos que

Teorema 9.25. *Sea $f(x) = f_1(x)^{e_1} f_2(x)^{e_2} \cdots f_s(x)^{e_s}$ la descomposición de $f(x)$ en factores irreducibles sobre $F_q[x]$, y sean n_1, n_2, ..., n_s los periodos de estos factores. El periodo de $f(x)$ es*

$$\text{mcm}\,(n_1, n_2, \ldots, n_s)\, p^{\lceil \log_p \max_{1 \le i \le s} e_i \rceil}.$$

Por lo tanto, el problema queda reducido a hallar el periodo de los factores irreducibles de $f(x)$. Puesto que el periodo de un factor de grado r es un divisor de $q^r - 1$, un método de cálculo consiste en descomponer $q^r - 1$ en factores primos, $q^r - 1 = \prod p_i^{m_i}$; si

$$x^{(q^r-1)/(p_i^{m_i})} \not\equiv 1 \mod f(x)$$

entonces $p_i^{m_i}$ es el periodo de $f(x)$. En otro caso, repetiremos la misma comprobación con los factores p_i^s, $s < m_i$, y con el resto de los factores primos de $q^r - 1$.

Ejemplo 9.9. Calculemos el periodo del polinomio irreducible $g(x) = x^8 + x^6 + x^5 + x^3 + 1$. La tabla siguiente contiene los residuos módulo $g(x)$ de algunas potencias x^s

$x^i \mod g(x)$		$x^i \mod g(x)$	
x^8	$x^6+x^5+x^3+1$	x^{16}	$x+1$
x^{10}	$x^7+x^6+x^3+x^2+1$	x^{32}	x^2+1
x^{12}	$x^7+x^3+x^2+x+1$	x^{64}	x^4+1
x^{14}	$x^7+x^6+x^5+x^3+x^2+x$	x^{128}	$x^6+x^5+x^3$

El periodo debe ser un divisor de $2^8-1=3\cdot 5\cdot 17$. Pues bien, se puede comprobar directamente, haciendo uso de la tabla, que $x^{15}\not\equiv 1 \mod g(x)$ y que $x^{17}\not\equiv 1 \mod g(x)$, de modo que el periodo no es ni 15 ni 17. Probando con $5\cdot 7=35$

$$x^{35}\equiv x^{32}x^2x\not\equiv 1 \mod g(x)$$

y con $3\cdot 17=51$

$$x^{51}\equiv x^{32}x^{16}x^2x\not\equiv 1 \mod g(x).$$

Haciendo lo mismo con $5\cdot 17=85$, $x^{85}\equiv x^{64}x^{16}x^4x\not\equiv 1 \mod g(x)$. De lo que se concluye que el periodo es, necesariamente, 255. ■

Notas bibliográficas

La teoría de cuerpos finitos es una rama ya clásica del álgebra formal, y se encuentra descrita en numerosos textos. Berlekamp [3] se ocupa de revisar no sólo los fundamentos sino también algunas relaciones con la teoría de números y la factorización de polinomios. La obra de McEliece [47] aporta un carácter más básico y asequible.

$+$	$\cdot$	Conjuntos ciclotómicos	Polinomio mínimo
00	0		
01	1	$\{0\}$	$x+1$
10	α	$\{1,2\}$	x^2+x+1
11	α^2		

Tabla 9.3. $\mathbb{F}_4, p(x) = x^2 + x + 1$.

$+$	$\cdot$	$+$	$\cdot$	Conjuntos ciclotómicos	Polinomio mínimo
000	0	011	α^3		
001	1	110	α^4	$\{0\}$	$x+1$
010	α	111	α^5	$\{1,2,4\}$	x^3+x+1
100	α^2	101	α^6	$\{3,6,5\}$	x^3+x^2+1

Tabla 9.4. $\mathbb{F}_8, p(x) = x^3 + x + 1$.

$+$	$\cdot$	$+$	$\cdot$	Conjuntos ciclotómicos	Polinomio mínimo
0000	0	1011	α^7		
0001	1	0101	α^8		
0010	α	1010	α^9	$\{0\}$	$x+1$
0100	α^2	0111	α^{10}	$\{1,2,4,8\}$	x^4+x+1
1000	α^3	1110	α^{11}	$\{3,6,12,9\}$	$x^4+x^3+x^2+x+1$
0011	α^4	1111	α^{12}	$\{5,10\}$	x^2+x+1
0110	α^5	1101	α^{13}	$\{7,14,13,11\}$	x^4+x^3+1
1100	α^6	1001	α^{14}		

Tabla 9.5. $\mathbb{F}_{16}, p(x) = x^4 + x + 1$.

$+$	$\cdot$	$+$	$\cdot$	Conjuntos ciclotómicos	Polinomio mínimo
00000	0	11111	α^{15}		
00001	1	11011	α^{16}		
00010	α	10011	α^{17}		
00100	α^2	00011	α^{18}		
01000	α^3	00110	α^{19}		
10000	α^4	01100	α^{20}	$\{0\}$	$x+1$
00101	α^5	11000	α^{21}	$\{1,2,4,8,16\}$	x^5+x^2+1
01010	α^6	10101	α^{22}	$\{3,6,12,24,17\}$	$x^5+x^4+x^3+x^2+1$
10100	α^7	01111	α^{23}	$\{5,10,20,9,18\}$	$x^5+x^4+x^2+x+1$
01101	α^8	11110	α^{24}	$\{7,14,28,25,19\}$	$x^5+x^3+x^2+x+1$
11010	α^9	11001	α^{25}	$\{11,22,13,26,21\}$	$x^5+x^4+x^3+x+1$
10001	α^{10}	10111	α^{26}	$\{15,30,29,27,23\}$	x^5+x^3+1
00111	α^{11}	01011	α^{27}		
01110	α^{12}	10110	α^{28}		
11100	α^{13}	01001	α^{29}		
11101	α^{14}	10010	α^{30}		

Tabla 9.6. $\mathbb{F}_{32}, p(x) = x^5 + x^2 + 1$.

$+$	$\cdot$	$+$	$\cdot$	$+$	$\cdot$	$+$	$\cdot$	Conjuntos ciclotómicos
00	0	50	α^{15}	45	α^{31}	47	α^{47}	
01	1	23	α^{16}	11	α^{32}	15	α^{48}	$\{0\}$
02	α	46	α^{17}	22	α^{33}	32	α^{49}	$\{1,2,4,8,16,32\}$
04	α^2	17	α^{18}	44	α^{34}	64	α^{50}	$\{3,6,12,24,48,33\}$
10	α^3	36	α^{19}	13	α^{35}	53	α^{51}	$\{5,10,20,40,17,34\}$
20	α^4	74	α^{20}	26	α^{36}	25	α^{52}	$\{7,14,28,56,49,35\}$
40	α^5	73	α^{21}	54	α^{37}	52	α^{53}	$\{9,18,36\}$
03	α^6	65	α^{22}	33	α^{38}	27	α^{54}	$\{11,22,44,25,50,37\}$
06	α^7	51	α^{23}	66	α^{39}	56	α^{55}	$\{13,26,52,41,19,38\}$
14	α^8	21	α^{24}	57	α^{40}	37	α^{56}	$\{15,30,60,57,51,39\}$
30	α^9	42	α^{25}	35	α^{41}	76	α^{57}	$\{21,42\}$
60	α^{10}	07	α^{26}	72	α^{42}	77	α^{58}	$\{23,46,29,58,53,43\}$
43	α^{11}	16	α^{27}	67	α^{43}	75	α^{59}	$\{27,54,45\}$
05	α^{12}	34	α^{28}	55	α^{44}	71	α^{60}	$\{31,62,61,59,55,47\}$
12	α^{13}	70	α^{29}	31	α^{45}	61	α^{61}	
24	α^{14}	63	α^{30}	62	α^{46}	41	α^{62}	

Tabla 9.7. $\mathbb{F}_{64}, p(x) = x^6+x+1$. Los números de las columnas impares están en octal.

Polinomios mínimos

$M_{\alpha^0}(x) = x + 1$	$M_{\alpha^{13}}(x) = x^6 + x^4 + x^3 + x + 1$
$M_{\alpha}(x) = x^6 + x + 1$	$M_{\alpha^{15}}(x) = x^6 + x^5 + x^4 + x^2 + 1$
$M_{\alpha^3}(x) = x^6 + x^4 + x^2 + x + 1$	$M_{\alpha^{23}}(x) = x^6 + x^5 + x^4 + x + 1$
$M_{\alpha^5}(x) = x^6 + x^5 + x^2 + x + 1$	$M_{\alpha^{21}}(x) = x^2 + x + 1$
$M_{\alpha^7}(x) = x^6 + x^3 + 1$	$M_{\alpha^{27}}(x) = x^3 + x + 1$
$M_{\alpha^9}(x) = x^3 + x^2 + 1$	$M_{\alpha^{31}}(x) = x^6 + x^5 + 1$
$M_{\alpha^{11}}(x) = x^6 + x^5 + x^3 + x^2 + 1$	

Tabla 9.8. Polinomios mínimos de $\mathbb{F}_{64}$.

+	·	+	·	+	·	+	·	+	·
000	0	065	α^{25}	037	α^{51}	047	α^{77}	007	α^{103}
001	1	152	α^{26}	076	α^{52}	116	α^{78}	016	α^{104}
002	α	135	α^{27}	174	α^{53}	025	α^{79}	034	α^{105}
004	α^2	063	α^{28}	161	α^{54}	052	α^{80}	070	α^{106}
010	α^3	146	α^{29}	153	α^{55}	124	α^{81}	160	α^{107}
020	α^4	105	α^{30}	137	α^{56}	041	α^{82}	151	α^{108}
040	α^5	003	α^{31}	067	α^{57}	102	α^{83}	133	α^{109}
100	α^6	006	α^{32}	156	α^{58}	015	α^{84}	077	α^{110}
011	α^7	014	α^{33}	125	α^{59}	032	α^{85}	176	α^{111}
022	α^8	030	α^{34}	043	α^{60}	064	α^{86}	165	α^{112}
044	α^9	060	α^{35}	106	α^{61}	150	α^{87}	143	α^{113}
110	α^9	140	α^{36}	005	α^{62}	131	α^{88}	117	α^{114}
031	α^{10}	111	α^{37}	012	α^{63}	073	α^{89}	027	α^{115}
062	α^{11}	033	α^{38}	024	α^{64}	166	α^{90}	056	α^{116}
144	α^{13}	066	α^{39}	050	α^{65}	145	α^{91}	134	α^{117}
101	α^{14}	154	α^{40}	120	α^{66}	103	α^{92}	061	α^{118}
013	α^{15}	121	α^{41}	051	α^{67}	017	α^{93}	142	α^{119}
026	α^{16}	053	α^{42}	122	α^{68}	036	α^{94}	115	α^{120}
054	α^{17}	126	α^{43}	055	α^{69}	074	α^{95}	023	α^{121}
130	α^{18}	045	α^{44}	132	α^{70}	170	α^{96}	046	α^{122}
071	α^{19}	112	α^{45}	075	α^{71}	171	α^{97}	114	α^{123}
162	α^{20}	035	α^{46}	172	α^{72}	173	α^{98}	021	α^{124}
155	α^{21}	072	α^{47}	175	α^{73}	177	α^{99}	042	α^{125}
123	α^{22}	164	α^{48}	163	α^{74}	167	α^{100}	104	α^{126}
057	α^{23}	141	α^{49}	157	α^{75}	147	α^{101}		
136	α^{24}	113	α^{50}	127	α^{76}	107	α^{102}		

Tabla 9.9. $\mathbb{F}_{128}, p(x) = x^7 + x^3 + 1$. Los números de las columnas impares están en octal.

Conjuntos ciclotómicos	Polinomios mínimos
$\{0\}$	$x+1$
$\{1,2,4,8,16,32,64\}$	x^7+x^3+1
$\{3,6,12,24,48,65,96\}$	$x^7+x^3+x^2+x+1$
$\{5,10,20,33,40,66,80\}$	$x^7+x^4+x^3+x^2+1$
$\{7,14,28,56,67,97,112\}$	$x^7+x^6+x^5+x^4+x^2+x+1$
$\{9,17,18,34,36,68,72\}$	$x^7+x^5+x^4+x^3+x^2+x+1$
$\{11,22,44,49,69,88,98\}$	$x^7+x^6+x^4+x^2+1$
$\{13,26,35,52,70,81,104\}$	x^7+x+1
$\{15,30,60,71,99,113,120\}$	$x^7+x^6+x^5+x^4+x^3+x^2+x+1$
$\{19,25,38,50,73,76,100\}$	$x^7+x^6+x^2+x+1$
$\{21,37,41,42,74,82,84\}$	$x^7+x^6+x^5+x^2+1$
$\{23,46,57,75,92,101,114\}$	x^7+x^6+1
$\{27,51,54,77,89,102,108\}$	$x^7+x^6+x^4+x+1$
$\{29,39,58,78,83,105,116\}$	$x^7+x^5+x^3+x+1$
$\{31,62,79,103,115,121,124\}$	$x^7+x^6+x^5+x^4+1$
$\{43,45,53,85,86,90,106\}$	$x^7+x^5+x^2+x+1$
$\{47,61,87,94,107,117,122\}$	$x^7+x^5+x^4+x^3+1$
$\{55,59,91,93,109,110,118\}$	$x^7+x^6+x^5+x^4+x^3+x^2+1$
$\{63,95,111,119,123,125,126\}$	x^4+x^4+1

Tabla 9.10. Conjuntos ciclotómicos y polinomios mínimos de $\mathbb{F}_{2^7}$.

CAPÍTULO 10

Códigos BCH

Durante 1959 y 1960, A. Hocquenghem y, en un trabajo independiente, R. C. Bose y A. K. Ray–Chaudhuri construyeron una familia de códigos cíclicos binarios con capacidad para corregir errores múltiples que, como caso particular, contenía a la clase de códigos Hamming. Poco después de ser descubiertos, los nuevos códigos fueron generalizados para alfabetos no binarios.

Los códigos de Bose, Chaudhuri y Hocquenghem (BCH) son una amplia e importante subclase de códigos cíclicos. Se encuentran entre los mejores códigos conocidos para longitudes de bloque moderadas, de hasta varias centenas o pocos miles de bits, y se han desarrollado para ellos algoritmos eficientes de decodificación. Se sabe, sin embargo, que su capacidad de corrección de errores se deteriora asintóticamente con la longitud. Es decir, que para una tasa de codificación fija, la distancia no aumenta proporcionalmente con la longitud. No obstante, esta disminución de la eficiencia no comienza a mostrarse hasta longitudes de código realmente muy elevadas, y no supone ninguna limitación seria en la mayoría de las aplicaciones.

Además del interés que presentan en sí mismos, los códigos BCH se utilizan en la síntesis de otros muchos tipos de códigos. Por mencionar algunos, los Reed–Solomon (RS), a los que dedicaremos el próximo capítulo, son una especialización de los BCH no binarios, y poseen notables propiedades que los hacen aptos para ser utilizados bien por sí solos o bien formando parte de otras categorías de códigos. Los códigos de Justesen se construyen precisamente a partir de los RS, y destacan por constituir la primera familia de códigos no degenerados en ser descubierta, es decir, la primera en la que ni la tasa de codificación ni la razón entre distancia y longitud se aproximan a cero a medida que la longitud aumenta. Los propios BCH son una subclase

de una familia más amplia, la de los códigos alternantes, en la que se conoce la existencia de algunos códigos asintóticamente eficientes, como los Goppa.

En el resto de este capítulo se comenzará definiendo la clase general de códigos BCH, y se presentarán sus propiedades más importantes. Después, se describirán los métodos de decodificación algebraica aplicables a los códigos binarios y se expondrá un algoritmo de este tipo computacionalmente eficiente.

10.1. Construcción y propiedades

Comencemos por recordar que las raíces n-ésimas de la unidad son todas las soluciones de la ecuación algebraica $x^n - 1 = 0$, y que constituyen un subgrupo cíclico de cierto cuerpo $\mathbb{F}_{q^m} = \mathrm{GF}(q^m)$, subgrupo cuyo elemento generador se llama raíz primitiva n-ésima de la unidad. A lo largo de este capítulo, el símbolo α se referirá siempre a una raíz primitiva n-ésima de la unidad, y se supondrá que q y n son coprimos, es decir, $\mathrm{mcd}(q, n) = 1$.

Definición 10.1 (Código BCH). *El código BCH, definido sobre el cuerpo $\mathbb{F}_q$, de longitud n y distancia de diseño δ es el mayor código cíclico entre cuyos ceros hay $\delta - 1$ potencias consecutivas*

$$\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta-2}, \qquad b \geq 0,\ \delta \geq 1$$

de una n-ésima raíz primitiva $\alpha \in \mathbb{F}_{q^m}$ de la unidad (siendo m el orden multiplicativo de q módulo n, es decir, el menor entero m tal que n divide a $q^m - 1$).

Como el mayor código cíclico es, para una longitud dada, aquél de menor número de símbolos de redundancia, se puede formular esta otra definición equivalente.

Definición 10.2 (Código BCH). *Un código cíclico $\mathcal{C}$ de longitud n definido sobre el cuerpo $\mathbb{F}_q$ es un código BCH con distancia de diseño δ si su polinomio generador es*

$$g(x) = \mathrm{mcm}\{M_{\alpha^b}(x), M_{\alpha^{b+1}}(x), \ldots, M_{\alpha^{b+\delta-2}}(x)\}$$

el mínimo común múltiplo de los polinomios mínimos sobre $F_q[x]$ de α^b, α^{b+1}, ..., $\alpha^{b+\delta-2}$, en donde b y δ son enteros arbitrarios.[1] *O, lo que es lo mismo, $\mathcal{C}$ es un código BCH si $g(x)$ es el polinomio mónico de menor grado sobre $\mathbb{F}_q$ que admite a α^b, ..., $\alpha^{b+\delta-2}$ como raíces.*

[1] Si $\delta = 1$, se conviene en definir el código BCH como aquél cuyo polinomio generador tiene cero raíces, o sea $g(x) = 1$.

Cuando $\mathrm{mcd}(q,n)=1$, el polinomio x^n-1 se descompone en factores lineales sobre $\mathbb{F}_{q^m}[x]$, y todo código cíclico de longitud n sobre el alfabeto $\mathbb{F}_q$ queda completamente especificado por el conjunto de ceros de su polinomio generador, que lo son también de cualquier otro polinomio del código. Así pues, la definición de los códigos BCH simplemente añade esta condición: el conjunto de ceros es

$$\left\{\alpha^i \ : \ i \in \bigcup_{j=b}^{b+\delta-2} \mathcal{A}_j\right\}$$

en donde $\mathcal{A}_j$ es el conjunto q-ciclotómico (módulo n) que contiene a j. Esta restricción aparentemente arbitraria es suficiente para hallar una cota inferior al número de símbolos de información del código, tal y como se verá en el teorema 10.1.

Asimismo, obsérvese que la definición implica que, fijos b y n, un código BCH con distancia de diseño δ_1 está contenido en el código BCH de distancia de diseño $\delta_2 < \delta_1$.

En los casos en que $n = q^m - 1$, la raíz primitiva, α, resulta ser un elemento primitivo del cuerpo $\mathbb{F}_{q^m}$, y el código BCH se dice también *primitivo*. Cuando $b = 1$ se habla de códigos BCH en sentido restringido o estricto. Otro importante subconjunto de códigos BCH se obtiene al fijar $n = q-1$ (naturalmente $q > 2$, es decir, son no binarios) y es el de los códigos Reed–Solomon. En este capítulo se tratará únicamente con códigos BCH en sentido restringido (lo que no supone pérdida alguna de generalidad) y primitivos.

Ejemplo 10.1. Los conjuntos 2-ciclotómicos módulo 31, y sus respectivos polinomios mínimos, son:

$$\begin{aligned}
\mathcal{A}_0 &= \{0\} && \longrightarrow M_{\alpha^0}(x) = x+1\\
\mathcal{A}_1 &= \{1,2,4,8,16\} && \longrightarrow M_{\alpha^1}(x) = x^5+x^2+1\\
\mathcal{A}_3 &= \{3,6,12,17,24\} && \longrightarrow M_{\alpha^3}(x) = x^5+x^4+x^3+x^2+1\\
\mathcal{A}_5 &= \{5,9,10,18,20\} && \longrightarrow M_{\alpha^5}(x) = x^5+x^4+x^2+x+1\\
\mathcal{A}_7 &= \{7,14,19,25,28\} && \longrightarrow M_{\alpha^7}(x) = x^5+x^3+x^2+x+1\\
\mathcal{A}_{11} &= \{11,13,21,22,26\} && \longrightarrow M_{\alpha 11}(x) = x^5+x^4+x^3+x+1\\
\mathcal{A}_{15} &= \{15,23,27,29,30\} && \longrightarrow M_{\alpha^{15}}(x) = x^5+x^3+1.
\end{aligned}$$

El polinomio binario con ceros $\{\alpha^i, i \in \mathcal{A}_1 \cup \mathcal{A}_3\}$ genera un código binario BCH$[31,21]$ con distancia de diseño $\delta = 5$; el de ceros $\{\alpha^i, i \in \mathcal{A}_1 \cup \mathcal{A}_3 \cup \mathcal{A}_5\}$, un código BCH$[31,16]$ con $\delta = 7$; y el de ceros $\{\alpha^i, i \in \mathcal{A}_1 \cup \mathcal{A}_3 \cup \mathcal{A}_5 \cup \mathcal{A}_7\}$, un código BCH$[31,11]$ con $\delta = 11$. Todos estos códigos son primitivos.

Los conjuntos 2-ciclotómicos módulo 15 son:

$$\begin{aligned} \mathcal{A}_0 &= \{0\} & \mathcal{A}_5 &= \{5, 10\} \\ \mathcal{A}_1 &= \{1, 2, 4, 8\} & \mathcal{A}_7 &= \{7, 11, 13, 14\} \\ \mathcal{A}_3 &= \{3, 6, 9, 12\}. \end{aligned}$$

El dual del código BCH primitivo definido por los ceros

$$T = \{\alpha^i \ : \ i \in \mathcal{A}_1 \cup \mathcal{A}_3 \cup \mathcal{A}_7\}$$

es el que tiene por ceros a $T^\perp = \{\alpha^i \ : \ i \in \mathcal{A}_0 \cup \mathcal{A}_5\}$, que, al contrario que T, no es BCH en sentido restringido porque α^5 no es una raíz primitiva 15-ésima de la unidad. ■

Una primera y fundamental consecuencia de haber elegido $\delta - 1$ potencias sucesivas de un elemento α como raíces del polinomio generador es que la distancia del código nunca es inferior a δ. Esto explica la terminología «distancia de diseño» para el parámetro δ y justifica la utilidad de esta clase de códigos.

Teorema 10.1 (Cota de distancia BCH). *Dados dos números enteros $b \geq 0$, $\delta \geq 1$, un código cíclico con un polinomio generador $g(x)$ que verifique*

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

tiene distancia mínima al menos δ.

Demostración. Se va a probar directamente que, en las condiciones del enunciado, no existe ninguna palabra del código no nula de peso menor que δ, la distancia de diseño.

Sea $\mathbf{v} = (v_{n-1}, \ldots, v_1, v_0)$ un vector del código y $v(x) = \sum_{i=0}^{n-1} v_i x^i$ su polinomio asociado. Las $\delta - 1$ ecuaciones

$$v(\alpha^b) = v(\alpha^{b+1}) = \cdots = v(\alpha^{b+\delta-2}) = 0$$

son lineales en el cuerpo $\mathbb{F}_{q^m}$ (m es el menor índice que cumple $q^m - 1 \bmod n = 0$), y se pueden escribir en forma matricial como

$$\mathbf{v} \cdot Q^T = \mathbf{0}$$

si se conviene en definir la matriz Q como

$$Q = \begin{pmatrix} \alpha^{b(n-1)} & \ldots & \alpha^b & 1 \\ \alpha^{(b+1)(n-1)} & \ldots & \alpha^{b+1} & 1 \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(b+\delta-2)(n-1)} & \ldots & \alpha^{b+\delta-2} & 1 \end{pmatrix}.$$

Q no tiene por qué ser idéntica a la matriz de comprobación de paridad del código, sino que, en general, la matriz de comprobación de paridad es una submatriz de filas suya.

Para probar que la distancia vale al menos δ, es suficiente con mostrar que ningún conjunto de $\delta-1$ o menos columnas de la matriz Q son linealmente dependientes sobre el cuerpo $\mathbb{F}_{q^m}$. Supóngase que $\mathbf{v}$ tuviese peso Hamming menor que δ, es decir, que menos de δ de sus símbolos, $v_{i_1}, v_{i_2}, \ldots, v_{i_r}$ ($r < \delta$), fuesen distintos de cero. Si así ocurriera, la condición $\mathbf{v} \cdot Q^T = \mathbf{0}$ implicaría que

$$(v_{i_1}, v_{i_2}, \ldots, v_{i_r}) \cdot \begin{pmatrix} \alpha^{bi_1} & \alpha^{bi_2} & \ldots & \alpha^{bi_r} \\ \alpha^{(b+1)i_1} & \alpha^{(b+1)i_2} & \ldots & \alpha^{(b+1)i_r} \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(b+r-1)i_1} & \alpha^{(b+r-1)i_2} & \ldots & \alpha^{(b+r-1)i_r} \end{pmatrix}^T = \mathbf{0}$$

para ciertos escalares $v_{i_1}, \ldots, v_{i_r}$ no simultáneamente nulos. Observe que la matriz anterior es cuadrada, puesto que se han eliminado en ella las columnas de Q que no se utilizan en la evaluación de esas r ecuaciones de comprobación de paridad, esto es, se ha prescindido de las columnas que no son $i_1, \ldots, i_r$.

Pero si este sistema homogéneo de r ecuaciones lineales tuviera solución no trivial, entonces el determinante de

$$A = \begin{pmatrix} \alpha^{bi_1} & \alpha^{bi_2} & \ldots & \alpha^{bi_r} \\ \alpha^{(b+1)i_1} & \alpha^{(b+1)i_2} & \ldots & \alpha^{(b+1)i_r} \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(b+r-1)i_1} & \alpha^{(b+r-1)i_2} & \ldots & \alpha^{(b+r-1)i_r} \end{pmatrix}$$

debería ser nulo. Ahora bien, sacando factor común α^{bi_1} en la primera columna, α^{bi_2} en la segunda, ..., α^{bi_r} en la última, el determinante de A vale

$$\begin{aligned} \det A &= \alpha^{b(i_1+i_2+\cdots+i_r)} \cdot \det \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha^{i_1} & \alpha^{i_2} & \ldots & \alpha^{i_r} \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(r-1)i_1} & \alpha^{(r-1)i_2} & \ldots & \alpha^{(r-1)i_r} \end{pmatrix} \\ &= \alpha^{b(i_1+i_2+\cdots+i_r)} \cdot \det V_r. \end{aligned}$$

Esta última matriz, V_r, es una matriz de Vandermonde; y es un hecho bien conocido que su determinante es no nulo si y solamente si los elementos α^{i_1}, α^{i_2}, ..., α^{i_r} son todos distintos entre sí, pues si a cada una de sus filas

$j = 2, 3, \ldots, r$ le sumamos la anterior multiplicada por $-\alpha^{i_1}$, entonces

$$\begin{aligned}\det V_r &= \det \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha^{i_1} & \alpha^{i_2} & \ldots & \alpha^{i_r} \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(r-1)i_1} & \alpha^{(r-1)i_2} & \ldots & \alpha^{(r-1)i_r} \end{pmatrix} \\ &= \det \begin{pmatrix} 1 & 1 & \ldots & 1 \\ 0 & \alpha^{i_2} - \alpha^{i_1} & \ldots & \alpha^{i_r} - \alpha^{i_1} \\ 0 & \alpha^{i_2}(\alpha^{i_2} - \alpha^{i_1}) & \ldots & \alpha^{i_r}(\alpha^{i_r} - \alpha^{i_1}) \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \alpha^{(r-2)i_2}(\alpha^{i_2} - \alpha^{i_1}) & \ldots & \alpha^{(r-2)i_r}(\alpha^{i_r} - \alpha^{i_1}) \end{pmatrix}.\end{aligned}$$

Desarrollándolo por los elementos de la primera columna y sacando de nuevo factor común en cada una de las otras columnas, resulta

$$\begin{aligned}\det V_r &= \prod_{j=2}^{r} (\alpha^{i_j} - \alpha^{i_1}) \det \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \alpha^{i_2} & \alpha^{i_3} & \ldots & \alpha^{i_r} \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(r-2)i_2} & \alpha^{(r-2)i_2} & \ldots & \alpha^{(r-2)i_r} \end{pmatrix} \\ &= \prod_{j=2}^{r} (\alpha^{i_j} - \alpha^{i_1}) \det V_{r-1}.\end{aligned}$$

Y de aquí es fácil deducir, iterando la expresión, que el determinante de la matriz de Vandermonde vale, por tanto,

$$\det V_r = \prod_{k=1}^{r-1} \prod_{j=k+1}^{r} (\alpha^{i_j} - \alpha^{i_k})$$

y que es distinto de cero por ser todas las potencias de α distintas entre sí.

Esta conclusión es incompatible con la premisa de existencia de un vector del código con peso menor que δ, como se quería probar. ▶

Conviene hacer dos observaciones acerca de este teorema. En primer lugar, el resultado es válido para *cualquier* código cíclico con $\delta - 1$ ceros cíclicamente consecutivos: los códigos BCH son tan sólo un subconjunto de todos los que satisfacen esta condición. En segundo lugar, resulta de indiscutible utilidad, porque establece una condición de diseño entre la distancia de un código cíclico y (los ceros de) su polinomio generador. Sin embargo, este conjunto de ceros depende de la elección del elemento primitivo del cuerpo. Si, por ejemplo, se toma como elemento primitivo a $\beta = \alpha^s$, en donde s y n son coprimos, entonces β es también una raíz primitiva n-ésima de la unidad. Y si s^{-1} es el inverso de s módulo n los polinomios mínimos

$M_{\alpha^r}(x)$ y $M_{\beta^{s^{-1}r}}(x)$ son idénticos, lo que significa que un conjunto de ceros T referido al elemento primitivo α es el conjunto de ceros $s^{-1}T \mod n$ referido al elemento primitivo β. La cota BCH, por consiguiente, puede producir resultados diferentes según cuál sea el elemento primitivo elegido para definir el cuerpo. Si bien existen diversos resultados que proporcionan, en algunos casos, estimaciones mejores que la cota BCH (por ejemplo, las cotas de Hartmann–Tzeng–Roos [41] y van Lint–Wilson [69]), determinar la distancia de un código cíclico, conocidos sus ceros, es aún un importante problema algebraico abierto.

Ejemplo 10.2. Sea $\mathcal{C}$ el código cíclico binario de longitud 7 definido por los ceros $\{\alpha^i : i \in T = \{0, 3, 6, 5\}\}$, con α una raíz primitiva séptima de la unidad. T contiene tres elementos consecutivos $\{5, 6, 0\}$ y $\mathcal{C}$ es un código con distancia de diseño 4. $\mathcal{C}$ resulta ser el subcódigo de palabras de peso par de $\mathcal{H}_3$.

Sea $\mathcal{D}$ el código $[31, 25, d]$ binario definido por los ceros $\{\alpha^i : i \in T\}$, con $T = \{0, 3, 6, 12, 24, 17\}\}$ y α un elemento primitivo de $\mathbb{F}_{32}$. Para este conjunto de ceros, la cota BCH produce $d \geq 2$. Sin embargo, tomando como elemento primitivo a $\beta = \alpha^3$, y dado que $3^{-1} \equiv 21 \mod 31$, el conjunto de ceros se convierte en $\{\beta^i : i \in \{0, 1, 2, 4, 8, 16\}\}$ que contiene 3 elementos consecutivos e implica que $d \geq 4$. De hecho, $\mathcal{D}$ es el subcódigo de peso par del código Hamming binario $\mathcal{H}_5$ y $d = 4$. ■

Ejemplo 10.3. En el capítulo 7, apartado 7.7, se vio que el código cíclico generado por un polinomio primitivo binario de grado r tiene distancia 3 y es el código Hamming binario $\mathcal{H}_r$. Vamos a deducir este mismo hecho por otro camino, y a probar, de paso, que los códigos Hamming binarios están incluidos en la clase de los BCH binarios.

Supóngase $g(x)$ primitivo y binario de grado r, y $\alpha \in \mathbb{F}_{2^r}$ un elemento primitivo. Entonces $g(\alpha) = g(\alpha^2) = 0$, ya que, por definición, $g(x)$ es el polinomio mínimo de α y α^2. El código $[2^r - 1, 2^r - r - 1]$ engendrado por $g(x)$ posee un par de ceros consecutivos; en consecuencia es, por definición, un código BCH binario primitivo, y su distancia es mayor o igual a 3, según el teorema 10.1. Un vector suyo se define por la condición $v(\alpha) = \sum_{i=0}^{n-1} v_i \alpha^i = 0$, o bien

$$(v_{n-1}, \ldots, v_1, v_0) \cdot \begin{pmatrix} \alpha^{n-1} & \alpha^{n-2} & \ldots & \alpha & 1 \end{pmatrix}^T = 0$$

en $\mathbb{F}_{2^r}$. Sea $\phi : \mathbb{F}_{2^r} \longrightarrow \mathbb{F}_2^r$ la aplicación que a cada elemento de $\mathbb{F}_{2^r}$ le hace corresponder sus r coordenadas en cierta base. Si en la matriz

$$\begin{pmatrix} \alpha^{n-1} & \alpha^{n-2} & \ldots & \alpha & 1 \end{pmatrix}$$

cada uno de sus elementos α^i se reemplaza por el vector columna binario de longitud r, $\phi(\alpha^i)$, se obtiene una matriz de dimensiones $r \times (2^r - 1)$ cuyas

columnas recorren el conjunto de vectores binarios no nulos de r elementos. De donde se sigue que la distancia del código engendrado por $g(x)$ es 3 y que es (equivalente a) un código Hamming. ■

EJEMPLO 10.4. El teorema 10.1 se puede utilizar directamente para construir códigos BCH binarios primitivos correctores de dos (o más) errores, generalizando así las propiedades de corrección de los códigos Hamming.

Considérese, por fijar un caso concreto, $q = 2$, $b = 1$, y una raíz primitiva $\alpha \in \mathbb{F}_{2^4}$ de $x^{15}-1$. Del requisito de corrección de errores dobles se desprende que la distancia del código debe ser al menos 5, y entonces el código BCH que se busca tendrá que tener 4 ceros consecutivos: α, α^2, α^3 y α^4. Por ser el código primitivo, α es elemento primitivo del cuerpo $\mathbb{F}_{2^4}$, y el polinomio binario de menor grado del que es raíz es su polinomio mínimo (y primitivo) $M_\alpha(x) = x^4 + x + 1$. También son raíces de $M_\alpha(x)$ los elementos α^2 y α^4. El polinomio mínimo de α^3 es

$$M_{\alpha^3}(x) = x^4 + x^3 + x^2 + x + 1,$$

por supuesto irreducible y sin factores en común con $M_\alpha(x)$. Luego el polinomio producto de ambos

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

cumple con todas las condiciones exigibles para poder afirmar de él que engendra un código cíclico binario $[15, 7]$ capaz de corregir dos errores. Sus polinomios $v(x)$ son aquéllos y sólo aquéllos que satisfacen $v(\alpha) = v(\alpha^2) = v(\alpha^3) = v(\alpha^4) = 0$. Pero, como $v(\alpha^2) = v^2(\alpha)$ en cualquier cuerpo de característica 2, entonces $v(\alpha^4) = v(\alpha^2) = 0 \Leftrightarrow v(\alpha) = 0$, y las ecuaciones $v(\alpha^4) = v(\alpha^2) = 0$ son redundantes. O, lo que es igual, si el polinomio del código admite una raíz α, entonces también serán raíces suyas los elementos conjugados de α. Por tanto, las dos únicas ecuaciones de comprobación de paridad son $v(\alpha) = v(\alpha^3) = 0$, esto es, $\mathbf{v} \cdot H^T = \mathbf{0}$ operando en el cuerpo $\mathbb{F}_{2^4}$. En esta expresión, la matriz H de comprobación de paridad es

$$\begin{pmatrix} \alpha^{14} & \alpha^{13} & \alpha^{12} & \alpha^{11} & \alpha^{10} & \alpha^9 & \alpha^8 & \alpha^7 & \alpha^6 & \alpha^5 & \alpha^4 & \alpha^3 & \alpha^2 & \alpha & 1 \\ \alpha^{12} & \alpha^9 & \alpha^6 & \alpha^3 & 1 & \alpha^{12} & \alpha^9 & \alpha^6 & \alpha^3 & 1 & \alpha^{12} & \alpha^9 & \alpha^6 & \alpha^3 & 1 \end{pmatrix}.$$

La razón de que no aparezcan todas las potencias de α en su segunda fila es que α^3 no es un elemento primitivo.

Al sustituir cada uno de los elementos de H por su correspondiente vector columna de 4 símbolos binarios (como se recordará, el cuerpo $\mathbb{F}_{2^4}$ es un espacio vectorial de dimensión 4 sobre $\mathbb{F}_2$), se consigue una matriz

binaria de comprobación de paridad del código:

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

Este último ejemplo ilustra el proceso general para la obtención de una matriz de comprobación de paridad de un código BCH, $\mathcal{C}_{\mathbb{F}_q}$. Pues, supuesto un polinomio $v(x) \in \mathcal{C}_{\mathbb{F}_q}$, las $\delta - 1$ ecuaciones

$$v(\alpha^b) = v(\alpha^{b+1}) = \cdots = v(\alpha^{b+\delta-2}) = 0$$

son linealmente independientes en la extensión $\mathbb{F}_{q^m}$, y su expresión en forma matricial es

$$(v_{n-1}, \ldots, v_0) \cdot \begin{pmatrix} \alpha^{(n-1)b} & \ldots & \ldots & \alpha^b & 1 \\ \alpha^{(n-1)(b+1)} & \ldots & \ldots & \alpha^{b+1} & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \alpha^{(n-1)(b+\delta-2)} & \ldots & \ldots & \alpha^{b+\delta-2} & 1 \end{pmatrix}^T = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T.$$

La matriz

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

vista como una matriz de comprobación de paridad, define un código lineal $\mathcal{C}$ en $\mathbb{F}_{q^m}$. Pues bien, es obvio que los vectores de $\mathcal{C}$ compuestos sólo por símbolos de $\mathbb{F}_q$ son precisamente $\mathcal{C}_{\mathbb{F}_q}$; es decir, $\mathcal{C}_{\mathbb{F}_q}$ es el *subcódigo–subcuerpo* de $\mathcal{C}$, o la restricción a $\mathbb{F}_q$ del código $\mathcal{C}$.

Pero, además, H se puede convertir directamente en una matriz de comprobación de paridad de $\mathcal{C}_{\mathbb{F}_q}$. Para ello, basta con elegir una base de $\mathbb{F}_{q^m}$ sobre $\mathbb{F}_q$, y sustituir después cada uno de los elementos de H por el vector columna de m coordenadas que lo representa en la base elegida. H será ahora una matriz con elementos en $\mathbb{F}_q$ de dimensiones $m(\delta-1) \times n$, pero en general no se dará la circunstancia de que las $m(\delta-1)$ filas sean linealmente independientes sobre $\mathbb{F}_q$. Suprimiendo las filas linealmente dependientes de H (que representan ecuaciones de comprobación de paridad redundantes para los vectores de $\mathcal{C}_{\mathbb{F}_q}$), se llega a una matriz H' de comprobación de paridad de $\mathcal{C}_{\mathbb{F}_q}$.

TEOREMA 10.2. *El número de símbolos de información de un código BCH definido sobre $\mathbb{F}_q$, con distancia de diseño δ y longitud n es mayor o igual que $n - m(\delta - 1)$.*

DEMOSTRACIÓN. H es de dimensiones $m(\delta-1) \times n$ una vez sustituidos sus elementos por vectores de m elementos de $\mathbb{F}_q$. Pero:

- Es posible que dos o más filas representen ecuaciones redundantes; y por tanto, salvo una, las demás se pueden omitir.
- Sobre el cuerpo $\mathbb{F}_q$, no todas las filas son siempre linealmente independientes.

De estas dos observaciones se deduce que el rango de H es siempre menor o igual que $m(\delta-1)$, y entonces la dimensión del código verifica la desigualdad $k = n - \text{rango}(H) \geq n - m(\delta - 1)$. ▶

Una aplicación del razonamiento arriba expuesto nos permitirá probar que la clase de códigos BCH primitivos contiene muchos códigos Hamming.

TEOREMA 10.3. *Si $n = (q^r - 1)/(q - 1)$, r y $q - 1$ no tienen divisores comunes, y α es una raíz primitiva n-ésima de la unidad, el código BCH primitivo de longitud n generado por $M_\alpha(x)$ es el código Hamming de r símbolos de redundancia sobre $\mathbb{F}_q$.*

DEMOSTRACIÓN. Sea γ un elemento primitivo de $\mathbb{F}_{q^r}$ y $\alpha = \gamma^{q-1}$ una raíz primitiva n-ésima de la unidad. Los escalares de $\mathbb{F}_q$ en $\mathbb{F}_{q^r}$ son aquellas potencias de γ cuyo exponente divide a n. Ahora bien, puesto que se puede escribir

$$n = (q-1)\sum_{j=1}^{r-1} j q^{r-1-j} + r$$

es obvio que $\text{mcd}(n, q-1) = \text{mcd}(r, q-1) = 1$ y, por tanto, que el único elemento de $\mathbb{F}_q$ que resulta ser una potencia de α es la identidad. Entonces, si se escriben los elementos de $\mathbb{F}_{q^r}$ como r-tuplas de $\mathbb{F}_q^r$, ninguna de las r-tuplas correspondientes a $\alpha^0, \ldots, \alpha^{n-1}$ son proporcionales entre sí usando los elementos de $\mathbb{F}_q$ como escalares. La matriz $r \times n$ que tiene por columnas precisamente las r-tuplas de $\alpha^0, \ldots, \alpha^{n-1}$ es, así, la matriz de comprobación de paridad del código Hamming de r símbolos de redundancia sobre $\mathbb{F}_q$. ▶

10.2. Códigos BCH binarios

Con motivo del gran número de situaciones prácticas en que se utilizan, parece oportuno estudiar con atención especial las propiedades de los códigos BCH binarios. Fijemos, por lo tanto, $q = 2$, y supongamos un código con distancia de diseño δ y definido por el conjunto de ceros $\cup_{j=1}^{\delta-1}\mathcal{A}_j$, en donde $\mathcal{A}_j$ es el conjunto 2-ciclotómico módulo n.

Dado que $\mathcal{A}_j = \mathcal{A}_{2j \mod n}$, el polinomio generador del código con distancia de diseño δ es el mínimo común múltiplo de los polinomios mínimos de las potencias impares de α:

$$g(x) = \text{mcm}\{M_\alpha(x), M_{\alpha^3}(x), \ldots, M_{\alpha^{2t-1}}(x)\}, \quad t = \left\lfloor \frac{\delta - 1}{2} \right\rfloor.$$

En rigor, basta considerar tan sólo los códigos de distancia de diseño $\delta = 2t+1$ impar, debido a que el código de distancia de diseño $2t$ también tiene a

$$g(x) = \text{mcm}\{M_\alpha(x), M_{\alpha^3}(x), \ldots, M_{\alpha^{2t-1}}(x)\}$$

como polinomio generador, y es, obviamente, idéntico al primero. Como resultado de la estructura del polinomio generador, es sencillo deducir una cota inferior a la dimensión del código mejor que la dada en el teorema 10.2.

Teorema 10.4. *El número de símbolos de información en un código BCH binario de longitud n y distancia de diseño $\delta = 2t$ o $\delta = 2t+1$ es como mínimo $n - mt$.*

Demostración. El polinomio $g(x)$ es el mínimo común múltiplo de t polinomios mínimos binarios de elementos de $\mathbb{F}_{2^m}$, cada uno de los cuales es de grado $\leq m$ (teorema 9.10). Luego el grado de $g(x)$ nunca es mayor que mt, y por consiguiente la dimensión del código no es inferior a $n - mt$. ▶

La matriz de comprobación de paridad de un código BCH binario de longitud n y distancia de diseño $2t+1$ es de la forma

$$H = \begin{pmatrix} \alpha^{n-1} & \ldots & \alpha & 1 \\ \alpha^{3(n-1)} & \ldots & \alpha^3 & 1 \\ \ldots & \ldots & \ldots & \ldots \\ \alpha^{(2t-1)(n-1)} & \ldots & \alpha^{2t-1} & 1 \end{pmatrix}$$

ya que las filas con conjuntos de exponentes pertenecientes al mismo conjunto 2-ciclotómico módulo n son redundantes; esto es,

$$j = i2^s \Rightarrow v(\alpha^j) = 0 \Leftrightarrow v\big((\alpha^i)^{2^s}\big) = 0 \Leftrightarrow v(\alpha^i)^{2^s} = 0 \Leftrightarrow v(\alpha^i) = 0.$$

n	Distancia de diseño	Polinomio generador	(k, d)
7	1	1	(7, 1)
	3	$x^3 + x + 1$	(4, 3)
	5,7	$(x^3 + x + 1)(x^3 + x^2 + 1)$	(1, 7)
15	1	1	(15, 1)
	3	$g_1(x) = x^4 + x + 1$	(11, 3)
	5	$g_2(x) = g_1(x)(x^4 + x^3 + x^2 + x + 1)$	(7, 5)
	7	$g_3(x) = g_2(x)(x^2 + x + 1)$	(5, 7)
	9, 11, 13, 15	$g_4(x) = g_3(x)(x^4 + x^3 + 1)$	(1, 15)
31	1	1	(31, 1)
	3	$g_1(x) = x^5 + x^2 + 1$	(26, 3)
	5	$g_2(x) = g_1(x)(x^5 + x^4 + x^3 + x^2 + 1)$	(21, 5)
	7	$g_3(x) = g_2(x)(x^5 + x^4 + x^2 + x + 1$	(16, 7)
	9, 11	$g_4(x) = g_3(x)(x^5 + x^3 + x^2 + x + 1)$	(11, 11)
	13, 15	$g_5(x) = g_4(x)(x^5 + x^4 + x^3 + x + 1)$	(6, 15)
	17, 19, ..., 31	$g_6(x) = g_5(x)(x^5 + x^3 + 1)$	(1, 31)

Tabla 10.1. Códigos BCH binarios de longitudes 7, 15 y 31.

Por ejemplo, el código BCH binario de longitud 15 y distancia de diseño 11 es el engendrado por

$$g(x) = \text{mcm}\{M_\alpha(x), M_{\alpha^3}(x), M_{\alpha^5}(x), M_{\alpha^7}(x), M_{\alpha^9}(x)\}.$$

Pero los polinomios mínimos de α^3 y α^9 son el mismo (porque 3 y 9 pertenecen al mismo conjunto ciclotómico módulo 15), y $g(x)$ se reduce a

$$g(x) = \text{mcm}\{M_\alpha(x), M_{\alpha^3}(x), M_{\alpha^5}(x), M_{\alpha^7}(x)\}$$

que es el mismo que el del código BCH binario de distancia de diseño 9, e igual asimismo al generador de los códigos de distancias de diseño 13 y 15. Este sencillo ejemplo enseña que códigos BCH con distancias de diseño diferentes no tienen por qué ser distintos.

Ejemplo 10.5. La tabla 10.1 clasifica los posibles códigos BCH binarios primitivos de longitudes 7, 15 y 31, junto con su dimensión y su distancia. El código $[7, 4, 3]$ es $\mathcal{H}_3$, y el $[7, 1, 7]$ es un simple código de repetición. ■

Definición 10.3 (Distancia de Bose). *La mayor de las distancias de diseño de un conjunto de códigos BCH idénticos es la distancia de Bose del código.*

Entonces, a tenor del teorema 10.1, la distancia real del código es como mínimo su distancia de Bose, y puede en ciertos casos ser mayor que ésta.

Otro ejemplo en que así ocurre es el código BCH no primitivo de longitud 23 y distancia de diseño 5, con polinomio generador

$$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1$$

ceros $\{\alpha^i \mid i \in \{1, 2, 4, 8, 16, 9, 18, 13, 3, 6, 12\}\} = \{\alpha^i \mid i \in \mathcal{A}_1\}$ y dimensión 12. Su distancia de Bose es 5, pero no es difícil probar que se trata de un código equivalente del Golay$(23, 12)$, cuya distancia es 7. En ocasiones, conocer la distancia de Bose facilita la tarea de descubrir la distancia mínima de un código BCH, recurriendo para ello a las proposiciones del próximo apartado.

En sistemas que requieran una capacidad de corrección de errores moderada en comparación con la longitud de los mensajes, resultan especialmente adecuados los códigos BCH binarios primitivos; para $m \geq 3$ y $t \geq 1$, el código BCH binario de longitud $n = 2^m - 1$ y distancia de diseño $2t + 1$ es un código corrector de t errores con parámetros

$$\begin{aligned} n &= 2^m - 1 \\ n - k &\leq mt \\ d_{\mathcal{C}} &\geq 2t + 1. \end{aligned}$$

Si, además, $t = 2^s - 1$ para algún $s \geq 1$, entonces la distancia es exactamente $2t + 1$.

De hecho, vamos a mostrar que la distancia de un código BCH binario y primitivo siempre es impar. Sea $\mathcal{C}$ un código BCH primitivo (no necesariamente binario) y $\widehat{\mathcal{C}}$ el código extendido que se obtiene añadiendo a los vectores de $\mathcal{C}$ un símbolo de paridad, que simbolizaremos como c_∞:

$$(c_{n-1}, \ldots, c_0) \in \mathcal{C} \Rightarrow (c_\infty, c_{n-1}, \ldots, c_0) \in \widehat{\mathcal{C}}, \quad c_\infty = -\sum_{i=0}^{n-1} c_i.$$

El polinomio $c_\infty x^\infty + c_{n-1}x^{n-1} + \cdots + c_0$ representará formalmente al vector del código $(c_\infty, c_{n-1}, \ldots, c_0)$, y el elemento α^i de $\mathbb{F}_{q^m}$ a la coordenada i. Por convenio, se establece que $1^\infty = 1$ y que $\alpha^{i\infty} = 0$ para $i \not\equiv 0 \mod n$ (y así la coordenada ∞ queda representada por el elemento 0 del cuerpo).

Habiendo identificado las coordenadas de las palabras del código $\widehat{\mathcal{C}}$ con los elementos de $\mathbb{F}_{q^m}$, es obvio que cualquier permutación de éstos corresponde a una permutación en los símbolos de $\widehat{\mathcal{C}}$.

Teorema 10.5. *El grupo afín de permutaciones*

$$\begin{aligned} P_{a,b} : \mathbb{F}_{q^m} &\longrightarrow \mathbb{F}_{q^m} \\ \alpha^i &\longrightarrow a\alpha^i + b \end{aligned}$$

$(a, b \in \mathbb{F}_{q^m}, a \neq 0)$, actuando sobre las posiciones de los símbolos de las palabras del código, es un automorfismo de $\widehat{\mathcal{C}}$:

$$(c_{\alpha^\infty}, c_{\alpha^{n-1}}, \ldots, c_{\alpha^0}) \in \widehat{\mathcal{C}} \Rightarrow (c_{P_{a,b}^{-1}(\alpha^\infty)}, \ldots, c_{P_{a,b}^{-1}(\alpha^0)}) \in \widehat{\mathcal{C}}.$$

DEMOSTRACIÓN. $P_{a,0}$ es una rotación cíclica que deja la coordenada ∞ invariada.

Sea $(c'_\infty, c'_{n-1}, \ldots, c'_0)$ el vector del código $(c_\infty, c_{n-1}, \ldots, c_0)$ transformado por $P_{a,b}$, con $b \neq 0$. Entonces, si $\{\alpha, \alpha^2, \ldots, \alpha^{\delta-1}\}$ son los ceros que definen $\mathcal{C}$ e $\mathcal{I} = \{0, 1, \ldots, n-1\} \cup \{\infty\}$

$$\begin{aligned}\sum_{i\in\mathcal{I}} c'_i \alpha^{ik} &= \sum_{i\in\mathcal{I}} c_i (a\alpha^i + b)^k = \sum_{i\in\mathcal{I}} c_i \sum_{l=0}^{k} \binom{k}{l} a^l \alpha^{il} b^{k-l} \\ &= \sum_{l=0}^{k} \binom{k}{l} a^l b^{k-l} \sum_{i\in\mathcal{I}} c_i \alpha^{il} = 0, \qquad 0 \leq k \leq \delta - 1\end{aligned}$$

ya que el sumatorio interior se anula para $0 \leq l \leq \delta - 1$. En consecuencia, $(c'_\infty, c'_{n-1}, \ldots, c'_0) \in \widehat{\mathcal{C}}$. Observe que, para que el grupo afín deje a $\widehat{\mathcal{C}}$ invariado, es esencial que $\mathcal{C}$ sea primitivo (esto es, que α sea un elemento primitivo). ▶

TEOREMA 10.6. *Un código BCH binario primitivo tiene distancia impar.*

DEMOSTRACIÓN. Por el teorema anterior, el código extendido es invariante ante la acción del grupo afín de permutaciones. En particular, el conjunto de palabras de $\widehat{\mathcal{C}}$ de peso mínimo es invariante. Ahora bien, siempre se podrá encontrar una permutación del grupo afín que traslade un 1 a la posición ∞. Luego $\widehat{\mathcal{C}}$ contiene vectores con $c_\infty = 1$, lo que sólo puede ocurrir si la distancia de $\mathcal{C}$ es impar. ▶

La extensión de un código BCH binario primitivo produce siempre un código con distancia una unidad mayor. Sin embargo, advierta que, en general, la extensión de un código BCH ni incrementa la distancia ni produce un código cíclico.

10.3. Dimensión y distancia

Los teoremas 10.1 y 10.2 han dejado abiertas dos cuestiones: ¿en qué condiciones es la distancia de un código BCH igual a su distancia de diseño?; y, conocidas la longitud y la distancia de diseño, ¿cuál es la dimensión de un código BCH?

Respecto a la primera, a pesar de que se conocen una variedad de casos particulares en los que la distancia real y la de diseño coinciden, se ignora por el momento qué condiciones son necesarias y suficientes para lograrlo. Con todo, algunas proposiciones de este tipo resultan muy útiles.

Teorema 10.7. *El código BCH binario de longitud $n = 2^m - 1$ y distancia de diseño $2t + 1$ tiene distancia $2t + 1$ si*

$$\sum_{i=0}^{t+1} \binom{2^m - 1}{i} > 2^{mt}.$$

Demostración. Supóngase que la distancia del código es impar (por el teorema 10.6) y mayor que $2t+1$. La dimensión del código es como mínimo $n - mt$ (véase el teorema 10.4), de modo que la desigualdad de Hamming implica que

$$2^{n-mt} \sum_{i=0}^{t+1} \binom{2^m - 1}{i} \leq 2^n$$

lo que contradice la desigualdad del enunciado. Por lo tanto, si aquélla se cumple, $d_{\mathcal{C}} = 2t + 1$. ▶

Teorema 10.8. *Si $n = rs$, el código BCH de longitud n y distancia de diseño r tiene distancia r.*

Demostración. Siendo α una raíz primitiva n-ésima de la unidad, $\alpha^{js} \neq 1$ para todo $j < r$. En consecuencia, los elementos $\alpha, \alpha^2, \ldots, \alpha^{r-1}$ no pueden ser raíces del polinomio $x^s - 1$. Pero

$$x^n - 1 = x^{rs} - 1 = (x^s - 1)(x^{s(r-1)} + \cdots + x^{2s} + x^s + 1)$$

y $\alpha, \ldots, \alpha^{r-1}$ deben, pues, ser raíces de $v(x) = x^{s(r-1)} + \cdots + x^{2s} + x^s + 1$; luego $v(x)$ es una palabra del código de peso r, y el código tiene distancia r exactamente. ▶

La prueba de la siguiente proposición es relativamente complicada y se omite. Puede encontrarse en [41].

Teorema 10.9. *Un código BCH sobre $\mathbb{F}_q$ de longitud $q^m - 1$ y distancia de diseño $q^r - 1$ tiene distancia $q^r - 1$.*

Por otro lado, la distancia real de un código BCH no puede ser arbitrariamente mayor que la distancia de diseño. La siguiente cota superior sólo posee alguna utilidad si δ y el alfabeto de codificación son pequeños.

TEOREMA 10.10. *La distancia de un código BCH primitivo sobre $\mathbb{F}_q$ con distancia de diseño δ no es mayor que $q\delta - 1$. En particular, la distancia de un código BCH binario primitivo con distancia de diseño δ está en el rango $\delta \leq d_{\mathcal{C}} \leq 2\delta - 1$.*

DEMOSTRACIÓN. Sea r_0 el menor entero tal que $q^{r_0} - 1 > \delta \geq q^{r_0-1}$. El código de distancia de diseño δ contiene al de distancia de diseño $q^{r_0} - 1$; pero, según el teorema anterior, la distancia de éste último es exactamente $q^{r_0} - 1$. Por lo tanto, $d_{\mathcal{C}} \leq q^{r_0} - 1 \leq q\delta - 1$. ▶

El cálculo de la dimensión entraña menor dificultad. Dado un código BCH primitivo sobre $\mathbb{F}_q$ de longitud n y polinomio generador $g(x)$, su dimensión será $n -$ grado de $g(x) = n -$ rango(H). Pero, por definición, $g(x)$ es el polinomio de menor grado que admite a $\alpha, \ldots, \alpha^{\delta-1}$ y sus conjugados como ceros. Así, para hallar el grado de $g(x)$ es suficiente con contar cuántos conjugados distintos tiene cada raíz α^j, $j = 1, \ldots, \delta - 1$, y sumar estas cantidades. El número de conjugados de α^j (incluido él mismo) es por supuesto el número de elementos del conjunto ciclotómico de representante j módulo $q^m - 1$, e igual también al grado del polinomio mínimo de α^j. Si m es el orden de α^j, el número de sus conjugados es también el orden multiplicativo de j módulo m.

Para contar los elementos del conjunto ciclotómico de representante j, se puede escribir el número entero j en base q como una secuencia de m símbolos. La operación de multiplicar j por q (módulo $q^m - 1$) equivale a un desplazamiento cíclico hacia la izquierda de esa representación. Así pues, el número de desplazamientos cíclicos distintos de j, expresado en base q, es el tamaño buscado del conjunto ciclotómico. El teorema siguiente resume esta argumentación.

TEOREMA 10.11. *El grado de $g(x)$ es igual al número de enteros en el intervalo $1 \leq i \leq q^m - 1$ tales que algún desplazamiento cíclico de su representación en base q es menor o igual que $\delta - 1$.*

DEMOSTRACIÓN. Un poco de reflexión debe convencer al lector de que el enunciado equivale a afirmar que α^i es una raíz de $g(x)$, puesto que existe un conjugado suyo, α^{iq^r}, con $1 \leq iq^r \leq \delta - 1$, que lo es. ▶

EJEMPLO 10.6. Evaluando la fórmula

$$\sum_{i=0}^{t+1} \binom{2^m - 1}{i}$$

para los valores correspondientes de m y t, se obtiene la tabla 10.2. En

n	m	t	$\delta = 2t+1$	Distancia del código
15	4	1	3	3
		2	5	5
31	5	1	3	3
		2	5	5
		3	7	7
63	6	1	3	3
		2	5	5
		3	7	7

Tabla 10.2. Algunos códigos BCH con distancia igual a la de diseño.

un caso práctico general, la condición dada en el teorema 10.7 es de difícil comprobación, pues ambos miembros de la desigualdad aumentan con rapidez. De todos modos, se puede observar de manera intuitiva en la fórmula que, para valores pequeños de t (la capacidad de corrección de errores) y $n = 2^m - 1$ (la longitud), los códigos BCH con distancia de diseño $2t+1$ tienen, en efecto, distancia mínima $2t+1$. Por ejemplo, para $t = 1$

$$\sum_{i=0}^{t+1} \binom{2^m - 1}{i} = 1 + (2^m - 1) + (2^m - 1)(2^{m-1} - 1) > 2^m$$

y acontece que todo código BCH binario primitivo con distancia de diseño 3 es un código Hamming.

El código BCH de longitud 1023 y distancia de diseño 341 tiene distancia mínima 341, y el de distancia de diseño 511 tiene distancia mínima igual a 511.

Los números enteros en el rango $1 \leq i \leq 31$ con una representación binaria que, desplazada cíclicamente, dé un entero menor que 11 son 20:

00001	00010	00011	00100
00101	00110	00111	01000
01001	01010	01100	01110
10000	10001	10010	10011
10100	11000	11001	11100

El grado del polinomio generador del código BCH de longitud 31 y distancia de diseño 11 tiene entonces grado 20, y la dimensión del código es $31 - 20 = 11$. La distancia del código es 11. ■

10.4. Métodos de decodificación algebraica

Para códigos lineales, y en el supuesto de pretender hacer mínima la probabilidad de equivocación en el receptor, el método de decodificación por síndrome es, según se sabe ya, óptimo y además completo, en el sentido de que cualquiera que sea la secuencia de símbolos que se reciba, se emite siempre la mejor estimación de la palabra del código transmitida. En la práctica, no obstante, la construcción de dispositivos decodificadores de códigos lineales con una capacidad de corrección moderada o grande basados directamente en este método no es factible, a causa del tamaño que alcanza dicha tabla. La de un código BCH$[32, 28]$ con símbolos en $\mathbb{F}_{2^8}$ (corrector de errores dobles) requiere, por ejemplo, 2^{32} entradas.

Cabe, empero, una estrategia de decodificación diferente con la que soslayar esta limitación. Consiste en realizar un decodificador que resuelva acerca de la palabra del código emitida sólo cuando estima que la secuencia más probable de errores de transmisión tiene peso Hamming ρ o menor. En otro caso (supuesto que se infieren más de ρ errores en la secuencia de símbolos que se acaba de recibir), el decodificador se inhibe de corregirlos y tan sólo señala el fallo. Por tanto, para aquellas secuencias recibidas que se encuentren a distancia mayor que ρ de todas las palabras del código, el decodificador no produce a su salida ninguna palabra del código. Este modo de proceder se denomina decodificación incompleta o, también, decodificación de distancia acotada, y resulta satisfactorio sólo cuando la probabilidad de que se dé una secuencia de error incorregible (entre las que se cuentan algunas con más de ρ errores) es suficientemente pequeña.

En tal caso, se puede definir el problema que afrontan los algoritmos de decodificación de distancia acotada por ρ diciendo que consiste en encontrar, supuesto un vector recibido $\mathbf{r}$, una palabra del código, $\mathbf{x}$, con $d_H(\mathbf{x}, \mathbf{r}) \leq \rho$, si esta palabra existe. Cuando $\rho \leq \lfloor (d_{\mathcal{C}} - 1)/2 \rfloor$, de existir solución al problema, ésta es única.

La decodificación incompleta se puede explicar de un modo muy gráfico en términos de la matriz típica: basta con eliminar en ella las filas en las que el vector de la primera columna es uno de peso mayor que ρ; ahora, si el vector recibido aparece en esta nueva tabla, se decodifica como la palabra del código que encabeza su columna; en otro caso, se declara un error de decodificación.

A consecuencia de sus propiedades algebraicas, se dispone de algoritmos eficientes (con una complejidad no exponencial) de decodificación completa de los códigos BCH correctores de errores dobles y triples. En cambio, se desconoce todavía si existen también para otros códigos BCH de mayor capacidad de corrección. Como quiera que estos últimos son precisamente los que más interesan en las aplicaciones, se han desarrollado para ellos

varios algoritmos específicos de decodificación incompleta, cuyos fundamentos explicaremos en este apartado. Por el momento, sólo se va a analizar la decodificación de códigos binarios, y se pospone la de los no binarios al capítulo 11.

Considérese un código BCH binario de longitud n y distancia de diseño $\delta = 2t + 1$ impar. Su matriz de comprobación de paridad en la extensión $\mathbb{F}_{2^m}$ es, como se sabe,

$$H = \begin{pmatrix} \alpha^{n-1} & \dots & \dots & \alpha & 1 \\ \alpha^{3(n-1)} & \dots & \dots & \alpha^3 & 1 \\ \dots & \dots & \dots & \dots & \dots \\ \alpha^{(\delta-2)(n-1)} & \dots & \dots & \alpha^{\delta-2} & 1 \end{pmatrix}.$$

Sea $\omega = (\omega_{n-1}, \dots, \omega_0)$ la secuencia binaria recibida en el extremo de salida del canal. El síndrome que corresponde a ω es

$$\omega \cdot H^T = \big(\omega(\alpha), \omega(\alpha^3), \dots, \omega(\alpha^{\delta-2})\big) \tag{10.1}$$

siendo $\omega(x) = \omega_{n-1}x^{n-1} + \cdots + \omega_1 x + \omega_0$. No se olvide que, en esta expresión, el producto del vector ω por la matriz H^T se realiza con las operaciones de suma y producto del cuerpo $\mathbb{F}_{2^m}$, y que las componentes del síndrome pertenecen también a $\mathbb{F}_{2^m}$.

La ecuación (10.1) muestra que los elementos del vector síndrome son los valores del polinomio $\omega(x)$ (asociado al vector recibido ω) evaluado en los puntos α, α^3, ..., $\alpha^{\delta-2}$. Sin embargo, existe una manera más eficiente que ésta de calcular el síndrome a partir de $\omega(x)$. Porque, si se divide $\omega(x)$ entre el polinomio mínimo $M_{\alpha^i}(x)$ del elemento α^i, entonces se puede escribir

$$\omega(x) = q(x)M_{\alpha^i}(x) + r_i(x)$$

y de aquí se desprende que $\omega(\alpha^i) = r_i(\alpha^i) \quad \forall i = 1, \dots, \delta - 1$. Esto es, el elemento j-ésimo del vector de síndromes es el valor del polinomio resto $r_{2j-1}(x)$ evaluado en el punto α^{2j-1}. Ahora bien, el valor $r(\alpha^{2j-1})$ se puede calcular fácilmente con un circuito divisor de polinomios, para obtener primero el resto, seguido de un sencillo circuito lógico combinacional.[2] Este método de división reduce el número de operaciones del anterior, dado que el resto $r_i(x)$ es siempre un polinomio de grado menor que $M_{\alpha^i}(x)$ y, claro está, también menor o igual que $\omega(x)$. Veamos con un ejemplo cómo se sintetizarían los circuitos de cómputo del síndrome.

[2] Con códigos binarios, este circuito combinacional solamente requiere sumadores binarios. Con códigos no binarios harían falta dispositivos sumadores y multiplicadores en $\mathbb{F}_q$.

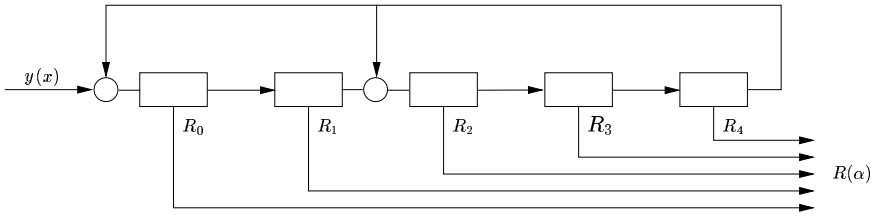


FIGURA 10.1. Divisor por $x^5 + x^2 + 1$ y cálculo de $R(\alpha)$.

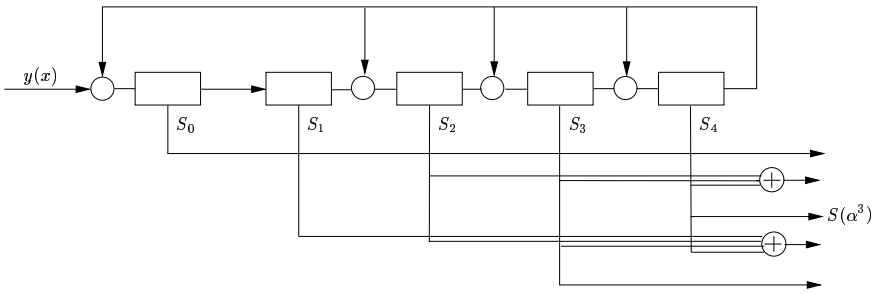


FIGURA 10.2. Divisor por $x^5 + x^4 + x^3 + x^2 + 1$ y cálculo de $S(\alpha^3)$.

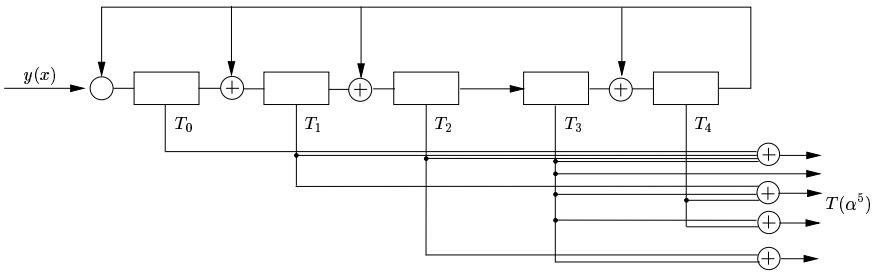


FIGURA 10.3. Divisor por $x^5 + x^4 + x^2 + x + 1$ y cálculo de $T(\alpha^5)$.

Ejemplo 10.7. El código BCH$[31, 16]$, binario y primitivo, de polinomio generador

$$\begin{aligned} g(x) &= M_\alpha(x)M_{\alpha^3}(x)M_{\alpha^5}(x) \\ &= (x^5+x^2+1)(x^5+x^4+x^3+x^2+1)(x^5+x^4+x^2+x+1) \end{aligned}$$

tiene distancia 7.

Sus síndromes son vectores de tres elementos del cuerpo $\mathbb{F}_{2^5}$; el primer elemento es $R(\alpha)$, en donde $R(x)$ es el resto de dividir la secuencia recibida $\omega(x)$ entre $M_\alpha(x)$:

$$\omega(x) = q_1(x)M_\alpha(x) + R(x),$$

y α es un elemento primitivo de $\mathbb{F}_{2^5}$. Si se escriben, en general, los coeficientes de $R(x)$ como $R(x) = R_4x^4 + R_3x^3 + R_2x^2 + R_1x + R_0$, entonces es $R(\alpha) = R_4\alpha^4 + R_3\alpha^3 + R_2\alpha^2 + R_1\alpha + R_0$. El esquema de la figura 10.1 ilustra el circuito de división y el cálculo posterior de $R(\alpha)$.

De la misma manera, si $S(x)$ es el resto de dividir el vector recibido entre el polinomio mínimo $M_{\alpha^3}(x)$,

$$\omega(x) = q_2(x)M_{\alpha^3}(x) + S(x)$$

y se escribe $S(x) = S_4x^4 + S_3x^3 + S_2x^2 + S_1x + S_0$, entonces la segunda componente del vector síndrome es $S(\alpha^3) = S_3\alpha^4 + (S_1+S_2+S_3+S_4)\alpha^3 + S_4\alpha^2 + (S_4+S_3+S_2)\alpha + S_0$. El correspondiente circuito divisor se ha representado en la figura 10.2.

Y el último elemento del síndrome es el resto $T(x)$, valorado en α^5, de la división de $\omega(x)$ entre $M_{\alpha^5}(x)$. Su expresión general es

$$T(\alpha^5) = T_0+T_1+T_2+T_3+T_3\alpha+(T_1+T_3+T_4)\alpha^2+(T_3+T_4)\alpha^3+(T_2+T_3)\alpha^4$$

si se toma $T(x) = T_4x^4+T_3x^3+T_2x^2+T_1x+T_0$. El diagrama de la figura 10.3 muestra el correspondiente circuito de cálculo. ■

Una vez hallado el síndrome, si no fuese nulo, analicemos qué relación algebraica se da entre su valor y las posiciones erróneas. Esta dependencia explícita, si existe, podrá en algunos casos resolverse, y proporcionará un método con el que corregir los errores de transmisión.

Supongamos entonces que el vector recibido ω difiere de la palabra del código emitida en las s coordenadas $i_1, \ldots, i_s$ o, de manera equivalente, que el polinomio error vale

$$e(x) = x^{i_1} + \cdots + x^{i_s}. \tag{10.2}$$

Entonces, las componentes del síndrome están contenidas en la secuencia de $2t$ coeficientes[3]

$$S_j = \omega(\alpha^j) = v(\alpha^j) + e(\alpha^j) = e(\alpha^j), \qquad j = 1, 2, \ldots, \delta - 1$$

la cual depende sólo del vector de error $\mathbf{e}$, ya que, por definición de los códigos BCH, $v(\alpha^j) = 0$. Pero, por (10.2),

$$e(\alpha^j) = \sum_{k=1}^{s} \alpha^{j i_k}$$

que, si se define $X_k = \alpha^{i_k} \in \mathbb{F}_{2^m}$, se podrá escribir como

$$S_j = e(\alpha^j) = \sum_{k=1}^{s} X_k^j, \quad j = 1, \ldots, \delta - 1.$$

Es decir, las componentes del síndrome son una combinación lineal de las potencias sucesivas de los elementos X_k. Tal cambio de variables resulta muy conveniente porque permite ver con facilidad que, a partir de la secuencia de elementos X_k, es factible localizar las posiciones erróneas, pues para averiguarlas basta con resolver las ecuaciones

$$X_k = \alpha^{i_k}, \qquad k = 1, \ldots, s$$

por simple enumeración o consultando una tabla de logaritmos del cuerpo $\mathbb{F}_{2^m}$. Los elementos $\{X_1, X_2, \ldots, X_s\}$ indican inequívocamente las posiciones erróneas en el vector recibido, por lo que se denominan *elementos localizadores de errores*. Según lo expuesto, acabamos de ver que decodificar equivale a hallar el valor de los localizadores.

Así los hechos, necesitaremos un procedimiento para resolver el sistema de ecuaciones

$$S_j = \sum_{k=1}^{s} X_k^j, \qquad j = 1, \ldots, \delta - 1 \tag{10.3}$$

en las incógnitas $\{X_1, X_2, \ldots, X_s\}$. Este es un sistema algebraico no lineal que consta de no más de $t = \frac{\delta - 1}{2}$ ecuaciones independientes (tratamos con códigos binarios) y, por cómo se han definido las componentes del síndrome y los localizadores de error, siempre admite al menos una solución. Pero su propiedad crucial es que el conjunto de elementos $\{X_1^*, X_2^*, \ldots, X_w^*\}$ de menor longitud w que lo resuelve es único y, en consecuencia, correcto (los valores X_k^* coinciden con los localizadores del vector de error) sólo cuando el número de errores que se producen durante la transmisión, s, es menor o

[3] Observe que, en códigos binarios, al menos la mitad de los términos de la secuencia son redundantes: $S_{2j} = \omega(\alpha^{2j}) = \omega^2(\alpha^j) = S_j^2$.

igual que t (teorema 10.13). En otro caso, es decir, si $s > t$, el sistema (10.3) posee múltiples soluciones. De tal manera que, cuando el canal introduce t o menos errores, un método para resolver (10.3) es un procedimiento de decodificación incompleta del código.

La dificultad de averiguar, cuando exista, la solución única de las ecuaciones (10.3) estriba en que son no lineales y s es desconocido. Las ecuaciones presentan, no obstante, una pauta regular reconocible: los segundos miembros son funciones simétricas de suma de potencias con los localizadores X_k, para $k = 1, \ldots, s$

$$\begin{aligned} S_1 &= X_1 + X_2 + \cdots + X_s \\ S_2 &= X_1^2 + X_2^2 + \cdots + X_s^2 \\ &\vdots \\ S_{2t} &= X_1^{2t} + X_2^{2t} + \cdots + X_s^{2t}. \end{aligned}$$

A efectos de simplificar la resolución del sistema, se conviene en definir un nuevo polinomio que, primero, va a permitir transformar las ecuaciones anteriores en un sistema lineal, y, segundo, posee un conjunto de ceros que harán posible posteriormente identificar los errores. Antes, presentemos la clase a la que pertenece este nuevo polinomio.

Definición 10.4 (Polinomio localizador). *En el anillo de polinomios* $F_q[x]/x^n - 1$, *con* $\mathrm{mcd}(q, n) = 1$, *se llama polinomio localizador de* $P(x)$ *a cualquier polinomio* $L(x)$ *no nulo para el que se cumple*

$$L(x)P(x) = 0 \mod x^n - 1.$$

De modo que si $L(x)$ es el polinomio localizador de $P(x)$,

$$L(\alpha^{-i})P(\alpha^{-i}) = 0, \qquad \forall i = 0, \ldots, n-1$$

y $L(x)$ contará entre sus ceros a todas las raíces n-ésimas de la unidad, α^{-i}, que no son ceros de $P(x)$: $P(\alpha^{-i}) \neq 0 \Rightarrow L(\alpha^{-i}) = 0$. Es fácil ver que el conjunto de todos los polinomios localizadores de uno dado, $P(x)$, forman un ideal principal de $F_q[x]/x^n - 1$, generado por cualquiera de los polinomios localizadores de menor grado.

De vuelta al problema de la decodificación de un código BCH binario, y dado un vector de error $(e_1, \ldots, e_n)$ de peso s, formemos a partir de él el polinomio transformado

$$S(x) = \sum_{k=0}^{n-1} S_k x^k$$

con $S_k = \sum_{j=1}^{s} e_{i_j} \alpha^{k i_j} = \sum_{j=1}^{s} \alpha^{k i_j}$. Los coeficientes $S_1, S_2, \ldots, S_{2t}$ de este polinomio son conocidos, ya que se trata de las componentes del síndrome del vector error. Pues bien, se puede demostrar que el polinomio

$$L(x) = \prod_{j=1}^{s} (1 - X_j x)$$

con $X_j = \alpha^{i_j}$ para $j = 1, \ldots, s$, es el localizador de grado mínimo de $S(x)$. Esto es, el localizador $L(x)$ de $S(x)$ es el polinomio de grado s cuyos ceros son X_j^{-1}, los recíprocos de los elementos localizadores de errores. Los coeficientes de $L(x) = \sum_{m=0}^{s} L_m x^m$ se pueden expresar directamente en términos de los $\{X_j\}$:

$$\begin{aligned}
L_0 &= 1 \\
L_1 &= (-1) \cdot \sum_{i=1}^{s} X_j \\
L_2 &= (-1)^2 \sum_{i<j} X_i X_j \\
L_3 &= (-1)^3 \sum_{i<j<k} X_i X_j X_k \\
&\vdots \\
L_s &= (-1)^s \sum_{i_1<i_2<\cdots<i_s} X_{i_1} X_{i_2} \cdots X_{i_s} = (-1)^s \prod_{i=1}^{s} X_i
\end{aligned} \tag{10.4}$$

y resultan ser las *funciones elementales simétricas* de los localizadores. Pero en álgebra es bien conocida la existencia de un conjunto de ecuaciones lineales simultáneas que relacionan las funciones elementales simétricas de un conjunto de variables con las funciones de suma de potencias de esas mismas variables. Se trata de las identidades de Newton, y se suelen presentar en la forma directa que se enuncia a continuación.

Teorema 10.12 (Identidades de Newton). *En cualquier cuerpo* $\mathbb{K}$*, las sumas de potencias*

$$S_j = \sum_{k=1}^{s} X_k^j, \qquad j \geq 1$$

en las que $\{X_1, \ldots, X_s\} \in \mathbb{K}^s$ *son elementos distintos del cuerpo, y los coeficientes del polinomio*

$$L(z) = \prod_{j=1}^{s} (1 - X_j z) = \sum_{j=0}^{s} L_j z^j$$

satisfacen el siguiente sistema de ecuaciones lineales

$$\begin{aligned} &S_1 + L_1 = 0 \\ &S_2 + L_1 S_1 + 2L_2 = 0 \\ &\cdots\cdots\cdots\cdots\cdots\cdots \\ &S_s + L_1 S_{s-1} + \cdots + L_{s-1} S_1 + sL_s = 0 \end{aligned} \tag{10.5}$$

y

$$S_i + L_1 S_{i-1} + \cdots + L_s S_{i-s} = 0, \qquad \forall i > s. \tag{10.6}$$

En el caso de un código BCH binario, estas ecuaciones implican que

$$\begin{pmatrix} 1 & 0 & \cdots\cdots\cdots & 0 \\ S_2 & S_1 & 1 \;\; 0 \;\; \cdots\cdots & 0 \\ S_4 & S_3 & S_2 \;\; S_1 \;\; 1 \;\; \ldots & 0 \\ \cdots\cdots & \cdots\cdots & \cdots\cdots & \cdots \\ S_{2s-4} & S_{2s-5} & \cdots\cdots\cdots & S_{s-3} \\ S_{2s-2} & S_{2s-3} & \cdots\cdots\cdots & S_{s-1} \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \\ L_3 \\ \vdots \\ L_s \end{pmatrix} = \begin{pmatrix} S_1 \\ S_3 \\ S_5 \\ \vdots \\ S_{2s-1} \end{pmatrix} \tag{10.7}$$

si $S_1, S_2, \ldots, S_{\delta-1}$ son las componentes del vector síndrome, y $L_0, L_1, \ldots, L_s$ son los coeficientes del polinomio localizador de errores.

Demostración. Sea el polinomio

$$S(z) = \sum_{j=1}^{\infty} S_j z^j \quad \text{con } S_j = \sum_{k=1}^{s} X_k^j.$$

El producto de z por la derivada formal[4] de $L(z)$ vale

$$\begin{aligned} zL'(z) &= -z \sum_{j=1}^{s} X_j \prod_{i \neq j} (1 - X_i z) - z \sum_{j=1}^{s} X_j \frac{L(z)}{1 - X_j z} \\ &= -L(z) \sum_{j=1}^{s} z X_j \sum_{k=0}^{\infty} (X_j z)^k = -L(z) \sum_{k=1}^{\infty} \left(\sum_{j=1}^{s} X_j^k \right) z^k \\ &= -L(z) S(z) \end{aligned}$$

aplicando la identidad formal $\sum_{k=0}^{\infty} (X_j z)^k = 1/(1 - X_j z)$. Es decir, se tiene que $L(z)S(z) + zL'(z) = 0$. Igualando coeficientes de esta expresión, resulta el conjunto de ecuaciones (10.5)-(10.6).

[4] La noción de límite carece de sentido en un cuerpo finito.

En el caso particular de que el cuerpo base tenga característica 2, basta con introducir las simplificaciones

$$\begin{aligned} jL_j &= L_j \quad \text{si } j \text{ es impar} \\ jL_j &= 0 \quad \text{si } j \text{ es par} \\ S_{2j} &= S_j^2 \end{aligned}$$

para eliminar todas las ecuaciones pares, que son redundantes, y obtener el sistema (10.7). ▶

Por la forma en que se definen las sumas de potencias, las identidades de Newton adquieren una gran utilidad en el caso de que el código sea binario. En el problema de la decodificación, existen dos métodos diferentes de explotar las identidades de Newton para obtener los coeficientes del polinomio localizador a partir de los del síndrome.

10.4.1. Algoritmo de Peterson

En virtud del teorema 10.12, los coeficientes del síndrome y los del polinomio localizador verifican las ecuaciones (10.7). Aunque s es desconocido, el teorema siguiente proporciona un método iterativo para resolverlas. Básicamente, muestra que existe una submatriz de síndromes regular si se producen menos de t errores de transmisión.

TEOREMA 10.13 (PETERSON). *Sean $X_1, \ldots, X_s$ elementos distintos de un cuerpo $\mathbb{K}$ de característica 2 y, para $j \geq 1$, sea $S_j = \sum_{k=1}^{s} X_k^j$. La matriz $r \times r$*

$$A_r = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & \ldots & 0 \\ S_2 & S_1 & 1 & 0 & 0 & \ldots & 0 \\ S_4 & S_3 & S_2 & S_1 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ S_{2r-4} & S_{2r-5} & \ldots & \ldots & \ldots & \ldots & S_{r-3} \\ S_{2r-2} & S_{2r-3} & \ldots & \ldots & \ldots & \ldots & S_{r-1} \end{pmatrix}$$

es regular si $s = r$ o $s = r-1$, y es singular si $s < r-1$.

DEMOSTRACIÓN.

a) Si $s < r-1$ y $L_1, \ldots, L_{r-2}$ son los coeficientes del polinomio $L(x) = \prod_{i=1}^{s}(1 - xX_i)$, entonces

$$A_r \begin{pmatrix} 0 & 1 & L_1 & \ldots & L_{r-2} \end{pmatrix}^T = 0$$

en virtud de las ecuaciones de Newton (10.7), de modo que A_r es singular.

b) Si $s = r$ y fijamos $X_i = X_j$ para algún $i \neq j$, entonces $S_j = \sum_{k=1}^{s-2} X_k^j$ (la característica del cuerpo es 2), y, por el apartado anterior, A_r sería singular. Por tanto,

$$\det A_r = K \prod_{i<j} (X_i + X_j)$$

para cierta constante K. De hecho, $K = 1$, ya que, sin ir más lejos, $\det A_2 = X_1 + X_2$. Como, por hipótesis, $X_i \neq X_j$, $\det A_r \neq 0$, y A_r no es singular.

c) Si $s = r - 1$, bastaría con fijar $X_i = 0$ para comprobar, según el caso previo, que $\det A_r \neq 0$. ▶

Aplicándolo al problema que nos ocupa, si el código BCH es de distancia de diseño $\delta = 2t + 1$ y ocurren $s \leq t$ errores, entonces podemos intentar resolver (10.7) suponiendo que el número de posiciones erróneas es t, es decir, postulando $s = t$. Según el teorema anterior, si se han producido t o $t - 1$ errores, existe solución única del sistema, lo que significa que, continuando con el proceso de decodificación, será posible descubrir las posiciones erróneas. Pero si han ocurrido menos de $t - 1$ errores (o más de t), las ecuaciones (10.7) son las de un sistema indeterminado. En ese caso, supongamos que el número de errores era de $t - 2$, e intentemos de nuevo resolver (10.7) con $s = t - 2$. Repitamos el procedimiento de reducir en cada paso dos errores, eliminando las dos últimas filas de la matriz A_r, hasta llegar a una de estas dos alternativas:

a) Una única solución de las ecuaciones, en cuyo caso se continúa con el resto del algoritmo de decodificación.

b) Ninguna solución única de las ecuaciones, y entonces con certeza se sabe que deben haberse producido más de t errores. Ya que no son corregibles y se asume decodificación incompleta, la decodificación termina en este punto.

Conviene señalar que, aunque exista una solución de las ecuaciones de Peterson, ésta puede no corresponder al polinomio localizador de errores del vector transmitido. En particular, si se recibe un vector a distancia mayor que t de cualquier palabra del código, el polinomio evaluador que se calcula puede poseer raíces múltiples o que no pertenecen al cuerpo al que pertenecen las raíces n-ésimas de la unidad. Tal situación, sin embargo, se detecta con facilidad, aunque es ajena al algoritmo. También puede darse el caso de que el vector recibido se encuentre a distancia menor que t de una palabra del código incorrecta, lo que llevaría a un error de decodificación indetectable.

En resumen, los pasos del algoritmo de decodificación de Peterson para un código corrector de t errores son:

1. Calcular los elementos $\{S_j, j = 1, \ldots, 2t\}$ del síndrome.
2. Construir la matriz de síndromes A_t.
3. Hallar el mayor $s \leq t$ tal que el determinante de A_s es distinto de cero.
4. Calcular los coeficientes del polinomio localizador de errores $L(x)$ resolviendo el sistema (10.7).
5. Obtener las raíces de $L(x)$; si no son todas distintas o no pertenecen al menor cuerpo de característica q que contiene a todas las raíces n-ésimas de la unidad, entonces se han producido más de t errores de transmisión y se debe declarar un fallo de decodificación.
6. Obtener el vector error (los localizadores) a partir de $L(x)$.
7. Si el vector corregido tiene síndrome nulo, emitirlo; en caso contrario, señalar un error de decodificación.

El inconveniente de este método es que se basa en la resolución de una sucesión de $t/2$ sistemas lineales, por lo que no resulta práctico a menos que t sea pequeño ($t = 6$ o $t = 7$) y se puedan manejar fórmulas analíticas sencillas para el determinante de A_t. Por referencia, damos a continuación las expresiones formales de los coeficientes del polinomio localizador en función del síndrome en códigos correctores de errores simples, dobles y triples:

Corrección de errores simples

$$L_1 = S_1$$

Corrección de errores dobles

$$\begin{aligned} L_1 &= S_1 \\ L_2 &= \frac{S_3}{S_1} + S_1^2 \end{aligned}$$

Corrección de errores triples

$$\begin{aligned} L_1 &= S_1 \\ L_2 &= \frac{S_1^2 S_3 + S_5}{S_1^3 + S_3} \\ L_3 &= (S_1^3 + S_3) + S_1 L_2 \end{aligned}$$

Ejemplo 10.8. Con códigos BCH correctores de errores dobles, el procedimiento de decodificación de Peterson puede describirse así:

1. Calcular el síndrome (S_1, S_3)
2. Si $S_1 \neq 0$ y $S_3 = S_1^3$, entonces se ha producido un error y el polinomio localizador vale
$$L(z) = 1 + S_1 z.$$
3. Si $S_1 \neq 0$ y $S_3 \neq S_1^3$, entonces han ocurrido dos errores, y para hallar el polinomio localizador se ha de resolver
$$\begin{pmatrix} 1 & 0 \\ S_2 & S_1 \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} = \begin{pmatrix} S_1 \\ S_3 \end{pmatrix}.$$
Como su solución es
$$\begin{aligned} L_1 &= S_1 \\ L_2 &= \frac{S_3}{S_1} + S_2 = \frac{S_3}{S_1} + S_1^2 \end{aligned}$$
el polinomio localizador de errores vale
$$L(x) = \left(\frac{S_3}{S_1} + S_1^2 \right) x^2 + S_1 x + 1$$
expresado en función de los elementos del síndrome (S_1, S_3). Con la sustitución $y = S_1 x$, el cálculo de las raíces del polinomio localizador consiste en la resolución de la ecuación cuadrática
$$y^2 \left(\frac{S_3}{S_1^3} + 1 \right) + y + 1 = 0,$$
que mediante otro cambio de variable se puede escribir en la forma canónica
$$y^2 + y + u = 0$$
para cierto $u \in \mathbb{F}_{2^m}$. El método de resolución de esta clase de ecuaciones cuadráticas en $\mathbb{F}_{2^m}$ resulta ser muy sencillo, y aparece descrito en [41, cap. 9], por ejemplo.
4. Si $S_1 = 0$ y $S_3 \neq 0$, entonces han ocurrido al menos tres errores, puesto que la matriz
$$\begin{pmatrix} 1 & 0 \\ S_2 & S_1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$
es singular. ■

EJEMPLO 10.9. Consideremos el código BCH$[31, 16, 7]$ binario de polinomio generador

$$g(x) = x^{15} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^5 + x^3 + x^2 + x + 1$$

y supongamos que el vector que se recibe es

$$\widehat{v}(x) = x^{15} + x^{12} + x^6 + x^5 + 1.$$

Apliquemos el algoritmo de Peterson para proceder a la decodificación. El síndrome de $\widehat{v}(x)$ tiene por componentes a

$$\begin{aligned}
S_1 &= \widehat{v}(\alpha) = \alpha^{15} \\
S_2 &= \widehat{v}(\alpha^2) = \alpha^{30} = S_1^2 \\
S_3 &= \widehat{v}(\alpha^3) = \alpha^5 \\
S_4 &= \widehat{v}(\alpha^4) = \alpha^{29} = S_2^2 \\
S_5 &= \widehat{v}(\alpha^5) = \alpha^{15} \\
S_6 &= \widehat{v}(\alpha^6) = \alpha^{10} = S_3^2.
\end{aligned}$$

La matriz de síndromes

$$A_3 = \begin{pmatrix} 1 & 0 & 0 \\ \alpha^{30} & \alpha^{15} & 1 \\ \alpha^{29} & \alpha^5 & \alpha^{30} \end{pmatrix}$$

es no singular, y por tanto los coeficientes del polinomio localizador se obtienen tras resolver

$$A_3 \cdot \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} = \begin{pmatrix} \alpha^{15} \\ \alpha^5 \\ \alpha^{15} \end{pmatrix}.$$

La solución de este sistema de ecuaciones en $\mathbb{F}_{32}$ es $(\alpha^{15}, \alpha^2, \alpha^{27})$, de modo que el polinomio localizador vale

$$L(x) = 1 + \alpha^{15}x + \alpha^2 x^2 + \alpha^{27} x^3.$$

Las raíces de $L(x)$ son los puntos α^{-16}, α^{-7} y α^{-4}, por lo que el receptor estima un error de transmisión

$$e(x) = x^{16} + x^7 + x^4.$$

La salida del decodificador será, pues,

$$\widehat{v}(x) + e(x) = x^{16} + x^{15} + x^{12} + x^7 + x^6 + x^5 + x^4 + 1 = (x+1)g(x). \quad \blacksquare$$

10.4.2. Algoritmo de Berlekamp

El método de Peterson se sirve de una solución simple y directa para el cálculo de los coeficientes del polinomio localizador de errores: la inversión de una matriz de síndromes. Puesto que la complejidad de esta operación matricial es $O(n^3)$ (para una matriz de tamaño n), parece claro que la carga de cómputo se vuelve excesiva en códigos con una capacidad de corrección moderada o grande.

La complejidad del algoritmo de Berlekamp, en cambio, se incrementa sólo cuadráticamente con la capacidad t de corrección; es, por tanto, mucho más eficiente que el de Peterson, y permite decodificar de forma práctica códigos correctores de decenas de errores. El fundamento del algoritmo es la identidad algebraica que se expone a continuación.

Formalmente, sea

$$S(x) = \sum_{j=1}^{\infty} S_j x^j$$

el *polinomio síndrome*, del que sólo son conocidos en recepción los coeficientes de los términos de grado no superior a $2t$. El producto de $1 + S(x)$ por el polinomio localizador de errores vale

$$\begin{aligned} V(x) &= \big(1 + S(x)\big)L(x) \\ &= (1 + S_1 x + S_2 x^2 + \cdots)(1 + L_1 x + \cdots + L_s x^s) \\ &= 1 + (S_1 + L_1)x + (S_2 + L_1 S_1 + L_2)x^2 + \cdots \end{aligned}$$

Pero, como los coeficientes del polinomio síndrome y del localizador satisfacen las identidades de Newton (10.5), entonces los coeficientes impares de $V(x)$ son nulos. Puesto que solamente $S_1, \ldots, S_{2t}$ son conocidos y se requieren $L_1, \cdots, L_s$, el problema de la decodificación equivale a encontrar dos polinomios $L(x)$ y $V(x)$, de grado menor o igual que t, que verifiquen

$$\big(1 + S(x)\big)L(x) \equiv (1 + V_2 x^2 + V_4 x^4 + \cdots + V_{2t} x^{2t}) \equiv V(x^2) \mod x^{2t+1}.$$

El algoritmo de Berlekamp resuelve esta ecuación mediante un procedimiento iterativo que se compone de los pasos siguientes:

1. Inicialización: $k = 0$, $L_0(x) = 1$, $T_0(x) = 1$
2. Iteración:
   a) Calcular
   $$L_{2k+2}(x) = L_{2k}(x) + \gamma_{2k} x T_{2k}(x)$$
   donde γ_{2k} es el coeficiente de x^{2k+1} en el producto $L_{2k}(x)(1 + S(x))$.

b) Calcular

$$T_{2k+2}(x) = \begin{cases} x^2 T_{2k}(x) & \text{si } \gamma_{2k} = 0 \text{ o grado}(L_{2k}(x)) > k \\ \gamma_{2k}^{-1} x L_{2k}(x) & \text{si } \gamma_{2k} \neq 0 \text{ y grado}(L_{2k}(x)) \leq k \end{cases}$$

c) $k = k + 1$; si $k < t$ volver al paso (a).

d) Determinar las raíces de $L(x) = L_{2t}(x)$; si son todas distintas y pertenecen al cuerpo adecuado, corregir las posiciones erróneas; en otro caso, señalar un error de decodificación.

El algoritmo calcula explícitamente un $L(x)$ de grado mínimo que satisface la ecuación $(1 + S(x))L(x) \equiv V(x^2) \mod x^{2t+1}$. Es una simplificación del algoritmo de Berlekamp–Massey para la decodificación de códigos BCH no binarios, que se explica en el apartado 11.4.2.

EJEMPLO 10.10. Tomemos el código BCH$[15, 7]$ de distancia 5 y polinomio generador

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

y supongamos que se producen errores en las posiciones primera y última del vector del código que se transmite, esto es,

$$e(x) = x^{14} + 1.$$

Por tanto, el polinomio localizador de errores vale

$$L(x) = (1 - x)(1 - \alpha^{14}x) = 1 - (\alpha^{14} + 1)x + \alpha^{14}x^2 = 1 + \alpha^3 x + \alpha^{14}x^2$$

y el polinomio síndrome

$$S(x) = 1 + \alpha^3 x + \alpha^6 x^2 + \alpha^{11} x^3 + \alpha^{12} x^4 + \cdots$$

Al aplicar el algoritmo de Berlekamp, se obtiene la secuencia de polinomios que recoge la tabla siguiente

k	$L_{2k}(x)$	$T_{2k}(x)$	γ_{2k}
0	1	1	α^3
1	$1 + \alpha^3 x$	$\alpha^{12} x$	α^2
2	$1 + \alpha^3 x + \alpha^{14} x^2$	—	—

donde, en efecto, $L_4(x) = L(x)$. ■

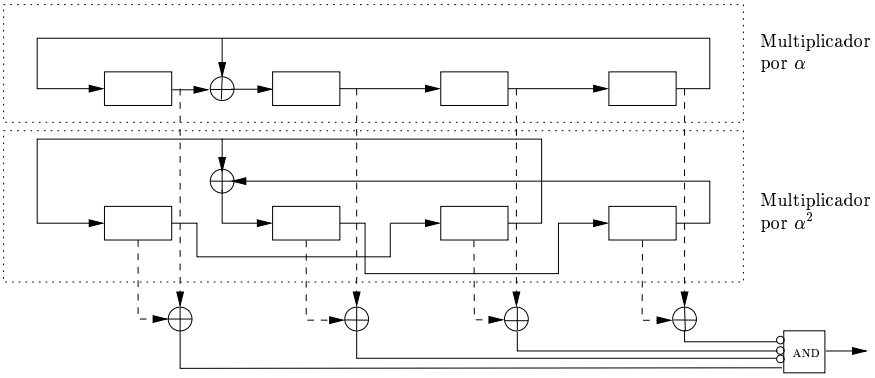


Figura 10.4. Circuito de búsqueda de Chien.

10.4.3. Búsqueda de las raíces

El último paso en la decodificación incompleta consiste en hallar las raíces del polinomio localizador de errores, que son justo los recíprocos de los localizadores. Basta para ello con comprobar si con cada uno de los elementos α^j, $j = 0, 1, \ldots, n-1$, es $L(\alpha^j) = 0$.

El mecanismo de corrección de errores es, en códigos binarios, igualmente sencillo, puesto que se habrá producido un error en el símbolo j de la secuencia recibida si, y solamente si, $L(\alpha^{-j}) = 0$. La búsqueda exhaustiva de los ceros de $L(x)$ (por simple evaluación) recibe el nombre de *búsqueda de Chien*, y se puede realizar con un circuito secuencial no demasiado complicado.

Este circuito consta de t registros de desplazamiento. El registro j es un multiplicador por α^j que se inicializa con la representación binaria del coeficiente L_j. El contenido de cada registro se itera n veces, para ir calculando $L_j\alpha^{ij}$ $(i = 0, \ldots, n-1)$. A la salida de los registros de desplazamiento, un circuito lógico detector verifica la condición $L(\alpha^{-j}) = 0$. Un ejemplo ilustrará mejor el proceso.

Ejemplo 10.11. En el ejemplo anterior se obtuvo un polinomio localizador

$$L(x) = 1 + \alpha^3 x + (\alpha^3 + 1)x^2$$

para errores en las posiciones 0 (primera) y 14 (última). El recíproco del elemento $\alpha^0 = 1$ es 1, y, en efecto,

$$L(1) = \alpha^3 + 1 + \alpha^3 + 1 = 0.$$

Asimismo, el recíproco del elemento α^{14} es $\alpha^{-14} = \alpha$, y es inmediato verificar que

$$L(\alpha) = (\alpha^3 + 1)\alpha^2 + \alpha^3\alpha + 1 = 0$$

pues $\alpha^5 = \alpha^2 + \alpha$ y $\alpha^4 = \alpha + 1$.

La figura 10.4 es el esquema del circuito de búsqueda de Chien para este código. El primer registro de desplazamiento se carga inicialmente con la representación binaria de L_1, y el segundo, con la de L_2; cada iteración de los registros sirve para calcular $L_1\alpha^i + L_2\alpha^{2i}$, y cuando se detecta $L_1\alpha^i + L_2\alpha^{2i} = 1$, se procede a corregir el error. ■

En resumen, la decodificación algebraica de códigos BCH se ha dividido en tres etapas: *a*) obtención del síndrome; *b*) cálculo de los coeficientes del polinomio localizador, bien con el algoritmo de Peterson o bien con el de Berlekamp (se precisa del polinomio localizador para convertir un problema no lineal en uno lineal); *c*) búsqueda exhaustiva de las raíces del polinomio localizador de errores.

Notas bibliográficas

La investigación de métodos algebraicos de decodificación de códigos polinómicos, y muy especialmente de códigos BCH, ocupó la primera mitad de la década de los sesenta [10]. La solución de Peterson al problema de resolver un sistema lineal con los elementos localizadores como incógnitas es muy similar a la dada por de Prony en 1795 al mismo problema en un cuerpo infinito. Las cotas de distancia BCH, de Hartmann–Tzeng y de Roos se estudian con detalle en [68]. En general, no se conocen aún ni la distancia ni la distribución de pesos ni el grupo de automorfismos de los códigos BCH, excepto en casos limitados como el de los códigos BCH en sentido restringido.

Las referencias más autorizadas sobre el algoritmo de Berlekamp–Massey continúan siendo [3] y [43], aunque una revisión más actualizada se puede encontrar en [53, cap. 19].

CAPÍTULO 11

Códigos Reed–Solomon

Los códigos Reed–Solomon forman una clase preferente dentro de los códigos BCH no binarios, dado que tienen la máxima distancia que puede alcanzar cualquier código lineal de su misma longitud y dimensión. Han adquirido una gran relevancia en las aplicaciones no sólo por su capacidad para corregir largas ráfagas de errores y de símbolos borrados, sino también porque se conoce para ellos un algoritmo de decodificación computacionalmente eficiente, el algoritmo de Berlekamp–Massey. Además, resultan muy convenientes para construir otros códigos. Por ejemplo, se pueden transformar directamente en códigos binarios de distancia elevada, se suelen utilizar como códigos exteriores en los sistemas de códigos concatenados y fueron en su momento la base para desarrollar la teoría de códigos algebraico–geométricos, de la que constituyen uno de los ejemplos más simples.

Hoy, los sistemas de transmisión o de almacenamiento de la información que utilizan alguno de los códigos Reed–Solomon son cada vez más numerosos, y abarcan desde las comunicaciones espaciales o la transmisión digital terrena (por ejemplo, la televisión digital de alta definición) hasta productos de electrónica de consumo como los grabadores y reproductores de discos compactos de audio y vídeo (DVD).

Este capítulo está dedicado a describir la construcción (cf. apartados 11.1 y 11.2) y los algoritmos de decodificación (apartados 11.3 a 11.5) de la familia de códigos Reed–Solomon. El apartado 11.7 presenta los códigos Reed–Solomon generalizados. El apéndice 11.A establece el fuerte vínculo existente entre cierta clase de códigos polinómicos y la transformada de Fourier en un cuerpo finito. Los fundamentos algebraicos de la técnica de decodificación por localización y del algoritmo de Berlekamp–Massey son la materia del apéndice 11.B.

11.1. Definición

Existen varias maneras alternativas de definir la familia de códigos Reed–Solomon, pero tal vez la más directa sea considerarlos en relación con los códigos BCH.

DEFINICIÓN 11.1 (CÓDIGO RS). *Un código Reed–Solomon (RS) sobre el cuerpo $\mathbb{F}_q$ es un código BCH de longitud $q-1$.*

Por tanto, la longitud de un código RS es el número de elementos no nulos del cuerpo base; y, naturalmente, $q \neq 2$, es decir, los códigos RS no son binarios.

En los términos de la definición 11.1, el procedimiento de construcción de un código RS es sencillo. Una raíz primitiva $(q-1)$-ésima de la unidad es cualquier elemento primitivo, α, del cuerpo $\mathbb{F}_q$. El polinomio mínimo de α^i sobre $F_q[x]$ es $x - \alpha^i$. Así, el polinomio generador de un código RS de distancia (de diseño) $2t+1$ es

$$g(x) = (x-\alpha^b)(x-\alpha^{b+1})\cdots(x-\alpha^{b+2t-1}).$$

Normalmente se elige $b=0$ o $b=1$.

Se desprenden de inmediato de la definición algunas proposiciones sencillas:

a) El dual de un código RS-$[n,k]$ es otro código RS-$[n,n-k]$; en efecto, así ocurre, ya que el conjunto de elementos no nulos de $\mathbb{F}_q$ es un grupo multiplicativo cíclico. Si $\mathcal{Z}$ es el conjunto de ceros del código RS-$[n,k]$, los ceros del dual también son consecutivos

$$\mathcal{Z}^{\perp} = \{\alpha^j : \alpha^{n-j} \notin \mathcal{Z}\}.$$

Obsérvese que esta propiedad no es cierta en el caso de un código BCH general: el dual de un código BCH no siempre es otro código BCH.

Además, si $\mathcal{C}$ es un código RS, uno de los códigos $\mathcal{C}$ o $\mathcal{C}^{\perp}$ es *par* y el otro es *impar*, porque sólo uno de ambos tiene a 1 entre sus raíces. Un código q-ario es par si para cualquier palabra del código $x_1x_2\ldots x_n$ es $\sum_{i=1}^{n} x_i = 0$, e impar si no se cumple tal condición.[1]

[1] Luego un código binario es par si está compuesto sólo por vectores de peso Hamming par.

b) Una matriz de comprobación de paridad de un código RS-$[n,k]$ es

$$H = \begin{pmatrix} \alpha^{n-1} & \dots & \alpha^2 & \alpha & 1 \\ \alpha^{2(n-1)} & \dots & \alpha^4 & \alpha^2 & 1 \\ \dots & \dots & \dots & \dots & \dots \\ \alpha^{(n-k)(n-1)} & \dots & \alpha^{2(n-k)} & \alpha^{n-k} & 1 \end{pmatrix} = \begin{pmatrix} \alpha_{n-1} & \dots & \alpha_2 & \alpha_1 & 1 \\ \alpha_{n-1}^2 & \dots & \alpha_2^2 & \alpha_1^2 & 1 \\ \dots & \dots & \dots & \dots & \dots \\ \alpha_{n-1}^{n-k} & \dots & \alpha_2^{n-k} & \alpha_1^{n-k} & 1 \end{pmatrix}$$

en donde $1, \alpha_1, \dots, \alpha_{n-1}$ son los elementos no nulos de $\mathbb{F}_q$.

c) Una matriz generadora de RS-$[n,k]$ es

$$G = \begin{pmatrix} \alpha^{(n-1)(k-1)} & \dots & \alpha^{2(k-1)} & \alpha^{k-1} & 1 \\ \alpha^{(n-1)(k-2)} & \dots & \alpha^{2(k-2)} & \alpha^{k-2} & 1 \\ \dots & \dots & \dots & \dots & \dots \\ \alpha^{n-1} & \dots & \alpha^2 & \alpha & 1 \\ 1 & \dots & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{\alpha}_{k-1} \\ \boldsymbol{\alpha}_{k-2} \\ \vdots \\ \boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_0 \end{pmatrix}$$

en donde $\boldsymbol{\alpha}_i = (\alpha_i^{n-1} \dots \alpha_i^1 \alpha_i^0)$ y $\alpha_i = \alpha^i$ para $0 \le i < k$. Es sencillo verificar que el producto escalar entre la fila $k-j$ de G y la fila h de H vale

$$\sum_{i=0}^{n-1} \alpha^{ij} \alpha^{ih} = \sum_{i=0}^{n-1} \alpha^{i(j+h)} = \frac{1-\alpha^{n(j+h)}}{1-\alpha^{j+h}} = 0$$

ya que, por definición, $\alpha^n = 1$ y $1 \le j+h < n$.

Entonces, si $(c_{k-1}, c_{k-2}, \dots, c_0)$ es cualquier secuencia de k símbolos de $\mathbb{F}_q$, un vector del código se puede calcular como

$$\mathbf{v} = (c_{k-1}, c_{k-2}, \dots, c_0) \cdot G.$$

Pero, atendiendo a la estructura de G, este producto se puede escribir también como

$$\begin{aligned} \mathbf{v} &= \left(\sum_{i=0}^{k-1} c_i \alpha^{i(n-1)}, \sum_{i=0}^{k-1} c_i \alpha^{i(n-2)}, \dots, \sum_{i=0}^{k-1} c_i \alpha^i, \sum_{i=0}^{k-1} c_i \right) \\ &= \big(P(\alpha^{n-1}), P(\alpha^{n-2}), \dots, P(\alpha), P(1) \big) \end{aligned}$$

siendo $P(x)$ el polinomio $c_{k-1}x^{k-1} + \dots + c_1 x + c_0$. En consecuencia, un código RS es el conjunto de puntos

$$\mathbf{v} = \big(P(\alpha^{n-1}), P(\alpha^{n-2}), \dots, P(\alpha), P(1) \big)$$

con $P(x)$ un polinomio arbitrario en $\mathbb{F}_q$ de grado inferior a k y α una raíz primitiva n-ésima de la unidad. Ésta es una definición muy similar a la original, elegante y simple, de Reed y Solomon.[2] Como para la mayoría de las cuestiones relativas a la decodificación es más cómoda la caracterización algebraica en términos de los ceros del polinomio generador, en el resto del capítulo se utilizará preferiblemente la definición 11.1.

d) El código RS-$[2^m-1, k]$ contiene al BCH binario primitivo de longitud 2^m-1 y distancia de diseño 2^m-k: en efecto, los elementos α, α^2, ..., α^{2^m-k-1}, que definen al polinomio generador del código RS, también son ceros del polinomio generador del código BCH.

La propiedad más importante de un código RS es que su distancia es igual al número de símbolos de redundancia más uno, la mayor posible. En otras palabras, la distancia de diseño es la distancia del código y es máxima.

TEOREMA 11.1 (DISTANCIA DE UN CÓDIGO RS). *Un código RS-$[n, k]$ es separable y de máxima distancia (MDS), es decir, tiene distancia $n-k+1$.*

DEMOSTRACIÓN. La cota de Singleton (teorema 6.17, página 181) establece que la distancia de cualquier código está acotada superiormente por

$$d_{\mathcal{C}} \leq n - k + 1$$

mientras que la cota de distancia BCH (teorema 10.1) establece que la distancia del código está acotada inferiormente por la distancia de diseño, que, en el caso de un código RS, es igual al grado $n-k$ del polinomio generador más uno,

$$d_{\mathcal{C}} \geq n - k + 1.$$

Por lo tanto, $d_{\mathcal{C}} = n - k + 1$, y los códigos RS son óptimos en cuanto a distancia en $\mathbb{F}_q$, ya que alcanzan la cota de Singleton. ▶

En realidad, la mayoría de las propiedades de interés de los códigos RS son una consecuencia de la propiedad MDS. Veamos algunas de ellas.

[2] I. Reed y G. Solomon estudiaron los códigos formados por los conjuntos de puntos

$$\big(P(\alpha^{n-1}), P(\alpha^{n-2}), \ldots, P(\alpha), P(1), P(0)\big)$$

para $P(x)$ un polinomio cualquiera de grado inferior a k en $\mathbb{F}_q$. Tales códigos son lineales pero no cíclicos. Generalicemos esta situación como sigue. Sea $\mathcal{X}$ un objeto geométrico cualquiera que contiene a una familia finita de puntos $\mathcal{P} = \{p_1, \ldots, p_n\}$ (por ejemplo, la recta afín $\mathcal{X} = \mathbb{F}_q$ y los puntos $\mathcal{P} = \mathbb{F}_q^*$). Sea L un espacio lineal de funciones de $\mathcal{P}$ en $\mathbb{F}_q$, como por ejemplo $L = F_q[x]$. El conjunto $\mathcal{C} = \{(\Lambda(p_1), \Lambda(p_2), \ldots, \Lambda(p_n)), \Lambda \in L\}$ es un código lineal. Esta es precisamente la forma en que se definen los códigos algebraico–geométricos. Los casos más interesantes (y menos triviales) se obtienen al tomar una curva o una superficie plana para $\mathcal{X}$.

Teorema 11.2. *El dual de un código separable de máxima distancia es otro código separable de máxima distancia.*

Demostración. Si un código $\mathcal{C}[n,k]$ es separable, de máxima distancia, cualquier conjunto de $n-k$ columnas de su matriz de comprobación de paridad H es linealmente independiente. Pero, como H engendra el código dual, esto significa que el único vector de $\mathcal{C}^{\perp}[n,n-k]$ con $n-k$ o más ceros es el vector nulo. Porque si existiese un vector del código dual con $n-k$ componentes $\{c_1,\ldots,c_{n-k}\}$ a cero, entonces, dado que es una combinación lineal de las filas de H, la submatriz compuesta por las columnas $c_1,c_2,\ldots,c_{n-k}$ de H sería singular y su rango no podría ser $n-k$. En consecuencia, $d_{\mathcal{C}^{\perp}} \geq k+1$; y como, por la inecuación de Singleton, $d_{\mathcal{C}^{\perp}} \leq n-(n-k)+1 = k+1$, se concluye que $d_{\mathcal{C}^{\perp}} = k+1$. ▶

Teorema 11.3. *En un código MDS, cualquier conjunto de k coordenadas de una palabra del código se pueden tomar como símbolos de información.*

Demostración. Por el mismo argumento que el teorema anterior, cualquier conjunto de k columnas de una matriz generadora del código es linealmente independiente. ▶

El término *separable* en la denominación MDS alude precisamente a esta propiedad: sin más que elegir k posiciones arbitrarias (es decir, sin necesidad de realizar ninguna transformación lineal), los vectores del código quedan sistematizados, divididos en una parte de símbolos de información y otra parte compuesta por los símbolos de redundancia.

El corolario que sigue es, entonces, evidente.

Corolario 11.4 (Representación sistemática de un código RS). *En un código RS-$[n,k]$, cualquier conjunto de k símbolos de las palabras del código se pueden tomar como símbolos de información.*

Además de la sencillez con que se puede disponer una representación sistemática, se da el caso excepcional de que la distribución de pesos de un código MDS es conocida; la de un código RS, entonces, también.

Teorema 11.5 (Distribución de pesos de un código MDS). *En un código MDS $[n,k]$ sobre $\mathbb{F}_q$ existen*

$$A_j = \binom{n}{j}(q-1)\sum_{i=0}^{j-d_{\mathcal{C}}}(-1)^i\binom{j-1}{i}q^{j-i-d_{\mathcal{C}}}, \quad j = d_{\mathcal{C}},\ldots,n \tag{11.1}$$

vectores de peso Hamming j, siendo $d_{\mathcal{C}} = n-k+1$ la distancia del código.

DEMOSTRACIÓN. Para códigos q-arios, la identidad de MacWilliams se expresa con la fórmula

$$W_{\mathcal{C}^\perp}(x,y) = \frac{1}{|\mathcal{C}|} W_{\mathcal{C}}(x+(q-1)y, x-y)$$

en la que $W_{\mathcal{C}}(x,y) = \sum_{i=0}^{n} A_i x^{n-i} y^i$ es el polinomio enumerador de pesos de $\mathcal{C}$, y $W_{\mathcal{C}^\perp}(x,y) = \sum_{i=0}^{n} B_i x^{n-i} y^i$ es el polinomio enumerador de pesos del dual. Sustituyendo en ambos miembros de la identidad de MacWilliams x por $y+z$, expandiendo los términos e igualando después los coeficientes, resultan las ecuaciones lineales

$$\sum_{j=0}^{n-i} \binom{n-j}{i} A_j = q^{k-i} \sum_{j=0}^{i} \binom{n-j}{n-i} B_j, \quad 0 \le i \le n, \tag{11.2}$$

que relacionan explícitamente las distribuciones de pesos de un código y su dual.

Ahora, sea $\mathcal{C}$ un código MDS de longitud n y distancia d. Obviamente $A_0 = 1$, $A_i = 0$ para $1 \le i < d$. Como $\mathcal{C}^\perp$ también es MDS, $B_0 = 1$ y $B_i = 0$ para $1 \le i \le k$. Por tanto, según (11.2),

$$\sum_{j=d}^{n-i} \binom{n-j}{i} A_j = \binom{n}{i}(q^{n-d+1-i} - 1) = \binom{n}{i}(q^{k-i} - 1), \quad 0 \le i \le n-d. \tag{11.3}$$

Veremos que este sistema de ecuaciones recurrentes posee una única solución, que es precisamente la dada por el enunciado del teorema.

En el sistema (11.3), para el índice $i = k-1$ tenemos la ecuación

$$A_{n-k+1} = \binom{n}{k-1}(q-1) = \binom{n}{n-k+1}(q-1)$$

que se ajusta a la fórmula general (11.1). Procedamos ahora por inducción, suponiendo que (11.1) se cumple para $j = n-k+1, \ldots, h-1$. Para $i = h$, de la ecuación (11.3) se tiene que

$$A_{n-h} = \binom{n}{h}(q^{k-h} - 1) - \sum_{j=d}^{n-h-1} \binom{n-j}{h} A_j.$$

Introduciendo en el sumatorio del segundo miembro la hipótesis de inducción

$$\begin{aligned} A_{n-h} = {} & \binom{n}{h}(q^{k-h} - 1) \\ & - \sum_{j=d}^{n-h-1} \binom{n-j}{h} \binom{n}{j} (q-1) \sum_{l=0}^{j-d} (-1)^l \binom{j-1}{l} q^{j-l-d}. \end{aligned}$$

Ahora bien, como

$$\binom{n-j}{h}\binom{n}{j} = \binom{n}{h}\binom{n-h}{j}$$

podremos escribir igualmente

$$\begin{aligned} A_{n-h} &= \binom{n}{h}(q^{k-h}-1) \\ &\quad - \binom{n}{h}(q-1)\sum_{j=d}^{n-h-1}\binom{n-h}{j}\sum_{l=0}^{j-d}(-1)^l\binom{j-1}{l}q^{j-l-d}. \end{aligned}$$

Efectuando un cambio de variable y permutando el orden de los sumatorios, la ecuación anterior queda en la forma

$$\begin{aligned} A_{n-h} &= \binom{n}{h}(q^{k-h}-1) \\ &\quad - \binom{n}{h}(q-1)\sum_{m=0}^{n-h-d-1} q^m \sum_{\nu=m+d}^{n-h-1}(-1)^{\nu-m-d}\binom{n-h}{\nu}\binom{j-1}{\nu-m-d}. \end{aligned} \tag{11.4}$$

Considérense ahora los términos definidos por

$$J_{a,b} = \sum_{i=a}^{b}\binom{b}{i}\binom{i-1}{i-a}(-1)^{i-a}$$

para a, b enteros, y la función real $f(x) = x^{-1}(1+x)^b$, que satisface

$$f(-1) = 0, \qquad \frac{\mathrm{d}^j}{\mathrm{d}x^j}f(x)\bigg|_{x=-1} = 0 \qquad \text{para } 1 \le j < b.$$

Si expandimos los términos de $f(x)$, su derivada de orden $p < b$ valdrá

$$\begin{aligned} \frac{\mathrm{d}^p}{\mathrm{d}x^p}f(x)\bigg|_{x=-1} &= \frac{\mathrm{d}^p}{\mathrm{d}x^p}\left(x^{-1} + \sum_{j=1}^{b}\binom{b}{j}x^{j-1}\right)\bigg|_{x=-1} \\ &= \frac{\mathrm{d}^p}{\mathrm{d}x^p}x^{-1}\bigg|_{x=-1} + \sum_{l=p+1}^{b}\binom{b}{l}(l-1)(l-2)\cdots(l-p)x^{l-p-1}\bigg|_{x=-1} \\ &= -p! + p!\, J_{p+1,b}. \end{aligned}$$

De lo que se concluye que $J_{p,b} = 1$ para todo $0 \le p \le b$. En este punto, en

la fórmula (11.4) introduzcamos la sustitución

$$\begin{aligned}&\sum_{\nu=m+d}^{n-h-1}(-1)^{\nu-m-d}\binom{n-h}{\nu}\binom{j-1}{\nu-m-d}\\&= J_{m+d,n+h}-(-1)^{n-h-1-(m+d)}\binom{n-h-1}{n-h-(m+d)}\\&= 1-(-1)^{n-h-1-(m+d)}\binom{n-h-1}{n-h-(m+d)}.\end{aligned}$$

Tendremos así

$$\begin{aligned}A_{n-h} &= \binom{n}{h}(q^{k-h}-1)-\binom{n}{h}(q-1)\sum_{m=0}^{n-h-d-1}q^m\\&\quad+\binom{n}{h}(q-1)\sum_{m=0}^{n-h-d-1}(-1)^{n-h-(m+d)}\binom{n-h-1}{n-h-(m+d)}q^m.\end{aligned}$$

Pero la suma del primer y el segundo sumandos vale

$$\binom{n}{h}(q-1)q^{n-h-d}$$

y por tanto

$$A_{n-h}=\binom{n}{h}(q-1)\sum_{m=0}^{n-h-d}(-1)^{n-h-(m+d)}\binom{n-h-1}{n-h-(m+d)}q^m.$$

Haciendo en esta última expresión el cambio de variable $i = n-h-(m+d)$, obtenemos la ecuación (11.1) en el caso particular $j = n-h$, lo que demuestra que la hipótesis de inducción se satisface para este caso. ▶

A partir de la fórmula explícita de la distribución de pesos se deducen con facilidad algunas cotas para la longitud y la dimensión de un código MDS.

TEOREMA 11.6. *Si existe $\mathcal{C}$, un código MDS $[n,k,d_{\mathcal{C}}]$ sobre $\mathbb{F}_q$:*

a) Si $k \geq 2$, entonces $d_{\mathcal{C}} = n-k+1 \leq q$.

b) Si $k \leq n-2$, entonces $k+1 \leq q$.

DEMOSTRACIÓN.

a) Si $d_{\mathcal{C}} < n$, por el teorema 11.5 se tiene

$$A_{d_{\mathcal{C}}+1} = \binom{n}{d_{\mathcal{C}}+1}(q-1)(q-d_{\mathcal{C}}).$$

Como $A_{d_{\mathcal{C}}+1} \geq 0$, se cumple $d_{\mathcal{C}} \leq q$.

b) Como $\mathcal{C}^{\perp}$ es un código MDS $[n, n-k, k+1]$, se cumple, por el apartado *a*), que $k+1 \leq q$ si $n-k \geq 2$, que es precisamente la proposición *b*). ▶

Cuando $k = 1$ o $k = n-1$, existen códigos MDS de longitud arbitraria, que son equivalentes a un código de repetición sobre $\mathbb{F}_q$ o bien a uno de sus duales. Todos ellos son códigos triviales. Pero si $k \leq n-2$, el teorema anterior implica que $k \leq q-1$ (parte *b*)), de modo que sólo podrán existir códigos MDS no triviales con dimensión $2 \leq k \leq \text{mín}\{n-2, q-1\}$. La parte *a*) implica que $n \leq q+k-1$, y significa que no existen códigos MDS no triviales con longitud arbitraria. Para los códigos RS (que poseen parámetros $[q-1, k, q-k]$), aquéllos que son no triviales satisfacen obviamente las condiciones $k \leq q-1$ (parte *b*) del teorema 11.6) y $2 \leq k \leq \text{mín}\{n-2, q-1\} = q-3$ (parte *a*) de 11.6).

Ejemplo 11.1. Para conseguir un código RS de longitud 7 capaz de corregir errores dobles, sea α una raíz primitiva séptima de la unidad, $\alpha^7 = 1$. El polinomio generador del código habrá de tener 4 potencias consecutivas de α como ceros, por ejemplo

$$g(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4) = x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3.$$

El grado de $g(x)$ es 4, y el código RS que engendra es uno $[7, 3]$ sobre $\mathbb{F}_8$; consta, por tanto, de $8^3 = 512$ vectores. Y tiene por matriz de comprobación de paridad a

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

Ejemplo 11.2. Sea α una raíz del polinomio primitivo binario x^2+x+1 y, por tanto, una raíz primitiva tercera de la unidad ($\alpha^3 = 1$). El código RS de longitud 3 sobre el cuerpo $\mathbb{F}_4$, con distancia de diseño 2, que tiene por polinomio generador a $g(x) = x - \alpha$, consta de las 16 palabras siguientes:

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

Dicho código es irreducible e isomorfo a $\mathbb{F}_{16}$ —este isomorfismo es $a(x)(x-\alpha) \to a(x) \mod (x^2+\alpha x+1)$; el elemento primitivo es $\alpha x \mod (x^2+\alpha x+1)$—; y su distribución de pesos es $A_0 = 1$, $A_2 = 9$, $A_3 = 6$ y $A_i = 0$ para cualquier otro i, tal y como anticipa el teorema 11.5. ■

11.2. Códigos RS modificados

En este apartado veremos que los procedimientos de extensión, punción o recorte de un código RS no producen, en general, códigos RS (ni tan siquiera, salvo en algunos casos aislados, códigos cíclicos), pero curiosamente sí producen códigos separables de máxima distancia. En este sentido, continúan siendo óptimos.

Desde un punto de vista práctico, posee mayor interés la transformación de un código RS en un código binario; se va a ver que esta conversión también suele dar códigos de distancia grande, aunque asintóticamente ineficientes.

11.2.1. Extensión

Se llama extensión de un código $\mathcal{C}$ a la operación que consiste en añadir a sus palabras $(c_{n-1}, \ldots, c_0)$ el símbolo

$$c_n = -\sum_{i=0}^{n-1} c_i.$$

El código extendido $\widehat{\mathcal{C}}$ tiene siempre longitud una unidad mayor y la misma dimensión que el código original pero, en general, su distancia no tiene por qué incrementarse. Pues bien, en los códigos RS la extensión sí aumenta la distancia.

TEOREMA 11.7. *La extensión de un código RS-*$[n,k]$ *produce un código MDS* $[n+1,k]$.

DEMOSTRACIÓN. Si $(c_{n-1} \ldots c_0)$ es un vector de RS-$[n,k]$ de peso $n-k+1$, se va a probar que el símbolo de paridad

$$c_n = -\sum_{j=0}^{n-1} c_j \neq 0$$

y que, por tanto, la distancia del código extendido se incrementa en una unidad.

Así, si $c(x)$ simboliza el polinomio del código, y $c(1) = -c_n = 0$, entonces, puesto que $c(x) = a(x)g(x)$ y

$$g(1) = (x-\alpha)\cdots(x-\alpha^{n-k})\big|_{x=1} \neq 0$$

debe ser $a(1) = 0$. Pero, en este caso, $c(x)$ sería un múltiplo de $(x-1)g(x)$, y, por la cota de distancia BCH, el peso de $c(x)$ sería mayor o igual que $n-k+2$. ▶

Los códigos RS extendidos pertenecen a la clase de los *códigos RS generalizados*. Éstos, a su vez, pueden ser extendidos añadiéndoles un nuevo símbolo, elegido de modo que la distancia aumente en una unidad. En general, el nuevo código no será par.

Teorema 11.8. *La extensión de un código RS-*$[n,k]$ *extendido produce un código MDS* $[n+2,k]$.

Demostración. La matriz de comprobación de paridad de un código RS doblemente extendido se define por

$$H = \begin{pmatrix} \alpha_1^{q-k} & \dots & \alpha_{q-1}^{q-k} & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots \\ \alpha_1^2 & \dots & \alpha_{q-1}^2 & 0 & 0 \\ \alpha_1 & \dots & \alpha_{q-1} & 0 & 0 \\ 1 & \dots & 1 & 1 & 0 \end{pmatrix}$$

en donde $\alpha_1, \dots, \alpha_{q-1}$ son los elementos no nulos de $\mathbb{F}_q$, $n = q-1$ y $\delta = n-k+1$ es la distancia de diseño.

Para mostrar que corresponde a un código MDS es suficiente con probar que cualesquiera $q-k+1$ columnas suyas son linealmente independientes. Pero si estas $q-k+1$ columnas se eligen de entre las $q-1$ primeras, se tiene una matriz de Vandermonde no singular; y si entre las $q-k+1$ columnas están incluidas alguna de las dos últimas, entonces basta con desarrollar el determinante por ellas para obtener una matriz de Vandermonde asimismo no singular. ▶

Se sabe que existen códigos MDS cíclicos sobre $\mathbb{F}_q$ con los mismos parámetros que un código RS doblemente extendido. Y que existen códigos MDS $[2^m+2, 3]$ y $[2^m+2, 2^m-1]$ que se obtienen por una extensión triple de un RS.

Ejemplo 11.3. La extensión simple del código RS dado en el ejemplo 11.2 produce el código

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

Vea que la distancia es 3 y que el código no es cíclico. ■

11.2.2. Punción

La punción o perforación es la operación de eliminar una o más coordenadas de todas las palabras del código.

Teorema 11.9. *Eliminando cualquier coordenada de un código RS-*$[n, k]$*, se obtiene un código MDS-*$[n-1, k]$.

Demostración. Puesto que cualquier conjunto de k símbolos de un código RS-$[n, k]$ se pueden tomar como símbolos de información, es claro que al eliminar una coordenada cualquiera se obtiene un código con longitud una unidad menor e idéntica dimensión. La distancia disminuye como máximo en una unidad y será, entonces, no inferior a $n-k$; pero, por la desigualdad de Singleton, tampoco puede ser mayor que $n-k$. En consecuencia, la supresión de un símbolo produce un código MDS. ▶

Ejemplo 11.4. Eliminando la primera coordenada del código dado en el ejemplo 11.2 resulta el siguiente código cuaternario $[2, 2, 1]$ trivial

$$\begin{array}{cccccccc} 00 & \alpha 0 & \alpha^2 0 & 0\alpha & 1\alpha & 0\alpha^2 & 1\alpha^2 & \alpha 1 \\ \alpha\alpha^2 & 11 & \alpha\alpha & 10 & \alpha^2 1 & \alpha^2\alpha & 01 & \alpha^2\alpha^2 \end{array}$$

que es sencillamente $\mathbb{F}_4 \times \mathbb{F}_4$. ■

11.2.3. Códigos acortados

Teorema 11.10. *Acortar un código RS-*$[n, k]$ *produce un código MDS-*$[n-1, k-1]$.

Demostración. Una vez más, dado que cualquier conjunto de k coordenadas de un código RS son linealmente independientes, al seleccionar todos los vectores del código con una coordenada a cero y eliminarla se obtiene un código con dimensión $k-1$, longitud $n-1$ y distancia $n-k+1$, es decir, un código MDS. ▶

Ejemplo 11.5. El código $\{00, 1\alpha, \alpha\alpha^2, \alpha^2 1\}$ resulta de recortar, del dado en el ejemplo 11.2, las palabras con la primera coordenada a cero. Es un código lineal cuaternario $[2, 1]$, con distancia 2, no cíclico. ■

11.2.4. Conversión de códigos RS en códigos binarios

En el capítulo 9 se vio que el cuerpo $\mathbb{F}_{p^m}$ es isomorfo a un espacio vectorial de dimensión m sobre $\mathbb{F}_p$. Esta propiedad da un procedimiento inmediato para convertir un código RS-$[2^m - 1, k]$ en un código binario.

Pues si $B = \{\eta_1, \eta_2, \ldots, \eta_m\}$ es una base de $\mathbb{F}_{2^m}$ sobre $\mathbb{F}_2$ (es decir, cualquier elemento de $\mathbb{F}_{2^m}$ es una combinación lineal de $\eta_1, \ldots, \eta_m$ utilizando sólo 0 y 1 como escalares), entonces la sustitución que a $\gamma \in \mathbb{F}_{2^m}$ le hace corresponder las coordenadas de γ en la base B transforma el código RS original en un código binario de parámetros $[m(2^m - 1), km]$ y distancia mayor o igual que $2^m - k$. Puesto que el código RS de partida es separable y de máxima distancia, típicamente esta cota inferior acostumbra a ser conservadora, y la distancia del código binario así construido es bastante más elevada.

Pero se puede incrementar aún más si se añade a cada conjunto de m símbolos binarios un bit de paridad. Este nuevo código binario tiene parámetros $[(m+1)(2^m-1), km]$ y distancia no inferior a $2(2^m-k)$. Se puede aplicar el mismo procedimiento de construcción a un código RS extendido para obtener un código binario $[(m + 1)2^m, km]$ y distancia no inferior a $2(2^m - k + 1)$.

Ejemplo 11.6.

a) $\{1, \alpha\}$ es una base de $\mathbb{F}_4$. En ella, 0 admite la representación 00; 1, la representación 10; α, el par de coordenadas 01; y α^2, el par 11. Así pues, el código RS del ejemplo 11.2 da, con estas sustituciones, el código binario $[6, 4]$

000000	100100	110001	110110
001001	011100	100011	101010
000111	111000	101101	010101
001110	010010	011011	111111

de distancia 2. Si a cada grupo de tres bits le añadimos un bit de paridad par, tendremos un código autodual binario $[8, 4]$ con distancia 4 equivalente al dual de $\mathcal{H}_3$ extendido.

b) A partir del código RS-$[15, 10]$ y el código RS extendido $[16, 10]$ se consiguen, respectivamente, dos códigos binarios excelentes: un $[75, 40]$

de distancia 12 y un $[80, 40]$ de distancia 14. Ambos se sitúan por encima de la cota de Gilbert–Varshamov.

c) El código RS-$[7, 3]$ de polinomio generador $g_1(x) = x^2 + \alpha x + \alpha^4$ se transforma, eligiendo la base binaria $\{1, \alpha, \alpha^6\}$ en $\mathbb{F}_8$, en el código cíclico BCH binario $[21, 15]$ engendrado por $g_2(x) = x^6+x^4+x^2+x+1$; se trata de uno de los escasos ejemplos conocidos de un código RS que origina otro código cíclico.

d) Sea $\mathcal{C}$ el código RS definido sobre el alfabeto $\mathbb{F}_{2^n}$ de parámetros $[2^n-1, 1, 2^n-1]$, generado por el vector código $(1, \alpha, \ldots, \alpha^{2^n-2})$ siendo α un elemento primitivo de $\mathbb{F}_{2^n}$. Considérese el código $\mathcal{B}$ que resulta de sustituir cada uno de los símbolos de una palabra de $\mathcal{C}$ por su representación binaria en una base de $\mathbb{F}_{2^n}$ sobre $\mathbb{F}_2$. $\mathcal{B}$ es un código binario lineal $[n(2^n - 1), n]$ y, puesto que para todo vector de $\mathcal{C}$ cualquier símbolo no nulo de $\mathbb{F}_{2^n}$ aparece en él una sola vez, su distancia es $n2^{n-1}$. Por tanto $\mathcal{B}$ alcanza la cota de Griesmer. ■

A pesar de lo que sugiere este ejemplo, se puede probar que, si se fija una tasa k/n constante, la distancia d_m de los códigos binarios $[(m + 1)(2^m - 1), km]$ así construidos no es una fracción constante de la longitud sino que, en realidad,

$$\lim_{m\to\infty} \frac{d_m}{(m + 1)(2^m - 1)} = 0,$$

de forma que, para longitudes grandes, no son buenos códigos.

Los códigos binarios obtenidos a partir de los RS sí son, en todo caso, especialmente convenientes para la detección y corrección de ráfagas de errores de larga longitud. Pues, por ser los RS cíclicos, podrán detectar todas las ráfagas de errores de longitud menor o igual que $n-k$ símbolos de $\mathbb{F}_{2^m}$. Pero una ráfaga de error de r bits afecta, como máximo, a s símbolos de $\mathbb{F}_{2^m}$, donde s se define por

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

De lo que se desprende que, si $n-k$ es mucho mayor que s (o lo que es igual, si m es grande), se podrán corregir gran cantidad de ráfagas de longitud r bits. Los datos almacenados en un disco compacto de audio, por ejemplo, se protegen codificando cada bloque de 16 bits con dos códigos RS (acortados, de longitud 255) concatenados (esto es, la salida del primer codificador se aplica a la entrada del segundo). La ráfaga de errores corregible de mayor longitud tiene alrededor de 4000 bits de datos, aproximadamente 2,5 mm de longitud en la superficie del disco.

11.3. Decodificación de códigos RS

Cualquier conjunto de k símbolos recibidos sin error en un vector de un código RS son suficientes para recuperar el mensaje transmitido. En efecto, si se ha recibido la secuencia $\mathbf{y} = (y_{n-1}, \ldots, y_0)$, cualquier grupo de k ecuaciones del sistema

$$\begin{aligned} y_{n-1} &= x_0 + x_1\alpha^{k-1} + \cdots + x_{k-1}\alpha^{(n-1)(k-1)} \\ &\vdots \\ y_1 &= x_0 + x_1\alpha + \cdots + x_{k-1}\alpha^{k-1} \\ y_0 &= x_0 + x_1 + \cdots + x_{k-1}, \end{aligned}$$

digamos las de los símbolos $y_{i_0}, y_{i_1}, \ldots, y_{i_{k-1}}$, admiten solución única, puesto que su matriz de coeficientes es de Vandermonde y no singular. Existen $\binom{n}{k}$ subsistemas diferentes de k ecuaciones y, en ausencia de errores de transmisión, todos proporcionan como solución los símbolos $(m_{k-1}, \ldots, m_0)$ del mensaje transmitido.

Supongamos, sin embargo, que se hubiesen producido $t \leq \lfloor (n-k)/2 \rfloor$ errores de transmisión. En tal caso, el vector $(m_{k-1}, \ldots, m_0)$ será la solución de cualquier conjunto de k ecuaciones tomadas de entre las $n-t$ coordenadas correctas de $\mathbf{y}$, esto es, de $\binom{n-t}{k}$ subsistemas; en tanto que cualquier otro vector distinto de $(m_{k-1}, \ldots, m_0)$ sólo podrá ser solución, a lo sumo, de aquellos subsistemas en los que al menos una de las ecuaciones corresponde a alguna de las t coordenadas con error, esto es, de $\binom{t+k-1}{k}$ subsistemas. Si por cada una de las soluciones $\mathbf{x}$ de los subsistemas $k \times k$ se anota un voto favorable al vector $\mathbf{x}$, entonces el razonamiento que se acaba de exponer permite afirmar que $(m_{k-1}, \ldots, m_0)$ recibe $\binom{n-t}{k}$ votos, mientras que cualquier otro vector distinto obtiene, en el mejor de los casos, $\binom{t+k-1}{k}$ votos. Y como $n - t \geq t + k - 1$, resulta obvio que el mensaje transmitido $(m_{k-1}, \ldots, m_0)$ recibe siempre más votos que ningún otro vector candidato. Pero, si bien una estrategia de decodificación por mayoría es perfectamente válida para corregir cualquier patrón de $\lfloor (n-k)/2 \rfloor$ o menos errores con un código RS, este método de decodificación presenta un interés meramente teórico ya que exige la resolución de $\binom{n}{k}$ sistemas lineales en el cuerpo base.

11.4. Decodificación por localización de códigos RS y BCH no binarios

En este apartado se presentan técnicas algebraicas de decodificación de códigos RS o, para ser precisos, de códigos BCH no binarios en general. Básicamente, los algoritmos son una extensión de los que se utilizan para

decodificar códigos BCH, extensión necesaria para resolver un nuevo aspecto: en la decodificación de códigos RS y BCH no binarios no basta con determinar las posiciones erróneas en un vector recibido, sino que también se han de conocer las magnitudes de los símbolos erróneos. De este modo, el esquema general de un procedimiento de decodificación algebraica se compone de tres etapas:

1. Obtención del síndrome.
2. Cálculo del polinomio localizador de errores. Se darán tres métodos para resolver este paso: el algoritmo de Peterson–Gorenstein–Zierler, el de Berlekamp–Massey y el de Sugiyama. El primero de ellos amplía el algoritmo de Peterson para códigos binarios recurriendo al uso de las identidades generalizadas de Newton pero, aparte de éstas, no introduce ninguna novedad conceptual. El algoritmo de Berlekamp–Massey parte, sin embargo, de un enfoque totalmente diferente. Sus fundamentos teóricos son más complicados pero, gracias a su menor complejidad computacional, se emplea mayoritariamente. Por último, se explicará el algoritmo de Sugiyama, que es básicamente una ingeniosa interpretación del algoritmo de división euclídeo. Aunque posee un coste computacional comparable al algoritmo de Berlekamp–Massey, es de aplicación inusual.
3. Cálculo de las magnitudes de error. Para esta tarea se presentan dos posibles métodos: la inversión directa de la matriz de localizadores (que aparece como etapa integrante del algoritmo de Peterson–Gorenstein–Zierler); y el algoritmo de Forney, mucho más eficiente, y cuya secuencia de operaciones se puede encajar en el propio algoritmo de Berlekamp–Massey.

Sea, entonces, un código RS (o BCH no binario) con distancia de diseño $2t+1$ y, sin pérdida de generalidad, con el conjunto de ceros $\{\alpha, \ldots, \alpha^{2t}\}$. Partir de cualquier otro conjunto de $2t$ ceros cíclicamente consecutivos sólo produciría una rotación cíclica del vector de síndrome que se definirá más adelante. Supongamos que el efecto del canal al transmitir un vector del código $v(z)$ es el polinomio de error[3]

$$e(z) = e_1 z^{i_1} + \cdots + e_w z^{i_w}$$

con $\{i_1, \ldots, i_w\}$ las posiciones erróneas ($w \leq t$) y $\{e_1, \ldots, e_w\}$, $e_i \neq 0$ para todo i, las magnitudes de los errores. El vector que se recibe es $r(z) = v(z) + e(z)$.

[3] *a*) En el resto del capítulo utilizaremos polinomios en la variable z para que no haya confusión con la notación de los elementos localizadores. *b*) Esta forma del polinomio error no significa que el canal tenga que transmitir símbolos no binarios; es sólo el procedimiento de decodificación el que se realiza con aritmética no binaria para beneficiarse de la estructura polinómica de los códigos.

Al igual que en el capítulo anterior, se define el *polinomio localizador de errores* por

$$L(z) = \prod_{j=1}^{w}(1 - X_j z), \quad X_j = \alpha^{i_j}.$$

Con esta notación, los elementos del vector síndrome son

$$S_j = r(\alpha^j) = e(\alpha^j) = \sum_{i=1}^{w} e_i X_i^j, \quad j = 1, \ldots, 2t$$

y con ellos se define de manera puramente formal, ya que sólo se conocen $2t$ coeficientes suyos, el *polinomio síndrome*

$$S(z) = \sum_{j=1}^{\infty} S_j z^j.$$

11.4.1. Algoritmo de Peterson–Gorenstein–Zierler

El algoritmo de Peterson para códigos binarios consistía en la resolución de un sistema lineal de ecuaciones, las identidades de Newton (10.7), que relacionaban los coeficientes del polinomio localizador con los del polinomio síndrome. D. Gorenstein y N. Zierler lo generalizaron en 1961 haciendo uso de esta otra versión de las identidades de Newton.

Teorema 11.11 (Identidades de Newton generalizadas). *Dada en un cuerpo* $\mathbb{K}$ *la sucesión*

$$S_j = \sum_{k=1}^{w} e_k X_k^j, \qquad j \geq 0,$$

definida por los elementos $e_k \in \mathbb{K}$ *y* $X_k \in \mathbb{K}$*, para* $k = 1, \ldots, w$*, y dados los coeficientes del polinomio*

$$L(z) = \prod_{j=1}^{w}(1 - X_j z) = \sum_{j=0}^{w} L_j z^j$$

se cumple la recurrencia lineal

$$S_{j+w} + L_1 S_{j+w-1} + \cdots + L_w S_j = 0, \quad \forall j \geq 0. \tag{11.5}$$

En particular, para $j = 1, \ldots, w$*, se verifica el sistema lineal*

$$A_w L = \begin{pmatrix} S_w & S_{w-1} & \ldots & S_1 \\ S_{w+1} & S_w & \ldots & S_2 \\ \ldots & \ldots & \ldots & \ldots \\ S_{2w-1} & S_{2w-2} & \ldots & S_w \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \\ \vdots \\ L_w \end{pmatrix} = - \begin{pmatrix} S_{w+1} \\ S_{w+2} \\ \vdots \\ S_{2w} \end{pmatrix} \tag{11.6}$$

en el que intervienen únicamente los elementos $S_1, \ldots, S_{2w}$ *y las incógnitas* $L_1, \ldots, L_w$.

DEMOSTRACIÓN. Evaluando $L(z)$ en el punto $z = 1/X_i$ y multiplicando por $e_i X_i^{w+j}$, $j \geq 0$, se tiene

$$e_i X_i^{w+j} L\left(\frac{1}{X_i}\right) = \sum_{k=0}^{w} L_k e_i X_i^{w+j-k} = 0$$

que es igual a

$$e_i X_i^{w+j} + L_1 e_i X_i^{w+j-1} + \cdots + L_w e_i X_i^{j} = 0.$$

Sumar estas ecuaciones desde $i = 1$ hasta $i = w$ produce como resultado (11.5). El sistema (11.6) son las w primeras ecuaciones de la recurrencia anterior escritas en forma matricial. ▶

Por construcción, la existencia de una solución del sistema (11.6) es una condición necesaria para que exista un vector de error de peso w con polinomio localizador $L(z)$ y vector de síndrome $(S_1, S_2, \ldots, S_{2t})$. Además, se puede probar que la matriz A_w formada con los elementos del vector síndrome (que es una matriz Toeplitz[4]) es no singular si el vector recibido está a distancia Hamming w de alguna palabra del código, pero es singular si existe algún vector del código a distancia menor que w del vector recibido.

TEOREMA 11.12. *Si* $\mathcal{C}$ *es un código BCH corrector de* t *errores y* $w \leq t$, *entonces la matriz de síndromes*

$$A_m = \begin{pmatrix} S_m & S_{m-1} & \ldots & S_1 \\ S_{m+1} & S_m & \ldots & S_2 \\ \ldots & \ldots & \ldots & \ldots \\ S_{2m-1} & S_{2m-2} & \ldots & S_m \end{pmatrix}$$

es no singular si $m \leq w$ *y es singular cuando* $m > w$.

DEMOSTRACIÓN. A_m se puede descomponer como el producto

$$V \cdot \begin{pmatrix} e_1 X_1 & 0 & 0 \\ 0 & e_2 X_2 & 0 \\ \vdots & & \vdots \\ 0 & & e_m X_m \end{pmatrix} \cdot V^T = V \cdot D \cdot V^T$$

[4] A es una matriz Toeplitz de orden n si $a_{ij} = b_{j-i}$ para ciertos $b_0, b_1, \ldots, b_{n-1}$.

siendo V la matriz

$$V = \begin{pmatrix} 1 & 1 & \dots & 1 \\ X_1 & X_2 & \dots & X_m \\ \vdots & & & \vdots \\ X_1^{m-1} & X_2^{m-1} & \dots & X_m^{m-1} \end{pmatrix}$$

Puesto que V es una matriz de Vandermonde y los localizadores son por definición elementos distintos, V es no singular. Si $m \leq w$, las magnitudes de error son todas no nulas, el determinante de la matriz diagonal es no nulo y $\det(A_m) \neq 0$. Si $m > w$, $\det(D) = 0 \Rightarrow \det(A_m) = 0$. ▶

Por consiguiente, hallando el mayor $w \leq t$ tal que el determinante de A_w no es nulo y resolviendo después

$$A_w \begin{pmatrix} L_1 \\ L_2 \\ \vdots \\ L_w \end{pmatrix} = \begin{pmatrix} -S_{w+1} \\ -S_{w+2} \\ \vdots \\ -S_{2w} \end{pmatrix} \tag{11.7}$$

se obtienen los coeficientes del polinomio localizador cuando el error en el canal tiene peso menor o igual que t. En tal caso, los localizadores $X_1, \dots, X_w$ son los recíprocos de las raíces de $L(z)$.

Cuando el patrón de error tiene peso mayor que t y, además, no es corregible, el sistema (11.7)

- O bien no tiene solución para ningún $w \leq t$ (si las matrices de síndromes son todas singulares).
- O bien el polinomio definido por los coeficientes de la solución no es un polinomio localizador válido. Esto sucede si sus raíces no son simples o no pertenecen al cuerpo $\mathbb{F}_q$.

Se habrá producido entonces un fallo en la decodificación que se detecta con facilidad.

Suponiendo que $w \leq t$, el polinomio $L(z)$ sintetiza toda la información sobre las posiciones de los símbolos erróneos, pero nada dice sobre el valor de estos símbolos. Para averiguar las magnitudes de los errores se aplica uno de los dos métodos que se introducen a continuación.

A. Inversión de la matriz de localizadores

El primer método es directo y consiste en invertir la relación entre los síndromes y los propios elementos localizadores. Una vez conocidos los localizadores $X_1, \ldots, X_w$,

$$\begin{pmatrix} X_1 & X_2 & \ldots & X_w \\ X_1^2 & X_2^2 & \ldots & X_w^2 \\ \ldots & \ldots & \ldots & \ldots \\ X_1^{2w} & X_2^{2w} & \ldots & X_w^{2w} \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_{2w} \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_{2w} \end{pmatrix} \tag{11.8}$$

constituye un sistema lineal de $2w$ ecuaciones y w incógnitas. Pero las w primeras ecuaciones son linealmente independientes, pues la matriz de coeficientes es de Vandermonde no singular por ser todos los elementos localizadores distintos. De modo que la decodificación finaliza tras resolver

$$\begin{pmatrix} X_1 & X_2 & \ldots & X_w \\ X_1^2 & X_2^2 & \ldots & X_w^2 \\ \ldots & \ldots & \ldots & \ldots \\ X_1^w & X_2^w & \ldots & X_w^w \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_w \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_w \end{pmatrix}.$$

En términos de coste computacional, la resolución de este sistema lineal se beneficia del hecho de que la matriz de coeficientes es de tipo Vandermonde y requiere sólo $O(w^2)$ operaciones elementales.

B. Algoritmo de Forney

En el segundo método se sustituye la resolución del sistema lineal anterior por la evaluación de una expresión algebraica.

Supongamos que se define formalmente el polinomio evaluador de errores $E(z)$ como

$$E(z) = L(z) + \sum_{i=1}^{w} z e_i X_i \prod_{\substack{j=1 \\ j \neq i}}^{w} (1 - X_j z).$$

Evaluando $E(z)$ en $z = X_i^{-1}$,

$$E(X_i^{-1}) = L(X_i^{-1}) + \sum_{j=1}^{w} X_i^{-1} X_j e_j \prod_{\substack{k=1 \\ k \neq j}}^{w} (1 - X_k X_i^{-1}) = e_i \prod_{\substack{j=1 \\ j \neq i}}^{w} (1 - X_j X_i^{-1}).$$

Por tanto, si $E(z)$ fuese conocido,

$$e_i = \frac{E(X_i^{-1})}{\prod_{j=1, j \neq i}^{w} (1 - X_j X_i^{-1})} = \frac{-X_i E(X_i^{-1})}{L'(X_i^{-1})}$$

expresión en la que $L'(z)$ representa la derivada formal del polinomio localizador de errores

$$L'(z) = \sum_{j=1}^{w} jL_j z^{j-1}.$$

Pues bien, $E(z)$ es conocido una vez que se conocen $S(z)$ y $L(z)$.

Teorema 11.13 (Ecuación clave (Berlekamp)). *En un código RS (o BCH no binario) de distancia de diseño $\delta = 2t+1$,*

$$E(z) = \big(1 + S(z)\big)L(z) \mod z^{2t+1}. \tag{11.9}$$

Demostración. A partir de las definiciones de $E(z)$ y $L(z)$

$$\frac{E(z)}{L(z)} = 1 + \sum_{i=1}^{w} \frac{ze_iX_i}{1 - zX_i} = 1 + \sum_{i=1}^{w} e_i \sum_{j=1}^{\infty} (zX_i)^j = 1 + S(z) \mod z^{2t+1}.$$

Vea que el grado de $E(z)$ nunca es mayor que el de $L(z)$. La reducción módulo z^{2t+1} significa sencillamente que los coeficientes de los términos de grado menor o igual que $2t$ de los polinomios $E(z)$ y $(1 + S(z))L(z)$ coinciden. ▶

Corolario 11.14 (Forney). *Las componentes del vector de error satisfacen la condición*

$$e_i = \begin{cases} 0 & \text{si } L(\alpha^{-i}) \neq 0, \\ -\dfrac{E(\alpha^{-i})}{\alpha^{-i}L'(\alpha^{-i})} & \text{si } L(\alpha^{-i}) = 0. \end{cases} \tag{11.10}$$

Las ecuaciones (11.10) son el *algoritmo de Forney* para la obtención de las magnitudes de error una vez conocido el polinomio localizador. Una gran ventaja del algoritmo de Berlekamp–Massey, que veremos en el próximo apartado, es que se puede aprovechar su estructura para calcular de paso el polinomio evaluador, sin necesidad de realizar la multiplicación explícita de los polinomios de síndrome y localizador.

En resumen, la decodificación de un código BCH no binario $[n, k]$ sobre $\mathbb{F}_q$ con distancia $2t + 1$ puede llevarse a cabo como sigue.

Algoritmo de Peterson–Gorenstein–Zierler

---

1. Calcular los coeficientes del síndrome, S_j, para $j = 1, \ldots, 2t$.

Vector	Elemento	Vector	Elemento
0000	0	1011	α^7
0001	1	0101	α^8
0010	α	1010	α^9
0100	α^2	0111	α^{10}
1000	α^3	1110	α^{11}
0011	α^4	1111	α^{12}
0110	α^5	1101	α^{13}
1100	α^6	1001	α^{14}

Tabla 11.1. Los elementos del cuerpo $\mathbb{F}_{16}$.

2. Construir la matriz de síndromes A_r (ecuación (11.6)) y hallar el máximo $r \leq t$ tal que A_r tiene determinante distinto de cero.
3. Resolver el sistema (11.6) y formar el polinomio localizador.
4. Hallar las raíces del polinomio localizador: si son todas simples, distintas y pertenecientes a $\mathbb{F}_q$, continuar; en otro caso, señalar un error de decodificación.
5. Formar el polinomio evaluador de errores con la ecuación clave (11.9) y calcular las magnitudes con la fórmula de Forney (ecuación (11.10)) o resolviendo el sistema lineal (11.8).

Ejemplo 11.7. Tomemos el código RS-$[15, 11]$, corrector de errores dobles, engendrado por

$$g(z) = (z-\alpha)(z-\alpha^2)(z-\alpha^3)(z-\alpha^4) = z^4 + \alpha^{13}z^3 + \alpha^6 z^2 + \alpha^3 z + \alpha^{10}$$

siendo α un elemento primitivo de $\mathbb{F}_{16}$ (véase la tabla 11.1).

Supongamos que durante la transmisión de un vector del código se producen los errores descritos por el polinomio de error

$$e(z) = \alpha z^2 + \alpha^3 z$$

y se recibe el vector

$$r(z) = z^4 + \alpha^{13}z^3 + \alpha^{11}z^2 + \alpha^{10}.$$

El polinomio localizador de errores vale, por tanto,

$$L(z) = (1-\alpha^2 z)(1-\alpha z) = 1 + \alpha^5 z + \alpha^3 z^2, \quad L'(z) = \alpha^5$$

y el evaluador

$$E(z) = L(z) + z\alpha^3(1-\alpha z) + z\alpha^4(1-\alpha^2 z) = 1 + \alpha^{13}z + \alpha^{10}z^2.$$

El primer paso de decodificación es la obtención del polinomio síndrome:

$$\begin{aligned} S_1 &= r(\alpha) = e(\alpha) = \alpha^7 \\ S_2 &= r(\alpha^2) = e(\alpha^2) = 0 \\ S_3 &= r(\alpha^3) = e(\alpha^3) = \alpha^{10} \\ S_4 &= r(\alpha^4) = e(\alpha^4) = 1 \end{aligned}$$

y, así, $S(z) = \alpha^7 z + \alpha^{10}z^3 + z^4$. Para hallar el polinomio localizador a partir de $S(z)$, se resuelve el sistema

$$\begin{pmatrix} S_2 & S_1 \\ S_3 & S_2 \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} = - \begin{pmatrix} S_3 \\ S_4 \end{pmatrix}$$

es decir,

$$\begin{pmatrix} 0 & \alpha^7 \\ \alpha^{10} & 0 \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} = \begin{pmatrix} \alpha^{10} \\ 1 \end{pmatrix}$$

de donde se obtiene $L_1 = \alpha^5$ y $L_2 = \alpha^3$. El polinomio evaluador que calcula el decodificador valdrá, entonces,

$$E(z) = \big(1 + S(z)\big)(1 + L_1 z + L_2 z^2) = 1 + \alpha^{13}z + \alpha^{10}z^2 + \cdots$$

y, aplicando el algoritmo de Forney,

$$\begin{aligned} e_1 &= \frac{-\alpha^2 E(\alpha^{-2})}{\alpha^5} = \alpha \\ e_2 &= \frac{-\alpha E(\alpha^{-1})}{\alpha^5} = \alpha^3 \end{aligned}$$

se llega a los valores correctos de las magnitudes de error. ■

Ejemplo 11.8. Tomemos ahora el código RS-$[15, 9]$ engendrado por el polinomio

$$g(z) = \prod_{i=1}^{6} (z - \alpha^i).$$

Utilizando la tabla 11.1, este polinomio generador vale

$$g(z) = \alpha^6 + \alpha^9 z + \alpha^6 z^2 + \alpha^4 z^3 + \alpha^{14} z^4 + \alpha^{10} z^5 + z^6.$$

Supongamos que se recibe el vector

$$y(z) = (\alpha + \alpha^3 z)g(z) + 1 + \alpha z + \alpha^2 z^2 + \alpha^3 z^3 + \alpha^4 z^4$$

correspondiente a cuatro errores de transmisión. El cálculo de los coeficientes del síndrome produce el resultado

$$\begin{aligned} S_1 &= y(\alpha) = \alpha^{12} & S_2 &= y(\alpha^2) = 0 \\ S_3 &= y(\alpha^3) = \alpha^9 & S_4 &= y(\alpha^4) = \alpha^{10} \\ S_5 &= y(\alpha^5) = 0 & S_6 &= y(\alpha^6) = \alpha. \end{aligned}$$

A continuación, el cálculo de los oeficientes del polinomio localizador consiste en resolver el sistema lineal

$$\begin{pmatrix} S_3 & S_2 & S_1 \\ S_4 & S_3 & S_2 \\ S_5 & S_4 & S_3 \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} = - \begin{pmatrix} S_4 \\ S_5 \\ S_6 \end{pmatrix}$$

es decir

$$\begin{pmatrix} \alpha^9 & 0 & \alpha^{12} \\ \alpha^{10} & \alpha^9 & 0 \\ 0 & \alpha^{10} & \alpha^9 \end{pmatrix} \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} = \begin{pmatrix} \alpha^{10} \\ 0 \\ \alpha \end{pmatrix}$$

cuya única solución es $(L_1, L_2, L_3) = (\alpha^{13}, \alpha^{14}, \alpha^9)$, de modo que el polinomio localizador de errores valdrá

$$L(z) = 1 + \alpha^{13} z + \alpha^{14} z^2 + \alpha^4 z^3.$$

Ahora bien, la única raíz de $L(z)$ en $\mathbb{F}_{16}$ es α^5, lo que significa que éste no es un polinomio localizador válido. El algoritmo no puede completarse y se debe de concluir que se han producido más de 3 errores. ■

11.4.2. Algoritmo de Berlekamp–Massey

Las identidades generalizadas de Newton (11.5), si se escriben de la forma

$$S_j = -L_1 S_{j-1} - L_2 S_{j-2} - \cdots - L_w S_0 = -\sum_{k=1}^{w} L_k S_{j-k}, \qquad j \geq w \quad (11.11)$$

representan de manera evidente el hecho de que los elementos del síndrome se pueden generar a partir de los del polinomio localizador por medio de un filtro lineal recurrente con coeficientes $(L_1, \ldots, L_w)$; es decir, con un registro de desplazamiento realimentado. La obtención del polinomio localizador equivale así a la síntesis de un filtro autorregresivo con memoria w mínima tal que, si las condiciones iniciales del mismo son la secuencia S_1, ..., S_w, la salida que se produce a continuación es S_{w+1}, ..., S_{2t}. La figura 11.1 ilustra el circuito que realiza la ecuación (11.11).

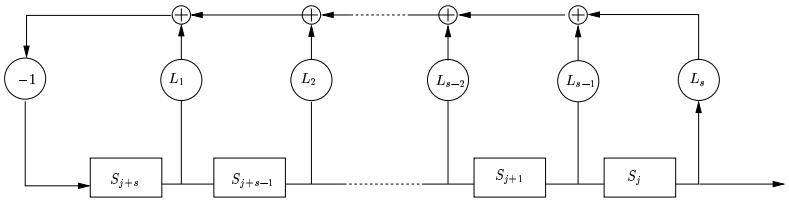


FIGURA 11.1. Filtro lineal recurrente basado en las identidades de Newton.

La idea de interpretar la decodificación como la recurrencia de memoria mínima que produce una salida determinada pertenece a J. L. Massey. Habrá, en principio, muchas soluciones de (11.11) pero Massey demostró que, bajo la condición de poseer memoria mínima, existe un único filtro realimentado de longitud máxima t capaz de generar $S_1, \ldots, S_{2t}$ partiendo de un estado inicial dado. E. Berlekamp y el propio Massey dieron además un algoritmo constructivo de resolución de este problema.

La manera de operar de este algoritmo consiste en calcular iterativamente los coeficientes de realimentación del filtro, comparando las salidas de éste con los coeficientes (conocidos) del síndrome. Si hay alguna diferencia entre ambos, se aplica un factor de corrección. Una vez sintetizado el filtro (o, lo que es igual, hallado el polinomio localizador y verificadas sus raíces), sólo resta aplicar la ecuación (11.10) para poder determinar la magnitud de los errores.

Algoritmo de Berlekamp–Massey

Sin ánimo de pretender una justificación rigurosa del método (cosa que el lector puede hallar en el apéndice 11.B de este capítulo), éstos son los pasos a seguir para decodificar un código BCH no binario con distancia de diseño $2t + 1$:

---

1. Calcular los coeficientes $S_1, \ldots, S_{2t}$ del síndrome.
2. Obtener el polinomio localizador:
   a) Inicializar las variables
   $$i = 0, \quad M_0 = 0, \quad L^{(0)}(z) = 1, \quad T^{(0)}(z) = 1$$
   b) $i = i + 1$; calcular la diferencia, Δ_i, entre el síndrome S_i y el i-ésimo símbolo de salida del filtro, de memoria M_i, definido por

el polinomio $L^{(i-1)}(z)$:

$$\Delta_i = \sum_{j=0}^{i-1} L_j^{(i-1)} S_{i-j}.$$

Aplicar las ecuaciones recurrentes

$$M_i = \delta_i(i - M_{i-1}) + (1 - \delta_i)M_{i-1}$$

$$\begin{pmatrix} L^{(i)}(z) \\ T^{(i)}(z) \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_i z \\ \Delta_i^{-1}\delta_i & (1-\delta_i)z \end{pmatrix} \begin{pmatrix} L^{(i-1)}(z) \\ T^{(i-1)}(z) \end{pmatrix}$$

con $\delta_i = 1$ si $\Delta_i \neq 0$ y $2M_{i-1} \leq i - 1$, y $\delta_i = 0$ en otro caso.

c) si $i < 2t$, volver al paso (b)

3. Hallar las raíces de $L(x) = L^{(2t)}(x)$; si no son simples, distintas o no pertenecen al cuerpo, señalar fallo en la decodificación.

4. Obtener el polinomio evaluador de errores con la ecuación (11.9).

5. Determinar la magnitud de los errores.

El polinomio evaluador de error se puede hallar también por medio de la recurrencia matricial:

$$\begin{pmatrix} E^{(k)}(z) \\ A^{(k)}(z) \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_k z \\ \Delta_k^{-1}\delta_k & (1-\delta_k)z \end{pmatrix} \begin{pmatrix} E^{(k-1)}(z) \\ A^{(k-1)}(z) \end{pmatrix}$$

con las condiciones iniciales $A^{(0)}(z) = 0$ y $E^{(0)}(z) = 1$. Tras $2t$ iteraciones, $E(z) = E^{(2t)}(z)$ es el polinomio buscado.

En cuanto a la complejidad de los métodos de decodificación algebraica, se debe observar que el paso crucial es la obtención del polinomio localizador a partir del polinomio síndrome. Mientras que el algoritmo de Peterson–Gorenstein–Zierler resuelve, para ello, un sistema de t ecuaciones lineales y requiere $O(t^3)$ operaciones, el de Berlekamp–Massey aplica una recurrencia lineal que termina en $2t$ iteraciones y requiere solamente operaciones de suma y multiplicación en $\mathbb{F}_q$. Son, en el peor de los casos, $2t$ multiplicaciones por iteración para actualizar la matriz y no más de t multiplicaciones por iteración para calcular la discrepancia Δ_i. Por tanto, el algoritmo precisa un máximo de $6t^2$ multiplicaciones. Para hallar las magnitudes de error se puede resolver un nuevo sistema lineal de ecuaciones o aplicar la fórmula de Forney. Se prefiere la fórmula de Forney porque el algoritmo de Berlekamp–Massey se puede modificar sin incremento de su complejidad para que produzca directamente, además del polinomio localizador, el polinomio evaluador. Un

pequeño inconveniente del algoritmo de Forney es que exige la división entre elementos de $\mathbb{F}_q$, que es una operación costosa en comparación con la suma o la multiplicación. Una manera muy sencilla de soslayar este problema es utilizar una tabla de inversos de los elementos de $\mathbb{F}_q$. Si el cardinal del cuerpo es muy elevado, otra posibilidad mejor es la de elegir adecuadamente la base en que se representan los elementos del cuerpo para poder efectuar las divisiones con sólo unas pocas operaciones elementales.

Ejemplo 11.9. Para ilustrar el algoritmo, tomemos el código RS-$[7,3]$ generado por

$$g(z) = (z-\alpha)(z-\alpha^2)(z-\alpha^3)(z-\alpha^4) = z^4 + \alpha^3 z^3 + z^2 + \alpha z + \alpha^3.$$

Si suponemos que se recibe la secuencia $r(z) = \alpha^2 z^6 + \alpha^2 z^4 + z^3 + \alpha^5 z^2$, el decodificador calcula en primer lugar su polinomio síndrome $S(z) = \alpha^6 z + \alpha^3 z^2 + \alpha^4 z^3 + \alpha^3 z^4$. Conocidos los síndromes, la ejecución del algoritmo de Berlekamp–Massey se resume en esta tabla:

i	$L^{(i)}(z)$	Δ_i	δ_i	M_i	$T^{(i)}(z)$
0	1	—	—	0	1
1	$1-\alpha^6 z$	α^6	0	1	α
2	$1-\alpha^4 z$	α^2	0	1	αz
3	$1-\alpha^4 z-\alpha^6 z^2$	α^5	1	2	$\alpha^2-\alpha^6 z$
4	$1+\alpha^2 z+\alpha z^2$	α^6	0	—	—

El polinomio localizador que se estima es, por tanto, $1+\alpha^2 z+\alpha z^2$, cuyas raíces son, en $\mathbb{F}_8$, los elementos α^2 y α^4. La ecuación clave se aplica inmediatamente para deducir que

$$E(z) = \big(1+S(z)\big)L(z) \mod z^5 \Rightarrow E(z) = \alpha^3 z^2 + z + 1.$$

Y de aquí se obtiene, por la fórmula de Forney, el polinomio de error

$$e(z) = \alpha^5 z^5 + \alpha z^3.$$

Se puede verificar sin dificultad que

$$r(z) + e(z) = \alpha^2 z^2 g(z).$$

El código es corrector de errores dobles, de modo que, si el error de transmisión es $e(z)$, se habrán subsanado. ■

11.4.3. Algoritmo euclídeo o de Sugiyama

El tercer método para hallar los coeficientes del polinomio localizador a partir de los del síndrome presenta cierta conexión con el de Berlekamp–Massey, y se comporta en cierta forma como una generalización de aquél, ya

que sirve asimismo para la decodificación de los códigos Goppa, una clase que incluye a los BCH.

El punto de partida es la ecuación clave de decodificación, (11.9), que implica la existencia de un polinomio $Q(z)$ tal que

$$\big(1+S(z)\big)L(z)+Q(z)z^{2t+1}=E(z). \tag{11.12}$$

Al formularlo de esta manera, el problema de la decodificación puede entenderse como el de hallar un $L(z)$ y un $E(z)$, ambos de grado mínimo y menor o igual que t, que verifiquen (11.12). El algoritmo euclídeo de división de polinomios resuelve este problema, hecho que observó por primera vez Y. Sugiyama en 1975 [66]. Vamos a precisar a continuación cómo lo consigue.

Teorema 11.15 (Algoritmo de división euclídeo). *El mayor divisor común de los polinomios $\pi_{-1}(z)$ y $\pi_0(z)$, siendo $\pi_0(z)$ el de menor grado de ambos, es el último resto $\pi_k(z)$ no nulo de la sucesión de divisiones*

$$\pi_{k-2}(z)=q_k(x)\pi_{k-1}(z)+\pi_k(z),\quad k\geq 1.$$

Dadas las condiciones iniciales $U_{-1}(z)=V_0(z)=0$ y $V_{-1}(z)=U_0(z)=1$, las sucesiones de polinomios definidas por las relaciones de recurrencia

$$\begin{aligned}U_i(z)&=U_{i-2}(z)+q_i(z)U_{i-1}(z)\\ V_i(z)&=V_{i-2}(z)+q_i(z)V_{i-1}(z)\end{aligned}$$

verifican

$$(-1)^i\big(U_i(z)\pi_0(z)-V_i(z)\pi_{-1}(z)\big)=\pi_i(z).$$

Demostración. Vamos a probar únicamente la identidad

$$(-1)^i\big(U_i(z)\pi_0(z)-V_i(z)\pi_{-1}(z)\big)=\pi_i(z).$$

Escribiendo la recurrencia

$$\begin{aligned}U_i(z)&=U_{i-2}(z)+q_i(z)U_{i-1}(z)\\ V_i(z)&=V_{i-2}(z)+q_i(z)V_{i-1}(z)\end{aligned}$$

en forma matricial, se tiene que

$$\begin{aligned}\begin{pmatrix}U_i(z) & U_{i-1}(z)\\ V_i(z) & V_{i-1}(z)\end{pmatrix}&=\begin{pmatrix}U_{i-1}(z) & U_{i-2}(z)\\ V_{i-1}(z) & V_{i-2}(z)\end{pmatrix}\begin{pmatrix}q_i(z) & 1\\ 1 & 0\end{pmatrix}=\cdots\\ &=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}q_1(z) & 1\\ 1 & 0\end{pmatrix}\cdots\begin{pmatrix}q_i(z) & 1\\ 1 & 0\end{pmatrix}.\end{aligned} \tag{11.13}$$

Escribiendo el algoritmo euclídeo también en forma matricial

$$\begin{pmatrix} \pi_{i-2}(z) \\ \pi_{i-1}(z) \end{pmatrix} = \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \pi_{i-1}(z) \\ \pi_i(z) \end{pmatrix}$$

e iterando, resulta

$$\begin{pmatrix} \pi_{-1}(z) \\ \pi_0(z) \end{pmatrix} = \begin{pmatrix} q_1(z) & 1 \\ 1 & 0 \end{pmatrix} \cdots \begin{pmatrix} q_{i-1}(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \pi_{i-1}(z) \\ \pi_i(z) \end{pmatrix}. \tag{11.14}$$

El determinante de la matriz que aparece en el segundo miembro de (11.13) vale $(-1)^i$. Por lo tanto, combinando (11.13) y (11.14) se obtiene

$$\begin{pmatrix} \pi_{i-1}(z) \\ \pi_i(z) \end{pmatrix} = (-1)^i \begin{pmatrix} V_{i-1}(z) & -U_{i-1}(z) \\ -V_i(z) & U_i(z) \end{pmatrix} \begin{pmatrix} \pi_{-1}(z) \\ \pi_0(z) \end{pmatrix}.$$

La segunda de estas ecuaciones es precisamente

$$\pi_i(z) = (-1)^i \big(U_i(z)\pi_0(z) - V_i(z)\pi_{-1}(z) \big). \qquad \blacktriangleright$$

Vea, entonces, a partir de esta última expresión para $\pi_i(z)$, que si identificamos $\pi_{-1}(z) = z^{2t+1}$ y $\pi_0(z) = 1 + S(z)$, el algoritmo de división euclídeo genera soluciones sucesivas de la ecuación (11.12). Por consiguiente, para obtener el polinomio localizador $L(z)$ basta iterar hasta que el grado de $\pi_i(z)$ sea menor o igual que t.

He aquí entonces las instrucciones para hallar $E(z)$ y $L(z)$ a partir de $S(z)$.

Algoritmo de Sugiyama

1. Iterar el algoritmo de división euclídeo para $\pi_{-1}(z) = z^{2t+1}$ y $\pi_0(z) = 1 + S(z) = 1 + \sum_{j=1}^{s} S_j z^j$ hasta alcanzar un resto $\pi_k(z)$ tal que

   $$\text{grado}\, \pi_{k-1}(z) > t \quad \text{grado}\, \pi_k(x) \leq t.$$

2. Obtener el polinomio $A_k(z)$ de la iteración

   $$A_j(z) = q_j(z)A_{j-1}(z) + A_{j-2}(z), \qquad j \geq 1$$

   con las condiciones iniciales $A_{-1}(z) = 0$, $A_0(z) = 1$.

3. Calcular

   $$E(z) = (-1)^k \delta \pi_k(z) \tag{11.15}$$
   $$L(z) = \delta A_k(z) \tag{11.16}$$

   siendo δ la constante que hace $L(0) = 1$.

Es posible demostrar, aunque no se hará en este libro,[5] que (11.15) y (11.16) son las únicas soluciones de

$$E(z) = L(z)\big(1 + S(z)\big) \mod z^{2t+1}$$

que cumplen $L(0) = 1$, grado $L(z) \leq t$, grado $E(z) \leq t$ y con el grado de $E(z)$ el menor posible.

La recurrencia de Berlekamp–Massey es en realidad un método eficiente de aplicar el algoritmo euclídeo sin necesidad de efectuar multiplicaciones y divisiones polinómicas, por lo que su implementación con registros de desplazamiento es simple. Se ha podido demostrar que el algoritmo euclídeo directo es computacionalmente menos costoso que el de Berlekamp a partir de longitudes de código del orden de 10^6 (que en el momento presente se encuentran fuera de todo rango práctico), aunque exige la multiplicación y división explícita de polinomios. De cualquier manera, no es ahí donde reside su mayor virtud, sino en el hecho de que resulta conceptualmente mucho más fácil de comprender que el algoritmo de Berlekamp–Massey.

Ejemplo 11.10. Tomemos, de nuevo, el código RS-$[15, 11]$ generado por

$$g(z) = z^4 + \alpha^{13}z^3 + \alpha^6 z^2 + \alpha^3 z + \alpha^{10}$$

y supongamos esta vez el polinomio de error $e(z) = \alpha^3 z^5 + \alpha^7 z^2$. El polinomio síndrome asociado valdrá $S(z) = \alpha^2 z^4 + \alpha^8 z^3 + \alpha^4 z^2 + \alpha^{12} z$, y los polinomios localizador y evaluador

$$\begin{aligned} L(z) &= (1 - \alpha^5 z)(1 - \alpha^2 z) = 1 + \alpha z + \alpha^7 z^2 \\ E(z) &= L(z) + \alpha^8 z(1 - \alpha^2 z) + \alpha^9 z(1 - \alpha^5 z) = 1 + \alpha^{13} z + \alpha^8 z^2. \end{aligned}$$

Se verifica, por supuesto, $E(z) = (1+S(z))L(z) \mod z^5$, pero veamos cómo se hallan $E(z)$ y $L(z)$ aplicando el algoritmo euclídeo. Las dos primeras iteraciones son

$$\begin{aligned} z^5 &= \big(1 + S(z)\big)(\alpha^{13} z + \alpha^4) + \alpha^7 z^3 + \alpha z^2 + \alpha^{12} z + \alpha^4 \\ 1 + S(z) &= (\alpha^7 z^3 + \alpha z^2 + \alpha^{12} z + \alpha^4)(\alpha^{10} z + 1) + \alpha^9 z^2 + \alpha^{14} z + \alpha \end{aligned}$$

y el grado del resto de esta última división es menor o igual que $t = 2$; así, $\alpha^9 z^2 + \alpha^{14} z + \alpha$ debe ser un múltiplo constante de $E(z)$. Para hallar la constante de proporcionalidad, basta aplicar la recurrencia

$$A_j(z) = q_j(z) A_{j-1}(z) + A_{j-2}(z)$$

[5] La demostración puede hallarse en [41], págs. 363–364.

con las condiciones iniciales $A_{-1}(z) = 0$ y $A_1(z) = 1$. Puesto que sólo ha sido necesario dividir dos veces, se tiene que

$$A_1(z) = q_1(z) = \alpha^{13}z + \alpha^4$$
$$A_2(z) = q_2(z)A_1(z) + 1 = \alpha^8 z^2 + \alpha^2 z + \alpha.$$

Y como $\alpha^{-1}A_2(0) = 1$, se deduce que

$$L(z) = \alpha^{-1}A_2(z) = \alpha^7 z^2 + \alpha z + 1$$
$$E(z) = \alpha^{-1}(\alpha^9 z^2 + \alpha^{14} z + \alpha) = \alpha^8 z^2 + \alpha^{13} + 1$$

como era lógico esperar. Las magnitudes de los errores se recuperarán preferiblemente con el algoritmo de Forney. ■

11.5. Decodificación por lista

M. Sudan y V. Guruswami [30] han ideado un algoritmo alternativo capaz de corregir cualquier fracción de errores menor o igual que $1 - \sqrt{R}$ con un código Reed-Solomon de tasa R. Este algoritmo, por tanto, supera la capacidad de corrección del algoritmo de Peterson-Gorenstein-Zierler y, para cualquier valor de la tasa $R \in [0, 1)$, corrige una fracción de errores superior a la clásica cota $(1 - R)/2$.

La idea principal del algoritmo consiste en considerar el vector recibido como una colección de puntos del espacio bidimensional $\mathbb{F}_q^2$ y en encontrar un polinomio de interpolación en dos variables $A(x, y)$ que pase por todos estos puntos. Bajo determinadas condiciones, los factores de $A(x, y)$ de la forma $y - f(x)$, en donde $f(x)$ es un polinomio con coeficientes en $\mathbb{F}_q$, identifican las palabras del código situadas a distancia relativa menor que $t = 1 - \sqrt{k/n}$ del vector recibido. Veamos el desarrollo del algoritmo de Sudan-Guruswami.

Sea $\mathbb{F}_q[x]$ el anillo de los polinomios con coeficientes del cuerpo $\mathbb{F}_q$ en la indeterminada x. Sean $\alpha_1, \ldots, \alpha_n$ elementos no nulos de $\mathbb{F}_q$ dados en cierto orden, y recordemos que los códigos Reed-Solomon se pueden definir como la evaluación de cierta clase de polinomios en estos puntos. Específicamente, el código Reed-Solomon $[n, k]$ es el conjunto

$$\mathcal{C}[n, k] = \left\{ \big(f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_n)\big) : f(x) \in \mathbb{F}_q[x], \text{grado}\big(f(x)\big) < k \right\}.$$

Definición 11.2. *Sea $A(x, y) = \sum_{i,j} a_{i,j} x^i y^j$ un polinomio de $\mathbb{F}_q[x, y]$ y sean ω_x y ω_y dos números enteros no negativos. El grado ponderado-(ω_x, ω_y) del polinomio en dos variables $A(x, y)$ es*

$$\text{máx}\{i\omega_x + j\omega_y : i \geq 0, j \geq 0, a_{i,j} \neq 0\}.$$

El grado ponderado-$(1,1)$ de $A(x,y)$ es sin más el grado de $A(x,y)$. El número de monomios de grado ponderado-(ω_x,ω_y) menor o igual que δ se denota por $N_{\omega_x,\omega_y}(\delta)$. El valor de $N_{1,k}(\delta)$ resulta de especial importancia para el algoritmo y se puede calcular con un simple argumento de cuenta de monomios [12].

LEMA 11.16.

$$N_{1,k}(\delta) = \left\lceil \frac{\delta+1}{k} \right\rceil \left(\delta + 1 - \frac{k}{2} \left\lfloor \frac{\delta}{k} \right\rfloor \right) > \frac{\delta^2}{2k}.$$

DEMOSTRACIÓN.

$$\begin{aligned} N_{1,k}(\delta) &= \big|\{i,j \geq 0 : i + jk \leq \delta\}\big| = \sum_{j=0}^{\lfloor \delta/k \rfloor} (\delta - jk + 1) \\ &= \left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1 \right) \left(\delta + 1 - \frac{k}{2} \left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1 \right) \right) \\ &= \left\lceil \frac{\delta+1}{k} \right\rceil \left(\delta + 1 - \frac{k}{2} \left\lceil \frac{\delta+1}{k} \right\rceil \right) \end{aligned}$$

en donde la última igualdad se sigue inmediatamente de la identidad $\lfloor \delta/k \rfloor + 1 = \lceil (\delta+1)/k \rceil$. ▶

La cota inferior es, de hecho, un caso particular del resultado más general $N_{\omega_x,\omega_y}(\delta) > \delta^2/2\omega_x\omega_y$.

Si $\mathbf{y} = \mathbf{x} + \mathbf{e} = (y_1, \ldots, y_n)$ es el vector recibido a la salida del canal, los pares

$$\mathcal{D} = \{(\alpha_1, y_1), (\alpha_2, y_2), \ldots, (\alpha_n, y_n)\}$$

pueden verse como una colección de puntos del espacio afín bidimensional $\mathbb{F}_q^2$. Observe que, en ausencia de errores de transmisión, $y_i = f(\alpha_i)$ para un único $f(x) \in \mathbb{F}_q[x]$.

DEFINICIÓN 11.3. *El polinomio $A(x,y)$ pasa por el punto (α,β) si $A(\alpha,\beta) = 0$. El polinomio $A(x,y)$ pasa por el punto (α,β) con multiplicidad m si $A(x+\alpha, y+\beta)$ contiene un monomio de grado m pero ningún monomio de grado menor que m.*

El concepto de multiplicidad de una raíz de $A(x,y)$ es una generalización del familiar para polinomios en una sola variable. Si para $f(x) \in \mathbb{F}_q[x]$ es $f(\alpha) = 0$ con multiplicidad m entonces $f(x) = (x-\alpha)^m g(x)$, con $g(\alpha) \neq 0$. O, de manera equivalente, $f(x+\alpha) = x^m h(x)$ con $h(0) \neq 0$, y $f(x+\alpha)$ no contiene ningún término de grado inferior a m. Con los polinomios de

$\mathbb{F}_q[x,y]$ no es posible separar del mismo modo las raíces, pero sí se mantiene la propiedad de desplazamiento, en el sentido de que $A(x+\alpha, y+\beta)$ no contiene ningún monomio de grado inferior a la multiplicidad de (α,β).

En el problema de la decodificación de códigos Reed-Solomon, si han ocurrido errores de transmisión e $y_i \neq f(\alpha_i)$ para algún $1 \leq i \leq n$, un polinomio de interpolación en dos variables que pase por todos los puntos de $\mathcal{D}$ tendrá la forma $a(x,y) = p(x,y)(y - f(x))$ en donde $p(\alpha_j, f(\alpha_j)) = 0$ para todas las coordenadas j en donde se haya producido error y $f(x)$ sería un polinomio del código. Por tanto, si se pudiese calcular $a(x,y)$, sus factores de la forma $y - f(x)$ permitirían identificar cierto número de palabras del código entre las que puede hallarse el vector transmitido. El algoritmo de Sudan-Guruswami resuelve precisamente estos dos pasos, imponiendo algunas condiciones al polinomio $a(x,y)$ para garantizar la corrección de los errores.

Algoritmo de Sudan-Guruswami

---

1. Fijemos un entero arbitrario m y elijamos δ como el menor entero positivo tal que
$$N_{1,k-1}(\delta) > \frac{nm(m+1)}{2}.$$
Sea $t = \left\lfloor \frac{\delta}{m} \right\rfloor + 1$.

2. Paso de interpolación. Calcular un polinomio $p(x,y) \in \mathbb{F}_q[x,y]$ de grado ponderado-$(1, k-1)$ como máximo δ que pase por cada elemento de
$$\mathcal{D} = \{(\alpha_1, y_1), (\alpha_2, y_2), \ldots, (\alpha_n, y_n)\}$$
con multiplicidad al menos m, en donde $(y_1, \ldots, y_n)$ es el vector recibido.

3. Paso de factorización. Identificar todos los factores de $p(x,y)$ de la forma $y - f(x)$, con $f(x) \in \mathbb{F}_q[x]$ cumpliendo $y_i = f(\alpha_i)$ para al menos t de los puntos de $\mathcal{D}$.

Para verificar que el algoritmo es correcto y hallar sus condiciones de validez se precisan los siguientes resultados.

Lema 11.17. *Sea $(\alpha,\beta) \in \mathbb{F}_q^2$ una raíz de $p(x,y) \in \mathbb{F}_q[x,y]$ con multiplicidad m o mayor. Si $f(x) \in \mathbb{F}_q[x]$ y $f(\alpha) = \beta$ entonces el polinomio $g(x) = p(x, f(x))$ es divisible por $(x-\alpha)^m$.*

DEMOSTRACIÓN. Sea $f_1(x) = f(x+\alpha) - \beta$. Como $f_1(0) = 0$, $f_1(x) = xf_2(x)$ para cierto $f_2(x) \in \mathbb{F}_q[x]$. Si definimos $g_1(x) = p(x+\alpha, f_1(x)+\beta)$, como (α, β) es raíz de $p(x,y)$ con multiplicidad no menor que m, entonces $p(x+\alpha, y+\beta)$ no posee ningún monomio de grado inferior a m. Fijando $y = f_1(x) = xf_2(x)$ en $p(x+\alpha, y+\beta)$ se deduce que $g_1(x)$ es divisible por x^m lo que a su vez implica que $g_1(x-\alpha)$ es divisible por $(x-\alpha)^m$. Ahora bien, $g_1(x-\alpha) = p(x, f_1(x-\alpha)+\beta) = p(x, f(x)) = g(x)$, lo que demuestra que $g(x)$ es divisible por $(x-\alpha)^m$. ▶

LEMA 11.18. *Sean m, t y δ tres enteros positivos tales que $mt > \delta$ y sea $p(x,y) \in \mathbb{F}_q[x,y]$ tal que (α_i, y_i) es una raíz suya con multiplicidad al menos m, para $1 \le i \le n$. Supóngase que el grado ponderado-$(1, k-1)$ de $p(x,y)$ no es mayor que δ. En estas condiciones, si $f(x)$ es un polinomio de $\mathbb{F}_q[x]$ con grado menor que k que cumple $y_i = f(\alpha_i)$ para al menos t índices de $\{1, 2, \ldots, n\}$ entonces $y - f(x)$ es factor de $p(x,y)$.*

DEMOSTRACIÓN. Puesto que $p(x,y)$ tiene grado ponderado-$(1, k-1)$ menor o igual que δ, $g(x) = p(x, f(x))$ es o bien el polinomio nulo o bien un polinomio de grado δ como máximo. Si $g(x)$ fuese no nulo entonces, en virtud del lema anterior, $(x-\alpha_i)^m$ dividiría a $g(x)$ para cualquier $i \in T = \{j : 1 \le j \le n, f(\alpha_j) = y_j\}$, y como los elementos α_i son todos distintos se tendría que $h(x) = \prod_{i \in T}(x-\alpha_i)^m$ divide a $g(x)$. Pero $h(x)$ es de grado al menos $mt > \delta$, lo que constituye una contradicción. Por lo tanto, $g(x) = 0$ y el polinomio $y - f(x)$ es un factor de $p(x,y)$. ▶

Veamos, por último, una relación combinatoria entre los coeficientes de $p(x,y)$ y los del polinomio desplazado $p(x+\alpha, y+\beta)$.

LEMA 11.19. *Dado $(\alpha, \beta) \in \mathbb{F}_q^2$, sean $p(x,y) = \sum_{i,j} p_{i,j} x^i y^j$, $p(x+\alpha, y+\beta) = \sum_{i,j} p'_{i,j} x^i y^j$. Se cumple que*

$$p'_{i,j} = \sum_{k=i}^{\infty} \sum_{l=j}^{\infty} \binom{k}{i} \binom{l}{j} \alpha^{k-i} \beta^{l-j} p_{k,l}.$$

DEMOSTRACIÓN. Desarrollando los binomios se tiene directamente

$$\sum_{i,j} p_{i,j} (x+\alpha)^i (y+\beta)^j = \sum_{i,j} p_{i,j} \sum_{k=0}^{i} x^k \alpha^{i-k} \sum_{l=0}^{j} y^l \beta^{j-l}.$$

Efectuando un cambio de variable y reuniendo términos se llega a la fórmula del lema. ▶

Con estas piezas podemos enunciar el resultado fundamental que justifica el algoritmo de Sudan-Guruswami.

Teorema 11.20. *Dados un vector recibido* $\mathbf{y} = (y_1, \ldots, y_n)$ *y un entero* m*, el paso de factorización del algoritmo de Sudan-Guruswami produce una lista con todas las palabras del código RS-*$[n, k]$ *a distancia menor que*

$$e = n - \left\lfloor \frac{\delta}{m} \right\rfloor > \left\lfloor n \left(1 - \sqrt{R \frac{m+1}{m}} \right) \right\rfloor$$

de $\mathbf{y}$*, siendo* $R = k/n$ *la tasa de codificación.*

Demostración. Una precondición para que el teorema sea válido es que exista el polinomio $p(x, y)$ buscado en la etapa de interpolación. Para ello es necesario que $p(x + \alpha_s, y + y_s)$ no posea términos de grado menor que m, para $1 \leq s \leq n$. Según el lema 11.19 esto ocurre si

$$\sum_{k=i}^{\infty} \sum_{l=j}^{\infty} \binom{k}{i} \binom{l}{j} \alpha^{k-i} \beta^{l-j} p_{k,l} = 0 \tag{11.17}$$

para todo $i, j \geq 0$ tales que $i + j < m$.

Para cualquier valor de s existen $m(m+1)/2$ ecuaciones del tipo (11.17) y por tanto un total de $nm(m+1)/2$ ecuaciones lineales homogéneas (no todas independientes, posiblemente) en las incógnitas $p_{i,j}$. Y como el número total de incógnitas es $N_{1,k-1}(\delta)$, ya que $p(x, y)$ se busca con grado ponderado-$(1, k-1)$ menor o igual que δ, el sistema tendrá una solución no trivial si

$$N_{1,k-1}(\delta) > \frac{nm(m+1)}{2}$$

que es el primer paso del algoritmo.

Ahora, como $t = \lfloor \delta/m \rfloor + 1$ y $mt > \delta$, si $f(x) \in \mathbb{F}_q[x]$ es un polinomio que cumple $f(\alpha_i) = y_i$ para al menos t puntos, el lema 11.18 garantiza que $y - f(x)$ es un factor de $p(x, y)$ y el algoritmo produce las palabras del código a distancia menor que $e = n - t + 1 = n - \left\lfloor \frac{\delta}{m} \right\rfloor$ de $\mathbf{y}$. Además, por cómo se elige δ, se tiene que

$$N_{1,k-1}(\delta - 1) \leq \frac{nm(m+1)}{2}.$$

Pero $N_{1,k-1}(\delta - 1) > \delta^2/2(k-1) > \delta^2/2k$ y por tanto

$$\frac{\delta^2}{2k} \leq \frac{nm(m+1)}{2}$$

y

$$\frac{\delta}{m} \leq n \sqrt{\frac{k}{n} \cdot \frac{m+1}{m}}$$

o lo que es igual

$$\frac{t}{n} > 1 - \sqrt{R\frac{m+1}{m}}.$$ ▶

Observe que, según este teorema, $t/n > 1-\sqrt{R}$ si $m \to \infty$. Sin embargo, aumentar m rápidamente incrementa el número de ecuaciones que es preciso resolver para hallar $p(x, y)$.

En la literatura técnica se pueden encontrar varios algoritmos computacionalmente eficientes para los pasos de interpolación y factorización necesarios en el algoritmo de Sudan-Guruswami. Más aún, estos algoritmos poseen complejidad polinómica, de donde se desprende que la decodificación por lista también. Otra ventaja adicional de esta estrategia de decodificación es que se puede aplicar a cierta clase de códigos BCH, a los códigos algebraico-geométricos y a los códigos Reed-Solomon generalizados que definiremos en el apartado 11.7. Por último, el algoritmo ha sido extendido al caso de la decodificación *soft* [36].

EJEMPLO 11.11. El código RS-$[255, 144, 112]$ sobre $\mathbb{F}_{256}$ tiene una capacidad de corrección de 55 errores. Si fijamos $m = 1$ el algoritmo de Sudan puede corregir hasta 55 errores. A medida que m aumenta lo hace igualmente la capacidad de corrección del algoritmo de Sudan, según refiere la tabla adjunta, en la que no se han indicado los valores intermedios de m para los que no varía el número de errores corregibles.

m	δ	e	Densidad
1	200	55	$3{,}71 \cdot 10^{-79}$
2	398	56	$3{,}38 \cdot 10^{-76}$
3	592	58	$2{,}62 \cdot 10^{-70}$
4	782	60	$1{,}86 \cdot 10^{-64}$
6	1164	61	$1{,}51 \cdot 10^{-61}$
7	1357	62	$1{,}21 \cdot 10^{-58}$
12	2314	63	$9{,}45 \cdot 10^{-56}$

La última columna indica el número medio de palabras del código contenidas en una bola de radio e elegida al azar. Los valores significan que es virtualmente imposible cometer un error de decodificación cuando se producen tal número de errores o, lo que es lo mismo, que el código es capaz de corregir prácticamente todos los errores de peso e. Vea que el paso de 62 a 63 errores corregibles requiere un incremento de m desde 7 hasta 12, lo que podría suponer una carga computacional elevada en los pasos de interpolación y factoriazación. ■

11.6. Decodificación con símbolos borrados

En algunos canales reales se presentan durante la transmisión, en ocasiones, periodos en los que el subsistema encargado de detectar y procesar la señal recibida no es capaz de decidir con una fiabilidad mínima cuál fue el símbolo emitido. Por ejemplo, la calidad de la señal puede haberse deteriorado, por lo general a causa del ruido o debido a la pérdida de sincronismo, hasta hacer irrecuperables uno o más símbolos. En estas circunstancias, una estimación de los mismos no sólo sería inservible (en el sentido de que la probabilidad de acierto no es mayor que la de fallo), sino que, lo que es aún peor, destruiría información: la incertidumbre acerca de los símbolos transmitidos sería entonces mayor que si no se hubiese resuelto la decisión, porque al hacerlo se pierde irreversiblemente la localización del error.

En lugar de obligar al receptor de la señal a realizar decisiones que con gran probabilidad pueden ser incorrectas, se podría optar por utilizar un símbolo ajeno al alfabeto de codificación con el que señalar al decodificador de canal la presencia de un símbolo erróneo en la secuencia recibida que no se sabe cómo clasificar. Vista desde el decodificador, se tiene de este modo la misma situación que se daría si algunos símbolos de la secuencia código se hubiesen borrado o eliminado, y otros se hubiesen transmitido con error. Siguiendo con esta interpretación, un símbolo borrado es a todos los efectos equivalente a un error del que se conoce la posición aunque se ignora la magnitud.

Considerar la presencia de símbolos borrados no modifica la abstracción ideal de canal discreto, noción con la que ha sido posible desarrollar una amplia variedad de técnicas de decodificación basadas solamente en las propiedades algebraicas de los códigos. Pero acontece en la realidad, en cambio, que el ruido en un canal es un fenómeno continuo, y lo que se ha transmitido como un símbolo discreto se convierte, a la salida del canal, en una observación de una variable aleatoria continua. Hasta el momento se ha venido suponiendo, por lo tanto, que estas observaciones continuas eran discretizadas por alguna clase de dispositivo antes de ser procesadas por el decodificador. Los que así operan se conocen como decodificadores de canal basados en decisiones *hard*. Como alternativa lógica, cabe pensar que un decodificador que procesara directamente los valores reales (continuos) observados a la salida del canal obtendría mejores prestaciones (menor probabilidad de equivocación), dado que en el proceso de discretización siempre se pierde información. Este argumento es esencialmente correcto y los decodificadores que aceptan a su entrada vectores de números reales (vectores que no enmascaran la magnitud del ruido en la señal recibida) se conocen como decodificadores basados en decisiones *soft*.

Las técnicas de decodificación *soft* merecen un tratamiento específico,

que no se va a abordar en este texto. Son intrínsecamente distintas de las técnicas *hard*, por cuanto prescinden de toda estructura algebraica o geométrica presente en el código y manejan tan sólo operadores sobre funciones de densidad de probabilidad. Además, comportan que el diseño del decodificador no sea independiente de las señales que representan los símbolos de una fuente. Sin embargo, se han discutido brevemente aquí porque guardan cierta relación con los modelos de canal con símbolos borrados. Éstos se pueden interpretar como canales discretos en los que las salidas se clasifican en un número diferente de niveles (o símbolos) que las entradas, de manera que cada uno de estos niveles corresponde a diferentes grados de información (o bien de incertidumbre) acerca del símbolo emitido. De acuerdo con este punto de vista, la decodificación *soft* aparece como el límite continuo de un clasificador con un número cada vez mayor de grados de clasificación.

En este libro se van a ignorar los métodos algorítmicos de decodificación *soft*. Consecuentemente, el objetivo del presente apartado será analizar en qué casos es posible, y cómo, reconstruir un vector del código cuando en él puede haber errores y símbolos borrados. La decodificación de símbolos borrados no produce un incremento sustancial de la fiabilidad en canales que distribuyen los errores uniformemente, pero sí en canales con errores a ráfagas (por ejemplo, canales radioeléctricos limitados en banda).

Un código de bloques $[n, k]$ es capaz de corregir t errores y b símbolos borrados si, para cualquier vector recibido, $\mathbf{r}$, con b o menos símbolos borrados, existe una única palabra del código, $\mathbf{v}$, a distancia Hamming menor o igual que t de $\mathbf{w}$, donde $\mathbf{w}$ es cualquier vector que coincide con el recibido en las $n - b$ posiciones de los símbolos no borrados. Dicho de otro modo, el vector $\mathbf{r}$ se puede corregir si existe una única palabra del código tal que un vector de error de peso no superior a t la convierte en un vector que coincide con el recibido en los símbolos no borrados.

TEOREMA 11.21. *Un código de bloques $[n, k]$ de distancia $d_{\mathcal{C}}$ puede corregir t errores y b símbolos borrados si $2t + b < d_{\mathcal{C}}$.*

DEMOSTRACIÓN. Sea $\mathbf{r}$ un vector con t errores y b símbolos borrados, y S el conjunto de vectores que coinciden con $\mathbf{r}$ en los $n - b$ símbolos no borrados. Supongamos que existen dos vectores del código, $\mathbf{x}_1$ y $\mathbf{x}_2$, tales que

$$d_H(\mathbf{x}_1, \mathbf{y}_1) \leq t; \quad d_H(\mathbf{x}_2, \mathbf{y}_2) \leq t$$

para ciertos $\mathbf{y}_1, \mathbf{y}_2 \in S$. Puesto que $\mathbf{y}_1$ e $\mathbf{y}_2$ difieren como máximo en b posiciones,

$$t + t + b \geq d_H(\mathbf{x}_1, \mathbf{y}_1) + d_H(\mathbf{x}_2, \mathbf{y}_2) + d_H(\mathbf{y}_1, \mathbf{y}_2).$$

Pero, en virtud de la desigualdad triangular,

$$2t + b \geq d_H(\mathbf{x}_1, \mathbf{y}_1) + d_H(\mathbf{x}_2, \mathbf{y}_1) \geq d_H(\mathbf{x}_1, \mathbf{x}_2)$$

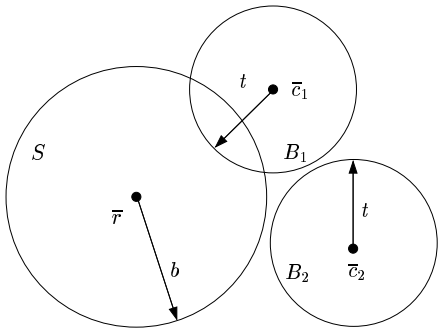


Figura 11.2. Interpretación geométrica de la decodificación con símbolos borrados. S es el conjunto de vectores de $\mathbb{F}_q^n$ que coinciden con el vector recibido, $\mathbf{r}$, en los símbolos no borrados; $\mathbf{c}_1$ y $\mathbf{c}_2$ son vectores del código, y B_1 y B_2 esferas de radio t.

y, entonces, $d_{\mathcal{C}} \leq 2t+b$. Esta conclusión es imposible, de forma que $\mathbf{x}_1 = \mathbf{x}_2$, y la combinación de t errores y b símbolos borrados es corregible. ▶

En particular, un código de distancia $d_{\mathcal{C}}$ corrige $d_{\mathcal{C}}-1$ símbolos borrados, el doble que símbolos con errores.

Los códigos BCH y, sobre todo, los RS permiten realizar la decodificación de símbolos borrados de manera muy eficiente. Por dos motivos:

- Porque existen para ellos algoritmos de decodificación intrínsecamente eficientes.
- En el caso de los códigos RS, porque son de máxima distancia; o dicho de otra forma, porque pueden corregir el mayor número posible de símbolos borrados y errores.

Bajo la suposición de que no se producen errores en los símbolos no borrados, en un código RS-$[n,k]$ es suficiente con detectar k símbolos para deducir (corregir) todos los borrados, ya que las k coordenadas de los símbolos sin error son linealmente independientes y determinan un único vector del código.

En un caso general, la decodificación de solamente los símbolos borrados equivale a la resolución de un sistema de k ecuaciones lineales y precisa, en consecuencia, $O(k^3)$ operaciones.

11.6.1. Decodificación de códigos binarios

En el caso binario, existe un método muy simple para determinar el vector del código emitido utilizando sólo decodificadores clásicos. Se trata de construir, a partir del vector recibido, dos nuevos vectores, sustituyendo en un caso los símbolos borrados por ceros y, en el otro, por unos. Seguidamente, se decodifican ambos vectores. De los dos vectores del código que se obtienen, se selecciona finalmente el más cercano en distancia Hamming al vector recibido. Si $2t+b<d_{\mathcal{C}}$, al menos en una de las dos sustituciones el número de errores en el vector que se construye es menor o igual que $t+b/2$, y ésta se habrá decodificado correctamente.

11.6.2. Decodificación de códigos RS y BCH no binarios

Se supondrá que el vector recibido a la salida del canal consta de $s \leq t$ errores y b símbolos borrados. Los errores ocurren en las posiciones $i_1, \ldots, i_s$ y tienen magnitudes $e_1, \ldots, e_s$, y los símbolos borrados afectan a las posiciones $j_1, \ldots, j_b$.

Como antes, sea α una raíz primitiva n-ésima de la unidad y sean $X_1 = \alpha^{i_1}, \ldots, X_s = \alpha^{i_s}$ los localizadores de error. De manera análoga, se representarán ahora por $Y_1 = \alpha^{j_1}, Y_2 = \alpha^{j_2}, \ldots, Y_b = \alpha^{j_b}$ los localizadores de los símbolos borrados, que, a diferencia de los localizadores de error, sí son conocidos.

Fijemos arbitrariamente el valor de los símbolos $f_1, \ldots, f_b$ en las posiciones que han sido borradas y calculemos el síndrome del vector recibido $r(z)$ mediante

$$S_j = r(\alpha^j) = \sum_{i=1}^{s} e_i X_i^j + \sum_{i=1}^{b} f_i Y_i^j \quad j = 1, \ldots, 2t$$

en donde $X_l = \alpha^{i_l}$ e $Y_m = \alpha^{j_m}$. Definamos ahora el polinomio localizador de errores

$$L(z) = \prod_{i=1}^{s} (1 - X_i z)$$

y el *polinomio localizador de símbolos borrados*

$$\beta(z) = \prod_{i=1}^{b} (1 - Y_i z), \tag{11.18}$$

éste último conocido. A partir de ellos, se puede definir el siguiente *polinomio*

evaluador de errores y de símbolos borrados

$$E(z) = L(z)\beta(z)\left(1 + \sum_{i=1}^{s} \frac{e_i z X_i}{1 - X_i z} + \sum_{i=1}^{b} \frac{f_i z Y_i}{1 - Y_i z}\right)$$

que recibe tal nombre a causa de estas dos sencillas propiedades que revelan de qué manera se pueden recuperar las magnitudes de los símbolos erróneos y de los símbolos borrados:

$$E(X_j^{-1}) = \beta(X_j^{-1}) e_j \prod_{i \neq j} \left(1 - X_i X_j^{-1}\right), \quad j = 1, \ldots, s$$
$$E(Y_j^{-1}) = L(Y_j^{-1}) f_j \prod_{i \neq j} \left(1 - Y_i Y_j^{-1}\right), \quad j = 1, \ldots, s.$$

En estas condiciones, se cumple la siguiente versión de la identidad clave

$$\big(1 + S(z)\big) L(z)\beta(z) = E(z) \mod z^{2t+1}$$

cuya demostración es idéntica a la del teorema 11.13, y que puede asimismo ser resuelta tanto por el algoritmo de Berlekamp–Massey como por el algoritmo euclídeo para obtener $L(z)$.

En caso de aplicar el algoritmo de Berlekamp–Massey, el procedimiento sería así:

---

1. Calcular los síndromes S_j, $j = 1, \ldots, 2t$, evaluando el vector recibido en α^j.

2. Formar el polinomio evaluador de símbolos borrados (11.18).

3. Aplicar el algoritmo de Berlekamp–Massey para obtener $L(z)$. Emplear las condiciones iniciales $i = b$, $L^{(b)}(z) = T^{(b)}(z) = \beta(z)$ y $M_b = 0$, y calcular la recurrencia

$$\Delta_i = \sum_{j=0}^{i-1} L_j^{(i-1)} S_{i-j}$$
$$M_i = \delta(i - M_{i-1} - b) + (1 - \delta) M_{i-1}$$
$$\begin{pmatrix} L^{(i)}(z) \\ T^{(i)}(z) \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_i z \\ \Delta_i^{-1}\delta & (1-\delta)z \end{pmatrix} \begin{pmatrix} L^{(i-1)}(z) \\ T^{(i-1)}(z) \end{pmatrix}$$

   con $\delta = 1$ si $\Delta_i \neq 0$ y $2M_{i-1} \leq i - 1 + b$, y $\delta = 0$ en otro caso. El polinomio localizador de errores es $L(x) = L^{(2t)}(z)/\beta(z)$.

4. Hallar las raíces de $L(z)$ con el método de búsqueda de Chien.

5. Determinar los valores de las magnitudes de error y los símbolos borrados con las ecuaciones

$$\begin{aligned} e_i &= \frac{E(X_i^{-1})}{\beta(X_i^{-1})\prod_{j\neq i}(1-X_jX_i^{-1})} = -X_i\frac{E(X_i^{-1})}{\beta(X_i^{-1})L'(X_i^{-1})} \\ &= -X_i\frac{E(X_i^{-1})}{\Psi'(X_i^{-1})}, \quad i=1,\ldots,s \\ f_i &= \frac{E(Y_i^{-1})}{L(X_i^{-1})\prod_{j\neq i}(1-Y_jY_i^{-1})} = -Y_i\frac{E(Y_i^{-1})}{L(Y_i^{-1})\beta'(Y_i^{-1})} \\ &= -X_i\frac{E(Y_i^{-1})}{\Psi'(Y_i^{-1})}, \quad i=1,\ldots,b \end{aligned}$$

en donde $\Psi(z)=L(z)\beta(z)$ y $\Psi'(z)$ es la derivada formal de $\Psi(z)$.

6. Corregir los errores y los símbolos borrados.

Es obvio que la elección más cómoda para simplificar todos estos cálculos es tomar $f_1 = f_2 = \cdots = f_b = 0$, esto es, sustituir al principio de la decodificación todos los símbolos borrados por ceros.

Ejemplo 11.12. Supongamos que se utiliza el código RS-$[7,3]$ de polinomio generador

$$g(z) = z^4 + \alpha^3 z^3 + z^2 + \alpha z + \alpha^3$$

y que se desea decodificar el vector $r(z) = \alpha^3 z^2 + bz^3 + z^4 + z^6$, en donde b señala un símbolo borrado que ocurre en el localizador $Y_1 = \alpha^3$. Si fijamos $b=0$, entonces el polinomio síndrome será

$$S(z) = \sum_{i=1}^{4}\left(\alpha^3(\alpha^i)^2 + (\alpha^i)^4 + (\alpha^i)^6\right) z^i = \alpha^2 z + \alpha^2 z^2 + \alpha^6 z^3 + z^4.$$

El polinomio localizador de símbolos borrados es $\beta(z) = 1 + \alpha^3 z$.

Ahora, estableciendo inicialmente $L^{(1)}(z) = T^{(1)}(z) = 1 + \alpha^3 z$, $M_1 = 0$, el algoritmo de Berlekamp–Massey produce

i	$L^{(i)}(z)$	$T^{(i)}(z)$	Δ_i	δ	M_i
1	$1+\alpha^3 z$	$1+\alpha^3 z$	—	—	0
2	$1+\alpha^6 z^2$	$\alpha^4 + z$	α^3	1	1
3	$1+\alpha^2 z+\alpha z^2$	$\alpha^2+\alpha z^2$	α^5	1	1
4	$1+\alpha^2 z+\alpha z^2$	—	0	—	—

de manera que $L(z)\beta(z) = 1+\alpha^2 z+\alpha z^2$, y por tanto se deduce que $L(z) = 1+\alpha^5 z$ es el polinomio localizador. Con la ecuación clave $E(z) = (1+S(z))L(z)\beta(z)$ se llega a que el polinomio evaluador de errores y símbolos borrados vale $E(z) = 1$. Por último, teniendo en cuenta que $L'(z) = \alpha^5$ y $\beta'(z) = \alpha^3$, hallamos las magnitudes de error con

$$e_1 = -\alpha^5 \frac{1}{\beta(\alpha^2)\alpha^5} = \alpha^3$$

$$b = -\alpha^3 \frac{1}{L(\alpha^4)\alpha^3} = \alpha$$

y se corrige en favor del vector recibido

$$\begin{aligned}\left(\alpha^3 z^2 + bz^3 + z^4 + z^6\right)\big|_{b=\alpha} + \alpha^3 z^5 &= \alpha^3 z^2 + \alpha z^3 + z^4 + \alpha^3 z^5 + z^6 \\ &= z^2 g(z).\end{aligned}$$

■

11.6.3. Decodificación de códigos RS acortados o perforados

Al inicio del capítulo vimos que las operaciones de perforación y recorte de un código RS preservaban la propiedad de máxima distancia en el código resultante, si bien, en general, éste no es cíclico. Ahora bien, si interpretamos como símbolos borrados las coordenadas que se han suprimido durante la punción o el recorte, entonces resulta posible decodificar directamente estos códigos modificados con los mismos procedimientos de decodificación de símbolos borrados.

Pues, si $\mathcal{C}^*$ es un código RS perforado de longitud n y dimensión k, y por tanto con distancia $n-k+1$, entonces el código matriz $\mathcal{C}$ del que procede es uno RS de longitud $q-1$, misma dimensión k y distancia $q-k$. La diferencia entre las distancias de $\mathcal{C}$ y $\mathcal{C}^*$ es de $q-n-1$, igual al número de coordenadas suprimidas. Si se añaden a un vector de $\mathcal{C}^*$ precisamente $q-n-1$ símbolos borrados, y el vector resultante (que tiene longitud $q-1$) se considera como un posible vector de $\mathcal{C}$, ocurre que la distancia de $\mathcal{C}$ es suficiente para decodificar con las inserciones, ya que se podrán corregir e errores si

$$2e + (q-n-1) < q-k,$$

es decir, si $2e < n-k+1$, que es precisamente la capacidad de corrección de errores de $\mathcal{C}^*$.

En cuanto a los códigos acortados, cabe emplear un argumento similar. Un código acortado se obtiene borrando símbolos de información nulos en todas las palabras del código. Ahora bien, como cualesquiera k coordenadas de $\mathcal{C}$ sirven como símbolos de información, la operación de recortar $q-n-1$ de ellas es, en realidad, equivalente a la supresión de esas $q-n-1$ coordenadas en un subcódigo de $\mathcal{C}$.

11.7. Códigos RS generalizados*

La definición dada en el apartado 11.1 se puede extender para dar lugar a los códigos RS generalizados. Sea $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{n-1})$ un vector de n elementos *distintos* de $\mathbb{F}_q$, con $n \leq q$, y sea $\gamma = (\gamma_0, \gamma_1, \ldots, \gamma_{n-1})$ una n-tupla de elementos *no nulos* de $\mathbb{F}_q$.

DEFINICIÓN 11.4 (CÓDIGO RS GENERALIZADO). *El código RS generalizado de dimensión k y longitud n es el conjunto de puntos*

$$\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma) = \left\{ \big(\gamma_0 P(\alpha_0), \gamma_1 P(\alpha_1), \ldots, \gamma_{n-1} P(\alpha_{n-1})\big) : P(x) \in \mathcal{F}_k[x] \right\}$$

siendo $\mathcal{F}_k[x]$ el conjunto de los polinomios de grado menor que k con coeficientes en $\mathbb{F}_q$.

Puesto que $P(x) \in \mathcal{F}_k[x]$ sólo puede tener, como máximo, $k-1$ raíces en $\mathbb{F}_q$, cualquier palabra de $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ tendrá al menos $n-k+1$ elementos no nulos. Por tanto, en virtud de la cota de Singleton, la distancia de $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ es $n-k+1$ y los códigos RS generalizados son códigos MDS.

Obviamente, si γ y ζ son dos vectores de $(\mathbb{F}_q^*)^n$ proporcionales, los códigos $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ y $\mathrm{RSG}_k(\boldsymbol{\alpha}, \zeta)$ son equivalentes. La familia de códigos RS son sencillamente $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ cuando $\gamma_i = 1$ y $\alpha_i = \alpha^i$, para $i = 0, \ldots, n-1$, con α una raíz primitiva n-ésima de la unidad y $n = q-1$.

Demostraremos seguidamente que el dual de un código RS generalizado es otro código RS generalizado, definido sobre el mismo conjunto de puntos $\boldsymbol{\alpha}$ pero con un vector de escala γ diferente.

TEOREMA 11.22. *Existe un vector $(\eta_0, \eta_1, \ldots, \eta_{n-1}) \in \mathbb{F}_q^n$ de elementos no nulos tal que, para cualquier $0 \leq k \leq n-1$,*

$$\mathrm{RSG}_k^{\perp}(\boldsymbol{\alpha}, \gamma) = \mathrm{RSG}_{n-k}(\boldsymbol{\alpha}, \eta).$$

DEMOSTRACIÓN. El caso $k = n-1$ es el más sencillo. Como $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ es separable y de máxima distancia, su código dual es $[n, 1, n]$ y poseerá una base compuesta por un único vector $(\eta_0, \eta_1, \ldots, \eta_{n-1})$ con todas sus componentes distintas de cero. Si h es un polinomio de grado menor que $k = n-1$, entonces

$$\sum_{i=0}^{n-1} \eta_i \gamma_i h(\alpha_i) = 0$$

ya que $(\gamma_0 h(\alpha_0), \gamma_1 h(\alpha_1), \ldots, \gamma_{n-1} h(\alpha_{n-1})) \in \mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$. Y en particular,

para cualesquiera $f(x) \in \mathcal{F}_k[x]$, $g(x) \in \mathcal{F}_{n-k}[x]$, $0 \leq k \leq n-1$, se tiene que

$$\sum_{i=0}^{n-1} \eta_i g(\alpha_i)\gamma_i f(\alpha_i) = 0.$$

En consecuencia, $\mathrm{RSG}_k^{\perp}(\boldsymbol{\alpha}, \gamma) \subseteq \mathrm{RSG}_{n-k}(\boldsymbol{\alpha}, \eta)$. Y como $\mathrm{RSG}_k^{\perp}(\boldsymbol{\alpha}, \gamma)$ tiene dimensión $n-k$, el teorema queda probado. ▶

No obstante, el vector de escala que define el código dual se puede calcular explícitamente haciendo uso directo del siguiente resultado, que requiere de un lema previo.

Lema 11.23. *Sea $\mathcal{K}_n$ el cuerpo de las funciones racionales en las variables $X_0, \ldots, X_{n-1}$. Si $P(x)$ es un polinomio de $\mathbb{F}_q[x]$ de grado menor que m y $m \geq 2$, entonces*

$$\sum_{i=0}^{n-1} \frac{P(X_i)}{\prod_{j \neq i}(X_j - X_i)} = 0.$$

Demostración. Sea Z una variable distinta de cualquiera de las X_i, y definamos la función

$$f(Z) = \frac{ZP(Z)}{X_0 X_1 \cdots X_{n-1}}.$$

Evaluando la función en el punto X_i

$$f(X_i) = \frac{P(X_i)}{\prod_{j \neq i} X_j} \qquad 0 \leq i \leq n-1.$$

Por la fórmula de interpolación de Lagrange

$$f(Z) = \sum_{i=0}^{n-1} f(X_i) \prod_{j \neq i} \frac{Z - X_i}{X_j - X_i}.$$

Y evaluando esta expresión en $Z = 0$ resulta la condición

$$\sum_{i=0}^{n-1} \frac{P(X_i)}{\prod_{j \neq i}(X_j - X_i)} = 0. \qquad ▶$$

Teorema 11.24. *Para $n \geq 2$, $\mathrm{RSG}_k^{\perp}(\boldsymbol{\alpha}, \gamma) = \mathrm{RSG}_{n-k}(\boldsymbol{\alpha}, \eta)$ con*

$$\eta_i = \frac{1}{\gamma_i \prod_{j \neq i}(\gamma_j - \gamma_i)}, \qquad 0 \leq i \leq n-1.$$

DEMOSTRACIÓN. La condición de ortogonalidad vista en la demostración del teorema 11.22

$$\sum_{i=0}^{n-1} \gamma_i \eta_i h(\alpha_i) = 0$$

en donde $h(x)$ es un polinomio arbitrario de grado menor que $n-1$, es justamente la del lema anterior sin más que identificar $X_j = \alpha_j$. ▶

Conforme a la definición, una matriz generadora de $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ es

$$G = \begin{pmatrix} \gamma_0\alpha_0^{k-1} & \gamma_1\alpha_1^{k-1} & \dots & \gamma_{n-1}\alpha_{n-1}^{k-1} \\ \gamma_0\alpha_0^{k-2} & \gamma_1\alpha_1^{k-2} & \dots & \gamma_{n-1}\alpha_{n-1}^{k-2} \\ \dots & \dots & \dots & \dots \\ \gamma_0\alpha_0 & \gamma_1\alpha_1 & \dots & \gamma_{n-1}\alpha_{n-1} \\ \gamma_0 & \gamma_1 & \dots & \gamma_{n-1} \end{pmatrix},$$

mientras que, por el teorema 11.22, una matriz de comprobación de paridad es cualquier generadora de $\mathrm{RSG}_{n-k}(\boldsymbol{\alpha}, \eta)$

$$H = \begin{pmatrix} \eta_0\alpha_0^{n-k-1} & \eta_1\alpha_1^{n-k-1} & \dots & \eta_{n-1}\alpha_{n-1}^{n-k-1} \\ \eta_0\alpha_0^{n-k-2} & \eta_1\alpha_1^{n-k-2} & \dots & \eta_{n-1}\alpha_{n-1}^{n-k-2} \\ \dots & \dots & \dots & \dots \\ \eta_0\alpha_0 & \eta_1\alpha_1 & \dots & \eta_{n-1}\alpha_{n-1} \\ \eta_0 & \eta_1 & \dots & \eta_{n-1} \end{pmatrix}.$$

La matriz G también se puede escribir como

$$G = \begin{pmatrix} \alpha_0^{k-1} & \dots & \alpha_{n-1}^{k-1} \\ \alpha_0^{k-2} & \dots & \alpha_{n-1}^{k-2} \\ \dots & \dots & \dots \\ \alpha_0 & \dots & \alpha_{n-1} \\ 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} \gamma_0 & & & \\ & \gamma_1 & & \\ & & \ddots & \\ & & & \gamma_{n-1} \end{pmatrix}$$

lo que revela explícitamente que un código $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ es monomialmente[6] equivalente a un código RS perforado. Lo mismo es válido para el código dual:

$$H = \begin{pmatrix} \alpha_0^{n-k-1} & \dots & \alpha_{n-1}^{n-k-1} \\ \alpha_0^{n-k-2} & \dots & \alpha_{n-1}^{n-k-2} \\ \dots & \dots & \dots \\ \alpha_0 & \dots & \alpha_{n-1} \\ 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} \eta_0 & & & \\ & \eta_1 & & \\ & & \ddots & \\ & & & \eta_{n-1} \end{pmatrix}.$$

[6] Dos códigos lineales son monomialmente equivalentes cuando $G' = G \cdot D$ es una matriz generadora de uno de ellos si G lo es del otro, siendo D una matriz diagonal con escalares no nulos.

Sea ahora $\mathcal{C}$ un código RS-$[q-1,k]$, y sea $f(x)$ un polinomio arbitrario con coeficientes en el cuerpo $\mathbb{F}_q$. Por evaluación directa, es fácil comprobar que

$$\sum_{\beta\in\mathbb{F}_q} f(\beta) = 0$$

de modo que el código extendido

$$\widehat{\mathcal{C}} = \left\{\big(f(1), f(\alpha), f(\alpha^2), \ldots, f(\alpha^{q-2}), f(0)\big) : f \in \mathcal{F}_k[x]\right\}$$

es un código RSG con longitud $n = q$, dimensión k, vector de escala $\gamma_i = 1$, $\alpha_i = \alpha^i$, para $i = 0, \ldots, n-2$, y $\alpha_{n-1} = 0$. Por lo tanto, $\widehat{\mathcal{C}}$ es un código MDS, con distancia $n-k+2$ y matriz generadora

$$\widehat{G} = \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 & 1 \\ 1 & \alpha & \alpha^2 & \ldots & \alpha^{n-1} & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & \alpha^{k-1} & \alpha^{2(k-1)} & \ldots & \alpha^{(k-1)(n-1)} & 0 \end{pmatrix}.$$

Utilizando un método similar, un código $\mathrm{RSG}_k(\boldsymbol{\alpha}, \gamma)$ de longitud n puede ser extendido a un código $\widetilde{\mathcal{C}}$ añadiendo a G una nueva columna. La matriz generadora de $\widetilde{\mathcal{C}}$ es

$$\widetilde{G} = \begin{pmatrix} G & \mathbf{u}^T \end{pmatrix}$$

en donde $\mathbf{u} = (a, 0, 0, \ldots, 0)$ para algún $a \in \mathbb{F}_q$ no nulo. Si se elige $b \in \mathbb{F}_q$ no nulo tal que

$$ab + \sum_{i=0}^{n-1} \gamma_i \eta_i \alpha_i^{n-1} = 0,$$

cosa que siempre es posible pues $a \neq 0$, entonces una matriz de comprobación de paridad del código $\widetilde{\mathcal{C}}$ es

$$\widetilde{H} = \begin{pmatrix} \eta_0\alpha_0^{n-k} & \eta_1\alpha_1^{n-k} & \ldots & \eta_{n-1}\alpha_{n-1}^{n-k} & b \\ \eta_0\alpha_0^{n-k-1} & \eta_1\alpha_1^{n-k-1} & \ldots & \eta_{n-1}\alpha_{n-1}^{n-k-1} & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \eta_0\alpha_0 & \eta_1\alpha_1 & \ldots & \eta_{n-1}\alpha_{n-1} & 0 \\ \eta_0 & \eta_1 & \ldots & \eta_{n-1} & 0 \end{pmatrix}.$$

Cualquier conjunto de $n-k+1$ columnas de esta matriz es linealmente independiente. Si la última columna no está entre las elegidas, entonces la submatriz formada por las $n-k+1$ columnas es de Vandermonde, ya que $\alpha_0, \ldots, \alpha_{n-1}$ son todos distintos. Y si la última columna es una de las elegidas, entonces el determinante de la submatriz será $\pm b$ veces el determinante de una matriz de Vandermonde, producto que será distinto de cero al ser $b \neq 0$. Por lo tanto, $\widetilde{\mathcal{C}}$ tiene distancia $n-k+2$ y es también un código MDS.

Para la decodificación de los códigos RS generalizados pueden seguirse dos estrategias diferentes. Si los vemos simplemente como transformaciones monomiales de un código RS perforado, entonces basta con aplicar cualquier algoritmo de decodificación de un código RS al vector

$$(\gamma_0^{-1} y_0, \ldots, \gamma_{n-1}^{-1} y_{n-1}),$$

siendo $(y_0, \ldots, y_{n-1})$ el vector recibido a la salida del canal.

Pero cabe un planteamiento alternativo más directo de la decodificación. Dado un vector recibido $\mathbf{y} = (y_0, \ldots, y_{n-1})$ con t o menos errores ($t = (n-k)/2$), su vector síndrome será

$$\mathbf{s} = \mathbf{y} \cdot H^T = \mathbf{y} \cdot \begin{pmatrix} \eta_0 & & & \\ & \eta_1 & & \\ & & \ddots & \\ & & & \eta_{n-1} \end{pmatrix} \cdot \begin{pmatrix} 1 & \ldots & 1 \\ \alpha_0 & \ldots & \alpha_{n-1} \\ \ldots & \ldots & \ldots \\ \alpha_0^{n-k-2} & \ldots & \alpha_{n-1}^{n-k-2} \\ \alpha_0^{n-k-1} & \ldots & \alpha_{n-1}^{n-k-1} \end{pmatrix}^T .$$

Es decir,

$$s_j = \sum_{i=0}^{n-1} y_i \eta_i \alpha_i^j, \quad j = 0, \ldots, 2t-1.$$

Si los errores ocurren en las posiciones $i_1, \ldots, i_t$, y tienen magnitudes $Y_1 = e_{i_1}, \ldots, Y_t = e_{i_t}$, entonces las componentes del síndrome serán

$$s_j = \sum_{k=1}^{t} \eta_{i_k} e_{i_k} \alpha_{i_k}^j = \sum_{k=1}^{t} \eta_{i_k} Y_k X_k^j,$$

en donde $X_k = \alpha_{i_k}$ son los localizadores de errores. Si ahora definimos los polinomios

$$S(z) = \sum_{k=0}^{2t-1} s_k z^k \quad \text{(síndrome)}$$

$$L(z) = \prod_{k=1}^{t} (1 - X_k z) \quad \text{(localizador)}$$

$$E(z) = \sum_{k=1}^{t} \eta_k Y_k \prod_{\substack{j=1 \\ j \neq k}}^{k} (1 - X_k z) \quad \text{(evaluador de errores)}$$

se verifica la siguiente versión de la identidad clave:

$$\frac{E(z)}{L(z)} = S(z) \mod z^{2t}.$$

La identidad clave puede ser resuelta, como se sabe, por el algoritmo euclídeo (apartado 11.4.3), y las magnitudes de error se deducen con el algoritmo de Forney, por ejemplo.

Ejemplo 11.13. Si se toma como vector de escala $\gamma_i = \alpha^{n-1-i}$ y $\alpha_i = \alpha^i$, siendo α una raíz primitiva séptima de la unidad, se obtiene la siguiente matriz generadora de un código RS generalizado, $\mathrm{RSG}_4(\boldsymbol{\alpha}, \gamma)$, en $\mathbb{F}_8$

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

Para obtener $\mathrm{RSG}_j^{\perp}(\boldsymbol{\alpha}, \gamma)$, $j \leq 4$, será necesario, en primer lugar, hallar un vector ortogonal a G. La resolución (con la regla de Cramer, por ejemplo) del sistema $G\eta^T = \mathbf{0}^T$ produce como resultado el vector

$$\eta = (\alpha^2, \alpha^2, \alpha^3, \alpha, 1)$$

en donde la normalización de la última componente es arbitraria. Por lo tanto,

$$\mathrm{RSG}_j^{\perp}(\boldsymbol{\alpha}, \gamma) = \mathrm{RSG}_{5-j}(\boldsymbol{\alpha}, \eta).$$

Las componentes del vector η también se pueden calcular expresamente a partir de γ con el teorema 11.24 sin necesidad de resolver ningún sistema lineal. ■

Notas bibliográficas

La descripción original de los códigos Reed–Solomon apareció en [56], si bien se conocían desde tiempo antes en otro contexto. La familia se puede describir ya como códigos polinómicos, ya como códigos geométricos [53, cap. 16]. El primer algoritmo de decodificación apareció en [28]. Berlekamp [3] continúa siendo la referencia fundamental para entender la decodificación algebraica. La descripción del algoritmo de Berlekamp–Massey que figura como apéndice a este capítulo sigue las líneas de Blahut en [53, cap. 16]. El algoritmo de Sudan, una extensión del algoritmo de Berlekamp–Massey ideada para la decodificación de códigos bicíclicos y de códigos cíclicos definidos sobre curvas, también se explica en [53, cap. 16]. El algoritmo de Forney para la obtención de las magnitudes de error es, en realidad, la bien conocida fórmula de interpolación polinómica de Lagrange. El algoritmo euclídeo fue propuesto por Sugiyama en [66]. Por último, el algoritmo de Sudan-Guruswami explota técnicas de interpolación polinómica que se

retrotraen hasta Newton y puede modificarse sin excesiva complicación para decodificar códigos definidos a partir de curvas algebraicas [30].

Wicker, primero en la monografía [74] y luego en [75], y Reed [57] ofrecen abundante información sobre la aplicación práctica de estos códigos. Las dos obras pueden resultar interesantes además porque muestran arquitecturas *hardware* de decodificadores RS.

Por otro lado, los códigos Reed–Solomon son uno de los puntos de partida para llegar a los códigos algebraico–geométricos, una clase de códigos sofisticados que, si bien no posee aún gran relevancia práctica, sí tiene un interés teórico indudable. Baste mencionar que algunos códigos de esta familia mejoran la cota inferior de Gilbert–Varshamov. Para más información, puede consultarse [53].

11.A. Los códigos cíclicos y la transformada de Fourier

La transformada de Fourier puede definirse en cualquier cuerpo abeliano, incluidos los cuerpos de Galois. Las propiedades de la transformada son una consecuencia de la propia estructura del cuerpo, no de la naturaleza particular de los elementos del mismo, por lo que se verifican con independencia de cuál sea el número de sus elementos.

La definición de la transformada de Fourier en cualquier cuerpo solamente precisa de la existencia de un elemento de orden n.

Definición 11.5. *La transformada de Fourier de un vector* $\mathbf{v}$ *de longitud* n *con elementos del cuerpo* $\mathbb{F}_q$ *es el vector*

$$V_j = \sum_{i=0}^{n-1} \omega^{ij} v_i, \qquad j = 0, \ldots, n-1$$

donde ω *es un elemento de orden* n *de* $\mathbb{F}_{q^m}$.

Si $m = 1$, ω es un elemento de $\mathbb{F}_q$; si $m > 1$, ω es un elemento de la extensión $\mathbb{F}_{q^m}$ de $\mathbb{F}_q$. Cuando $n = q^m - 1$, ω es un elemento primitivo de $\mathbb{F}_{q^m}$.

El vector $\mathbf{V} = (V_0, \ldots, V_{n-1})$ formado por los coeficientes de la transformada de Fourier es el *espectro* de $\mathbf{v}$. La notación habitual para representar la transformación biunívoca de Fourier es $\mathbf{v} \longleftrightarrow \mathbf{V}$.

Resulta sencillo verificar que se preservan todas las propiedades usuales de la transformada de Fourier sobre el campo complejo:

a) Fórmula de inversión:

$$v_i = \frac{1}{n} \sum_{i=0}^{n-1} \omega^{-ik} V_k, \qquad i = 0, \ldots, n-1$$

b) Linealidad: $\lambda\mathbf{u} + \mu\mathbf{v} \longleftrightarrow \lambda\mathbf{U} + \mu\mathbf{V}$ para $\lambda, \mu \in \mathbb{F}_q$.

c) Modulación: $(v_i\omega^{il}) \longleftrightarrow (V_{(k+l)})$. $(V_{(k+l)})$ es la rotación cíclica del espectro $\mathbf{V}$ en l posiciones. Los paréntesis en el subíndice indican la reducción módulo n.

d) Traslación: $(v_{(i-l)}) \longleftrightarrow (V_k\omega^{lk})$.

e) Convolución cíclica:

$$\Big(e_i = \sum_{l=0}^{n-1} f_{(i-l)} g_l\Big) \longleftrightarrow (E_k = F_k G_k)$$

$$(f_i g_i) \longleftrightarrow n^{-1} \sum_{k=0}^{n-1} G_k F_{(i-k)}.$$

f) El polinomio $v(x) = \sum_{i=0}^{n-1} v_i x^i$ tiene un cero en ω^k si y solamente si $V_k = 0$.

g) En el cuerpo $\mathbb{F}_{q^m}$, la transformada de Fourier de $\mathbf{v}$ tiene componentes en el subcuerpo $\mathbb{F}_q$ si y sólo si $v_j^q = v_{(jq)}$.

h) Diezmado: si $n = n_1 n_2$

$$(v_0, v_{n_2}, v_{2n_2}, \ldots, v_{(n_1-1)n_2}) \longleftrightarrow \Big(\frac{1}{n_2} \sum_{j=0}^{n_2-1} V_{i+n_1 j}\Big)_{i=0,\ldots,n_1-1}$$

i) Permutación cíclica: si $(a, n) = 1$, entonces

$$\big(v_{(ai)}\big) \longleftrightarrow \big(V_{(bk)}\big)$$

con $ab = 1 \mod n$.

La relación entre ceros espectrales y raíces de un polinomio es especialmente útil en el caso de los códigos cíclicos. Con ella, las palabras de un código BCH de distancia de diseño δ, definido sobre $\mathbb{F}_q$ y con ceros en $\{\omega^b, \omega^{b+1}, \ldots, \omega^{b+\delta-2}\}$ son el conjunto de secuencias $\mathbf{v}$ de $\mathbb{F}_q^n$ cuya transformada de Fourier cumple $V_{(b)} = V_{(b+1)} = \cdots = V_{(b+\delta-2)} = 0$ y $V_{(jq)} = V_j^q$. En general, en un código BCH con distancia de diseño δ, $\delta - 1$ coeficientes (cíclicamente) consecutivos del espectro de cualquier palabra del código son nulos. Consecuentemente, el cálculo del síndrome con un código BCH es el cálculo de $n - k$ coeficientes espectrales del vector recibido.

11.B. El algoritmo de Berlekamp–Massey

Como ya se ha comentado en el capítulo, el algoritmo de Berlekamp–Massey es un procedimiento computacional eficiente para la decodificación de códigos cíclicos y de códigos algebraicos definidos sobre curvas. Pertenece a la clase de algoritmos de decodificación por localización, aquéllos en los que se calcula un *polinomio localizador* $\Lambda(x)$, basado en la estructura algebraica del código, que permite identificar los errores de transmisión. Este apéndice ofrece una explicación más rigurosa de los fundamentos del algoritmo.

Supóngase un código BCH de longitud n definido en el cuerpo $\mathbb{F}_q$, con ceros $\{1, \alpha, \alpha^2, \ldots, \alpha^{\delta-2}\}$, $\delta = 2t+1$, y α una raíz primitiva n-ésima de la unidad. Sea $\mathbf{e}$ un vector de error de peso $s \leq t$

$$\mathbf{e} = e_{i_1} x^{i_1} + \cdots + e_{i_s} x^{i_s}.$$

El problema de la decodificación consiste en resolver el sistema de ecuaciones no lineal

$$S_j = \sum_{i=0}^{n-1} \alpha^{ij} e_i, \quad j = 0, \ldots, 2t-1$$

entre las $2t$ componentes conocidas del síndrome (que son las $2t$ primeras componentes de la transformada de Fourier de $\mathbf{e}$) y el vector de error $\mathbf{e}$, para un $\mathbf{e}$ de la menor longitud posible y peso no mayor que t.

En el procedimiento de decodificación por localización, este problema no lineal se descompone en dos pasos. Primero se establece una relación *lineal* entre los coeficientes del síndrome y los del polinomio localizador $\Lambda(x)$. Luego, se resuelve una relación no lineal sencilla (una recurrencia lineal) entre los coeficientes de $\Lambda(x)$ y los de $\mathbf{E}$, la transformada de Fourier de $\mathbf{e}$, para hallar el resto de valores desconocidos de $\mathbf{E}$. Por último, se invierte el espectro $\mathbf{E}$ para recuperar el vector de error.

11.B.1. Polinomios localizadores

En el anillo de polinomios $F_q[x]/x^n - 1$ se llama *polinomio localizador* de $P(x)$ a cualquier polinomio $\Lambda(x)$ que cumpla

$$\Lambda(x)P(x) = 0 \mod x^n - 1.$$

El conjunto de polinomios localizadores de uno dado, $P(x)$, es un ideal de $F_q[x]/x^n-1$. Como $F_q[x]/x^n-1$ es un anillo de ideales principales, el ideal de polinomios localizadores está generado por el localizador de grado mínimo.

Si α es un elemento de orden n, entonces, para $i = 0, 1, \ldots, n-1$,

$$\Lambda(\alpha^{-i})P(\alpha^{-i}) = 0$$

y $\Lambda(x)$ permite identificar, a través de sus ceros, aquellas raíces n-ésimas de la unidad que no son ceros de $P(x)$.

En el caso de los códigos BCH, y supuesto el vector de error $\mathbf{e}$, el polinomio

$$\Lambda(x) = \prod_{i=1}^{s}(1 - x\alpha^{i_l})$$

es el polinomio localizador de grado mínimo de $E(x) = \sum_{i=0}^{n-1} E_i x^i$ en el anillo $F_{q^m}[x]/x^n - 1$, siendo $E(x)$ el polinomio cuyos coeficientes son el espectro de $\mathbf{e}$.[7] Para demostrarlo, vea que, por la definición, $\lambda_i = \Lambda(\alpha^{-i})/n = 0$ si y sólo si $e_i \neq 0$, y por tanto $e_i\lambda_i = 0$ para cualquier índice i. Además, los coeficientes de $\Lambda(x) = \sum_{i=0}^{s} \Lambda_i x^i$ son la transformada de Fourier de los λ_i. Por la propiedad de convolución de la transformada de Fourier, la condición $e_i\lambda_i = 0$ equivale a

$$\Lambda(x)E(x) = 0 \mod x^n - 1.$$

Escrita componente a componente, esta relación se convierte en la recurrencia lineal

$$\sum_{k=0}^{s} \Lambda_k E_{(j-k)} = 0, \quad j = 0, 1, \ldots, n-1.$$

Los paréntesis en el subíndice indican reducción módulo n. Si se advierte que $\Lambda_0 = 1$, se puede escribir

$$E_j = -\sum_{k=1}^{s} \Lambda_k E_{(j-k)}, \quad j = 0, 1, \ldots, n-1.$$

De estas n ecuaciones, hay t en las que figuran las $2t$ componentes conocidas de $\mathbf{E}$ (los síndromes) y las t incógnitas de $\Lambda(x)$. Expresadas en forma matricial, estas t ecuaciones son:

$$\begin{pmatrix} E_{t-1} & E_{t-2} & \ldots & E_0 \\ E_t & E_{t-1} & \ldots & E_1 \\ \vdots & & & \vdots \\ E_{2t-2} & E_{2t-3} & \ldots & E_{t-1} \end{pmatrix} \begin{pmatrix} \Lambda_1 \\ \Lambda_2 \\ \vdots \\ \Lambda_t \end{pmatrix} = - \begin{pmatrix} E_t \\ E_{t+1} \\ \vdots \\ E_{2t-1} \end{pmatrix}. \tag{11.19}$$

Este sistema de ecuaciones posee al menos una solución pues, por hipótesis, el vector de error consta de t errores o menos y su polinomio localizador es una solución. Esto significa que si ha habido t errores exactamente, entonces tal solución es la única y el determinante de la matriz no es nulo. Si

[7] Dada una palabra $\mathbf{v}$ de un código cíclico, al polinomio $v(x) = \sum_{i=0}^{n-1} V_i x^i$ cuyos coeficientes son la transformada de Fourier de $\mathbf{v}$ se lo denomina también *polinomio de Mattson–Solomon* de $\mathbf{v}$.

el determinante de la matriz de coeficientes del sistema fuese nulo, y como máximo se han dado t errores, entonces existen múltiples soluciones. En tal caso, se debe reducir en uno el tamaño de la matriz y tratar de resolver el problema en la misma forma (teorema 11.12). Es decir, se intenta descubrir un vector de error con $t-1$ posiciones no nulas.

Por otra parte, las ecuaciones (11.19) son bien conocidas en álgebra: se trata de las ecuaciones generalizadas de Newton, que relacionan los coeficientes del polinomio $\Lambda(x)$ con las potencias de los elementos α^{i_l}.

El sistema de ecuaciones planteado se puede resolver con cualquier procedimiento conveniente (la matriz del sistema es una matriz de Toeplitz). Una vez conocido $\Lambda(x)$, el resto del espectro del vector de error se obtiene de

$$E_j = -\sum_{k=1}^{s} \Lambda_k E_{(j-k)}, \quad j = 0, 1, \ldots, n-1.$$

Realizando la transformada inversa de Fourier de $\mathbf{E}$, se obtienen las componentes de $\mathbf{e}$, que restadas del vector recibido dan los símbolos de la palabra del código transmitida.

11.B.2. Complejidad lineal de secuencias

Se define la *complejidad lineal* de la secuencia $\mathbf{v} = (v_0, v_1, \ldots, v_{r-1})$, con elementos en $\mathbb{F}_q$ y en donde $r-1$ no tiene por qué ser finito, como el menor valor de L para el que existe una recurrencia

$$v_i = -\sum_{j=1}^{L} \lambda_j v_{i-j}, \quad i = L, \ldots, r-1$$

que produce el resto de términos de la secuencia a partir de los L primeros. La complejidad lineal de $\mathbf{v}$ se representará por el signo $\mathcal{L}(\mathbf{v})$. Si la secuencia es de longitud finita, $\mathcal{L}(\mathbf{v})$ no será mayor que dicha longitud; si la secuencia es nula, su complejidad será cero; y para secuencias infinitas, si no existe una recurrencia como la descrita (por ejemplo, porque la sucesión no es lineal), será $\mathcal{L}(\mathbf{v}) = \infty$.

La complejidad lineal de una secuencia puede verse, por tanto, como la longitud mínima de un registro de desplazamiento con realimentación lineal que produce $\mathbf{v}$ cuando las condiciones iniciales son $(v_0, \ldots, v_{L-1})$. Los coeficientes λ_j son los de realimentación del registro. Si se define $\lambda_0 = 1$, con los L coeficientes λ_i se puede formar el *polinomio de conexión*

$$L(x) = \sum_{i=0}^{L} \lambda_i x^i.$$

Ahora bien, el polinomio $L(x)$ no define unívocamente la recurrencia, ya que puede ocurrir que $\lambda_L = 0$: en tal caso, la secuencia definida por $L(x)$ es simplemente una versión desplazada y con un término inicial más que una recurrencia definida por $(L(x), L-1)$. En general, sólo se sabe que el grado de $L(x)$ no es mayor que L. En lo sucesivo, una recurrencia dada por el polinomio de conexión $L(x)$ y complejidad lineal L se representará por $(L(x), L)$.

El siguiente teorema dice que dos recurrencias distintas generan la misma sucesión de elementos si generan un número suficiente de términos iguales.

TEOREMA 11.25 (DE CONCORDANCIA). *Si* $(L(x), L)$ *y* $(L'(x), L')$ *producen ambas la secuencia* $(v_0, v_1, \ldots, v_{r-1})$, *y* $r \geq L + L'$, *entonces*

$$-\sum_{k=1}^{L} \lambda_k v_{r-k} = -\sum_{k=1}^{L'} \lambda'_k v_{r-k}.$$

DEMOSTRACIÓN. Por hipótesis

$$v_i = -\sum_{j=1}^{L} \lambda_j v_{i-j}, \quad i = L, \ldots, r-1,$$

$$v_i = -\sum_{j=1}^{L'} \lambda'_j v_{i-j}, \quad i = L', \ldots, r-1.$$

Además, puesto que $r \geq L + L'$, se puede escribir

$$v_{r-k} = -\sum_{j=1}^{L'} \lambda'_j v_{r-k-j}, \quad k = 1, \ldots, L,$$

$$v_{r-k} = -\sum_{j=1}^{L} \lambda_j v_{r-k-j}, \quad k = 1, \ldots, L'$$

Por lo tanto,

$$\begin{aligned} -\sum_{k=1}^{L} \lambda_k v_{r-k} &= \sum_{k=1}^{L} \lambda_k \sum_{j=1}^{L'} \lambda'_j v_{r-k-j} \\ &= \sum_{j=1}^{L'} \lambda'_j \sum_{k=1}^{L} \lambda_k v_{r-k-j} \\ &= -\sum_{j=1}^{L'} \lambda'_j v_{r-j}. \end{aligned}$$

▶

Teorema 11.26 (Massey). *Si* $(L(x), L)$ *es una recurrencia que produce* $(v_0, v_1, \ldots, v_{r-2})$ *pero no* $(v_0, v_1, \ldots, v_{r-1})$*, entonces* $\mathcal{L}(\mathbf{v}) \geq r - L$*.*

Demostración. Suponga que existe una recurrencia lineal $(L'(x), L')$ que produce $(v_0, \ldots, v_{r-1})$ y con $L' < r - L$. Entonces, $(L(x), L)$ y $(L'(x), L')$ producen ambas $(v_0, \ldots, v_{r-2})$ y $L + L' \leq r - 1$. Pero, por el teorema anterior, esto implica que deben generar el mismo valor de la secuencia en la iteración $r - 1$, contra lo que se había supuesto. ▶

Massey probó además el siguiente corolario inmediato: si $(L(x), L)$ es la recurrencia de mínima longitud que genera $(v_0, \ldots, v_{r-2})$, entonces $\mathcal{L}(\mathbf{v}) \geq$ máx$\{L, r - L\}$.

El polinomio de conexión de la secuencia $\mathbf{E} = (E_0, \ldots, E_{n-1})$ no tiene por qué coincidir necesariamente con el polinomio localizador de $E(x) = \sum_{i=0}^{n-1} E_i x^i$. Para ser iguales, el polinomio de conexión tiene que producir la secuencia $\mathbf{E}$ repetida periódicamente, con periodo n, al igual que lo hace el polinomio localizador. Esta observación conduce al siguiente resultado.

Teorema 11.27. *Sea* $\mathbf{E} = (E_0, \ldots, E_{n-1})$ *una secuencia de longitud* n *y sea el polinomio asociado* $E(x) = \sum_{i=0}^{n-1} E_i x^i$*. El polinomio localizador de* $E(x)$ *y el polinomio de conexión de* $\mathbf{E}$ *coinciden si el grado del polinomio localizador no es mayor que* $n/2$*.*

Demostración. La complejidad lineal de la secuencia no es mayor que la de la secuencia repetida cíclicamente. Por tanto, si el grado del polinomio localizador no es mayor que $n/2$, las recurrencias que definen los polinomios localizador y de conexión son como máximo de longitud $n/2$ y concuerdan hasta el n-ésimo símbolo. Por el teorema de concordancia, deben coincidir en las iteraciones siguientes. ▶

11.B.3. El algoritmo de Berlekamp–Massey

El algoritmo de Berlekamp–Massey es un procedimiento para encontrar la recurrencia

$$E_j = -\sum_{k=1}^{v} \Lambda_k E_{j-k}, \quad j = v, \ldots, 2t-1$$

para el mínimo valor posible de v. Esta formulación del problema es más general que el problema de resolver el sistema (11.19), porque puede suceder

que aquel sistema de ecuaciones no posea solución. Sin embargo, el algoritmo de Berlekamp–Massey encuentra en este caso una recurrencia

$$E_j = -\sum_{k=1}^{v} \Lambda_k E_{j-k}, \quad j = v, \ldots, 2t-1$$

con $v > t$.

La idea básica del algoritmo es la de hallar, en cada etapa $r \geq 1$, la recurrencia $(\Lambda^{(r)}(x), L_r)$ de longitud mínima que produce los r primeros términos de la secuencia $\mathbf{E}$. Así pues,

$$E_j = -\sum_{k=1}^{L_r} \Lambda_k^{(r)} E_{j-k}, \quad j = L_r, \ldots, r-1.$$

La discrepancia o diferencia entre E_r y el término $r+1$ producido por $(\Lambda^{(r)}(x), L_r)$ es

$$\Delta_r = E_r - \left(-\sum_{k=1}^{L_r} \Lambda_k^{(r)} E_{j-k} \right) = \sum_{k=0}^{L_r} \Lambda_k^{(r)} E_{r-k}.$$

Si $\Delta_r = 0$, es obvio que la recurrencia en la siguiente etapa es la misma

$$(\Lambda^{(r+1)}(x), L_{r+1}) = (\Lambda^{(r)}(x), L_r).$$

Pero, en general, $\Delta_r \neq 0$.

Para plantear el algoritmo con una notación conveniente, obsérvese que la recurrencia se puede escribir de manera equivalente como

$$\Lambda^{(r)}(x) E(x) = p^{(r)}(x) + x^r g^{(r)}(x)$$

con $p^{(r)}(x)$ un polinomio de grado menor que L_r, de manera que en esta expresión los coeficientes de $\Lambda^{(r)}(x)E(x)$ sean nulos para los índices $j = L_r, \ldots, r-1$.

El teorema siguiente establece los cálculos de la iteración fundamental.

TEOREMA 11.28. *Si se cumple que la recurrencia* $(\Lambda^{(r)}(x), L_r)$ *produce la secuencia* $(E_0, \ldots, E_{r-2})$ *pero no* $(E_0, \ldots, E_{r-1})$*, y para algún* $m < r$ *la recurrencia* $(\Lambda^{(m)}(x), L_m)$ *produce* $(E_0, \ldots, E_{m-2})$ *pero no* $(E_0, \ldots, E_{m-1})$*, entonces*

$$(\Lambda(x), L) = (\Lambda^{(r)}(x) - \Delta_m^{-1} \Delta_r x^{r-m} \Lambda^{(m)}(x), \text{máx}\{L_r, L_m + r - m\})$$

produce $(E_0, \ldots, E_{r-1})$.

Demostración. Sean $E^{(r)}(x)$, $E^{(m)}(x)$ los polinomios correspondientes a las secuencias $(E_0, \ldots, E_{r-1})$, $(E_0, \ldots, E_{m-1})$. Por las hipótesis del teorema se puede escribir que

$$\begin{aligned}
&\Lambda^{(r)}(x)E^{(r)}(x) = p^{(r)}(x) + \Delta_r x^r + x^{r+1}g^{(r+1)}(x),\ \text{grado}(p^{(r)}(x)) < L_r,\\
&\Lambda^{(m)}(x)E^{(m)}(x) = p^{(m)}(x) + \Delta_m x^m + x^{m+1}g^{(m+1)}(x),\\
&\text{grado}(p^{(m)}(x)) < L_m < L_r.
\end{aligned}$$

Δ_r y Δ_m son distintos de cero, y los polinomios $g^{(r+1)}(x)$, $g^{(m+1)}(x)$ no presentan interés alguno. Si se multiplica $\Lambda^{(m)}(x)$ por $E^{(r)}(x)$, aparecen nuevos monomios de grado mayor que m, de modo que simplemente se tiene

$$\Lambda^{(m)}(x)E^{(r)}(x) = p^{(m)}(x) + \Delta_m x^m + x^{m+1}g^{(m+1)}(x),$$

con $\text{grado}(p^{(m)}(x)) < L_m$ y para un $g^{(m+1)}(x)$ diferente.

Luego, operando miembro a miembro

$$\left[\Lambda^{(r)}(x) - \frac{\Delta_r}{\Delta_m}x^{r-m}\Lambda^{(m)}(x)\right]E^{(r)}(x) = \left[p^{(r)}(x) - \frac{\Delta_r}{\Delta_m}x^{r-m}p^{(m)}(x)\right] + \left[\Delta_r - \frac{\Delta_r}{\Delta_m}\Delta_m\right]x^r + x^{r+1}[\ldots].$$

Pero esta ecuación tiene la forma

$$\Lambda(x)E^{(r)}(x) = p(x) + x^{r+1}g(x)$$

con $\text{grado}(\Lambda(x)) \leq \text{máx}\{L_r, L_m + r - m\}$ y $\text{grado}(p(x)) < \text{máx}\{L_r, L_m + r - m\}$. En consecuencia, $(\Lambda(x), \text{máx}\{L_r, L_m + r - m\})$ es una recurrencia que genera $(E_0, \ldots, E_{r-1})$. ▶

En general, existirá más de una iteración previa $m < r$ con $\Delta_m \neq 0$, y el teorema anterior no permite especificar una elección. El siguiente resultado muestra que la iteración más reciente con $\Delta_m \neq 0$ es la que produce la recurrencia de longitud mínima.

Teorema 11.29 (Berlekamp–Massey). *Sea* $\mathbf{E}^n = (E_0, \ldots, E_{n-1})$ *y sea* $\mathcal{L}(\mathbf{E}^{n-1}) = L$. *Si* $(\Lambda(x), L)$ *produce* $\mathbf{E}^{n-1}$ *pero no* $\mathbf{E}^n$, *entonces* $\mathcal{L}(\mathbf{E}^n) = \text{máx}\{L, n - L\}$.

Demostración. Por el teorema de Massey, es suficiente con demostrar que $\mathcal{L}(\mathbf{E}^n) \leq \text{máx}\{L, n - L\}$. Sea $\mathbf{0}^{n-1}$ la secuencia compuesta por $n - 1$ ceros. Tenemos entonces dos casos

a) $\mathbf{E}^{n-1} = \mathbf{0}^{n-1}$ y $\mathbf{E}^n = \mathbf{0}^{n-1}$, con $\Delta \neq 0$. El resultado es inmediato.

b) El caso $\mathbf{E}^{n-1} \neq \mathbf{0}^{n-1}$ se demuestra por inducción. Sea m tal que $\mathcal{L}(\mathbf{E}^{m-1}) < \mathcal{L}(\mathbf{E}^m) = \mathcal{L}(\mathbf{E}^{n-1})$. La hipótesis de inducción consiste en que $\mathcal{L}(\mathbf{E}^{n-1}) = m - \mathcal{L}(\mathbf{E}^{m-1}) = m - L_{m-1}$. En virtud del teorema anterior,

$$\begin{aligned}\mathcal{L}(\mathbf{E}^n) &\leq \text{máx}\{L, L_{m-1} + n - m\} \\ &= \text{máx}\{\mathcal{L}(\mathbf{E}^{n-1}), n - \mathcal{L}(\mathbf{E}^{n-1})\} = \text{máx}\{n - L, L\}.\end{aligned}$$ ▶

TEOREMA 11.30 (ALGORITMO DE BERLEKAMP–MASSEY). *Supóngase que $S_1, S_2, \ldots, S_{2t}$ son elementos dados de un cuerpo cualquiera. Con las condiciones iniciales $\Lambda^{(0)}(x) = 1$, $B^{(0)}(x) = 1$ y $L_0 = 0$, el algoritmo siguiente calcula el polinomio $\Lambda^{(2t)}(x)$:*

$$\Delta_r = \sum_{j=0}^{n-1} \Lambda_j^{(r-1)} S_{r-j}$$

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

$$\begin{pmatrix}\Lambda^{(r)}(x) \\ B^{(r)}(x)\end{pmatrix} = \begin{pmatrix}1 & -\Delta_r x \\ \Delta_r^{-1}\delta_r & (1-\delta_r)x\end{pmatrix} \begin{pmatrix}\Lambda^{(r-1)}(x) \\ B^{(r-1)}(x)\end{pmatrix}, \quad r = 1, \ldots, 2t.$$

en donde $\delta_r = 1$ si $\Delta_r \neq 0$ y $2L_{r-1} \leq r - 1$, y $\delta_r = 0$ en otro caso. $\Lambda^{(2t)}(x)$ es el polinomio de grado mínimo con $\Lambda_0^{(2t)} = 1$ que satisface

$$S_r + \sum_{j=1}^{n-1} \Lambda_j^{(2t)} S_{r-j} = 0, \quad r = L_{2t}, \ldots, 2t - 1.$$

En la forma compacta de este teorema, $\Delta_r = 0$ sólo cuando $\delta_r = 0$, y en ese caso debe entenderse que $\Delta_r^{-1}\delta_r = 0$. En el algoritmo, el polinomio $B^{(r)}(x)$ vale $\Delta_m^{-1} x^{r-m} \Lambda^{(m)}(x)$. Si $\delta_r = 1$, $B^{(r)}(x)$ se sustituye por $\Lambda^{(r)}(x)$; y si $\delta_r = 0$, $B^{(r)}(x)$ se multiplica por x.

Tal como se ha presentado, el algoritmo es válido para cualquier cuerpo (finito o no). En el caso especial de los códigos binarios ocurre una simplificación: como las magnitudes de error son conocidas, es suficiente con hallar el polinomio localizador, ya que $e_{i_l} = 1$ para $l = 1, \ldots, s$. Pero con el algoritmo de Berlekamp–Massey ocurre otra simplificación adicional: $\Delta_r = 0$ en todas las iteraciones de índice r par.

TEOREMA 11.31. *En $\mathbb{F}_2$, para cualquier polinomio de conexión $\Lambda(x)$ y cualquier secuencia de elementos $S_1, \ldots, S_{2t-1}$ en la que $S_{2j} = S_j^2$ para $2j \leq 2v - 1$, supóngase*

$$S_j = -\sum_{k=1}^{n-1} \Lambda_j S_{j-i}, \quad j = v, \ldots, 2v - 1.$$

Entonces, el valor siguiente que esta recurrencia produce cumple $S_{2v} = S_v^2$.

Demostración. Sea $\mathbf{v}$ un vector de un código binario. En consecuencia

$$S_{2j} = v\left(\alpha^{2j}\right) = \left(v(\alpha^j)\right)^2 = S_j^2.$$

El teorema quedará demostrado si se hallan expresiones idénticas para S_{2v} y S_v^2. Así, en primer lugar

$$S_v^2 = \left(\sum_{i=1}^{n-1} \Lambda_i S_{v-i}\right)^2 = \sum_{i=1}^{n-1} \Lambda_i^2 S_{v-i}^2 = \sum_{i=1}^{n-1} \Lambda_i^2 S_{2v-2i}.$$

Y

$$S_{2v} = \sum_{k=1}^{n-1} \Lambda_k S_{2v-k} = \sum_{k=1}^{n-1} \sum_{i=1}^{n-1} \Lambda_k \Lambda_i S_{2v-k-i}.$$

En esta suma doble, los términos con $i \neq k$ aparecen dos veces, por simetría, y en $\mathbb{F}_2$ ambos suman cero. Por lo tanto, a la suma sólo contribuyen los términos con $i = k$, y

$$S_{2v} = \sum_{k=1}^{n-1} \Lambda_k^2 S_{2v-2k}. \qquad \blacktriangleright$$

La aplicación de este corolario es inmediata. Puesto que $\Delta_r = 0$ en las iteraciones pares, es posible combinarlas de dos en dos y tener, para r impar,

$$\begin{pmatrix} \Lambda^{(r)}(x) \\ B^{(r)}(x) \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_r x^2 \\ \Delta_r^{-1}\delta_r & (1-\delta_r)x^2 \end{pmatrix} \begin{pmatrix} \Lambda^{(r-2)}(x) \\ B^{(r-2)}(x) \end{pmatrix}.$$

De este modo, la complejidad del algoritmo de Berlekamp–Massey para códigos binarios es sólo la mitad, aproximadamente.

CAPÍTULO 12

Códigos convolucionales

El presente capítulo se ocupa de la clase de códigos convolucionales. Esencialmente, los convolucionales son códigos lineales *con memoria*. Pero aunque conservan como aspecto fundamental la linealidad, la estructura matemática de los códigos convolucionales es muy distinta a la de los códigos de bloques algebraicos.

En concreto, mientras que el proceso de codificación en los códigos de bloques lineales consiste en una asignación lineal de palabras del código de longitud fija a cada grupo de símbolos de una longitud dada, en los códigos convolucionales la generación de la redundancia se realiza con un proceso más general: la secuencia del código es la respuesta de un sistema lineal e invariante en el tiempo a la secuencia completa de símbolos del mensaje. Los códigos convolucionales (introducidos en 1955 por P. Elias y presentados de nuevo en 1959 por D. W. Hagelbarger bajo el nombre de códigos recurrentes) son conceptualmente simples y poseen, en la mayoría de los casos, una capacidad de control de errores tan buena o mejor, a igual tasa de codificación, que la de los códigos de bloques; por contra, presentan una estructura matemática menos simple que la de éstos últimos, y eso explica que sea más difícil analizar con exactitud sus prestaciones, así como que sólo se hayan podido determinar hasta la fecha unos pocos criterios empíricos de diseño.

Desde el hallazgo de un algoritmo óptimo y eficiente de decodificación (el algoritmo de Viterbi), los códigos convolucionales comenzaron a ser ampliamente aplicados a la construcción de sistemas de comunicaciones digitales. Encuentran utilidad, sobre todo, en canales con fuertes limitaciones de ancho de banda (por ejemplo, en la transmisión de datos a través de una línea telefónica, en ciertos sistemas de comunicaciones por satélite o en redes de

telefonía móvil digital) porque, cuando se combinan con técnicas de modulación digital apropiadas, consiguen un notable incremento de la fiabilidad del canal sin el coste de aumentar el ancho de banda de la señal modulada. El uso integrado de códigos convolucionales y modulaciones digitales se conoce como modulación *trellis*, y ha demostrado ser uno de los avances más significativos en la aplicación de las técnicas de codificación de canal durante la década de los ochenta.

En este capítulo se comienza por definir la clase de codificadores convolucionales y se desarrollan después ciertas técnicas gráficas para aprehender su estructura dinámica, así como técnicas analíticas para investigar sus propiedades de corrección de errores. A continuación, se deduce el algoritmo de decodificación óptimo ML y se presenta una realización eficiente del mismo, con un orden de complejidad igual al número de estados del codificador. El capítulo termina con un estudio de la probabilidad de error residual sobre un canal discreto ruidoso.

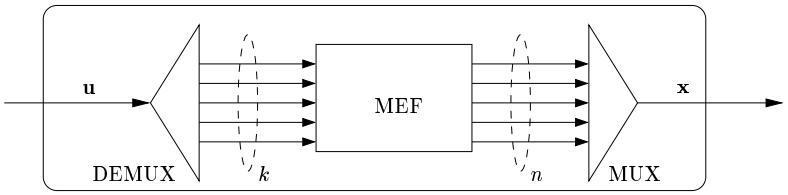
12.1. Definición

Sea $\mathbf{u}_0$, $\mathbf{u}_1$, ... una secuencia de bloques de k símbolos del cuerpo $\mathbb{F}_q = \mathrm{GF}(q)$. Un codificador convolucional $[n, k]$ sobre $\mathbb{F}_q$ es un método de transformar linealmente cada k-tupla $\mathbf{u}_j$ en una n-tupla $\mathbf{x}_j$ de símbolos de $\mathbb{F}_q$ de tal forma que $\mathbf{x}_j$ sea dependiente de $\mathbf{u}_j$, $\mathbf{u}_{j-1}$, ..., $\mathbf{u}_{j-m}$, ...; por consiguiente, la respuesta de un codificador convolucional es función tanto de los símbolos del bloque actual presentes en la entrada como de cierto número de símbolos de los bloques anteriores, siendo éstos últimos los que determinan en conjunto el *estado* o *memoria* del codificador. Tal estado interno del codificador, que adopta siempre valores de un conjunto finito, puede depender sólo de un número finito de bloques anteriores, es decir, sólo de $\mathbf{u}_{j-1}, \ldots, \mathbf{u}_{j-m}$ para algún $m \geq 1$ fijo; o bien de toda la secuencia de entradas previas, es decir, de $\mathbf{u}_i$, para cualquier $i < j$.

Dada esta forma de proceder, es claro que los símbolos de $\mathbf{u}_j$ afectan no sólo a la n-tupla de salida $\mathbf{x}_j$, sino también a las m siguientes, $\mathbf{x}_{j+1}, \ldots, \mathbf{x}_{j+m}$, ..., en donde no hay por qué suponer m finito. Este grado añadido de redundancia, inexistente en los códigos de bloques y mayor a medida que la memoria m aumenta, es justamente el que hace posible la construcción de códigos robustos (con baja probabilidad de error si la memoria es grande).

He aquí una definición genérica formal de un codificador convolucional.

DEFINICIÓN 12.1. *Un codificador convolucional $C[n, k]$, con alfabeto de codificación $\mathbb{F}_q$, es una máquina de estados finita con entradas $\mathbf{u} \in \mathbb{F}_q^k$ y*



MEF: Máquina de estados finita

Figura 12.1. Esquema conceptual de un codificador convolucional.

salidas $\mathbf{x} \in \mathbb{F}_q^n$*, en la que en cada instante de tiempo* $t = 0, 1, \ldots$ *(*$t \in \mathbb{Z}^+$*):*

a) el próximo estado del codificador es una combinación lineal del estado y la entrada actuales;

b) los símbolos de salida son combinación lineal del estado y las entradas actuales.

En un instante arbitrario de tiempo t, el modelo dinámico de variables de estado de un codificador convolucional responde entonces al par de ecuaciones lineales

$$\mathbf{s}_{t+1} = \mathbf{u}_t \cdot \mathcal{A} + \mathbf{s}_t \cdot \mathcal{B} \tag{12.1}$$

$$\mathbf{x}_t = \mathbf{u}_t \cdot \mathcal{C} + \mathbf{s}_t \cdot \mathcal{D} \tag{12.2}$$

siendo $\mathbf{u}_t$ el vector de entrada en el instante t, $\mathbf{x}_t$ el vector de salida en t y $\mathbf{s}_t$ el vector de variables de estado en t (con dimensión M, por ejemplo); y siendo $\mathcal{A}$ una matriz $k \times M$, $\mathcal{B}$ una matriz $M \times M$, $\mathcal{C}$ una matriz $k \times n$ y $\mathcal{D}$ una matriz $M \times n$, todas ellas con elementos de $\mathbb{F}_q$. El parámetro M, la dimensión del vector de estado, se denomina *grado* del codificador.

Definición 12.2. *El conjunto de todas las secuencias posibles de símbolos de salida de un codificador convolucional es un código convolucional.*

Por decirlo de un modo equivalente: un codificador convolucional es un sistema lineal discreto invariante en el tiempo; un dispositivo que transforma linealmente una *secuencia* de vectores de dimensión k, $\mathbf{u}_0, \mathbf{u}_1, \ldots$, en una *secuencia del código* de vectores[1] de dimensión n, $\mathbf{x}_0, \mathbf{x}_1, \ldots$, ambas sobre

[1] Cabe, pues, definir un codificador convolucional como un operador lineal

$$\mathcal{L} : \mathcal{F}(\mathbb{F}_q^k) \longrightarrow \mathcal{F}(\mathbb{F}_q^n)$$
$$\mathbf{a} + \mathbf{b} \longrightarrow (\mathbf{a} + \mathbf{b})\mathcal{L} = \mathbf{a}\mathcal{L} + \mathbf{b}\mathcal{L}$$

entre el espacio funcional de las sucesiones de vectores de $\mathbb{F}_q^k$ y el espacio de las sucesiones

el alfabeto $\mathbb{F}_q$.

En la teoría de sistemas lineales discretos, la relación matemática entre secuencias de entrada y de salida se expresa de un modo más adecuado no en el dominio del tiempo sino en el dominio transformado-D. Dada una sucesión $s_0, s_1, \ldots$ de símbolos de $\mathbb{F}_q$, diremos que la expresión formal

$$s(D) = \sum_{t=0}^{\infty} s_t D^t$$

es su transformada-D.[2] La suma de transformadas-D se define como

$$x(D) + y(D) = \sum_{t=0}^{\infty} (x_t + y_t) D^t \tag{12.3}$$

y el producto como

$$x(D) \cdot y(D) = \sum_{t=0}^{\infty} \left(\sum_{i=0}^{t} x_i y_{t-i} \right) D^t, \tag{12.4}$$

es decir, de manera análoga a como si los operandos fuesen polinomios en la variable D. La transformada-D de una secuencia de vectores de $\mathbb{F}_q^n$, $\{\mathbf{s}_t : t \geq 0\}$, se puede definir entonces formalmente como

$$\mathbf{s}(D) = \sum_{t=0}^{\infty} \mathbf{s}_t D^t,$$

expresión que debe entenderse como el vector cuyos elementos son las transformadas-D de cada una de las n componentes de la secuencia $\{\mathbf{s}_t : t \geq 0\}$. La suma de transformadas-D de dos sucesiones de vectores se define ahora de la manera usual a partir de la operación de suma de vectores

$$\mathbf{x}(D) + \mathbf{y}(D) = \sum_{t=0}^{\infty} (\mathbf{x}_t + \mathbf{y}_t) D^t.$$

Multiplicando ambos miembros de las ecuaciones de estado (12.1)-(12.2)

de vectores de $\mathbb{F}_q^n$.

[2] Esta transformada es una mera operación formal en la que D admite el significado de un operador de retardo. No se debe confundir D con una variable compleja como la que aparece en otras transformaciones en el ámbito del tratamiento de señales, como en la transformada-z.

Por otra parte, resulta elemental ver que la transformada-D es una operación invertible, ya que la sucesión de símbolos son los coeficientes de la expansión única en serie de potencias de la función transformada.

por D^t y sumando para todo $t \geq 0$, se tiene

$$\sum_{t\geq 0} \mathbf{s}_{t+1} D^t = \sum_{t\geq 0} \mathbf{u}_t D^t \mathcal{A} + \sum_{t\geq 0} \mathbf{s}_t D^t \mathcal{B} \tag{12.5}$$

$$\sum_{t\geq 0} \mathbf{x}_t D^t = \sum_{t\geq 0} \mathbf{u}_t D^t \mathcal{C} + \sum_{t\geq 0} \mathbf{s}_t D^t \mathcal{D}. \tag{12.6}$$

Las expresiones

$$\mathbf{s}(D) = \sum_{i\geq 0} \mathbf{s}_i D^i, \quad \mathbf{u}(D) = \sum_{i\geq 0} \mathbf{u}_i D^i, \quad \mathbf{x}(D) = \sum_{i\geq 0} \mathbf{x}_i D^i$$

son, respectivamente, las transformadas-D de la secuencia de vectores de estado, de la secuencia de vectores de entrada y de la secuencia de vectores de salida. Por lo tanto, suponiendo que $\mathbf{s}_0 = \mathbf{0}$, las ecuaciones (12.5)-(12.6) equivalen a

$$\begin{aligned} \mathbf{s}(D)D^{-1} &= \mathbf{u}(D)\mathcal{A} + \mathbf{s}(D)\mathcal{B} \\ \mathbf{x}(D) &= \mathbf{u}(D)\mathcal{C} + \mathbf{s}(D)\mathcal{D}. \end{aligned}$$

Despejando en ambas obtendremos expresiones explícitas para las transformadas $\mathbf{s}(D)$ y $\mathbf{x}(D)$:

$$\begin{aligned} \mathbf{s}(D) &= \mathbf{u}(D)(\mathcal{A}(D^{-1}\mathcal{I}_M - \mathcal{B})^{-1}) = \mathbf{u}(D)E(D) \\ \mathbf{x}(D) &= \mathbf{u}(D)(\mathcal{C} + E(D)\mathcal{D}) = \mathbf{u}(D)G(D) \end{aligned}$$

en función de las matrices $E(D)_{k\times M}$ y $G(D)_{k\times n}$ y la matriz identidad $\mathcal{I}_M$. La matriz $G(D)$ depende únicamente de la cuaterna $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$, pero, en principio, nada impide que existan muchos conjuntos de matrices $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ que den origen a la misma matriz transformada $G(D)$.

En cualquier caso, la transformada de una secuencia del código satisface la ecuación funcional

$$\mathbf{x}(D) = \mathbf{u}(D)G(D), \tag{12.7}$$

en la que $\mathbf{u}(D)$ identifica a una secuencia posible de mensajes y $\mathbf{x}(D)$ a una secuencia del código particular. Así pues, la matriz $G(D)$ *define* un codificador convolucional, puesto que establece, en el seno del dominio transformado D, la acción de éste sobre cualquier secuencia de símbolos de entrada. Por esta razón se llama a $G(D)$ *matriz generadora* del código convolucional.

Cada uno de los elementos de los vectores $\mathbf{x}(D)$, $\mathbf{u}(D)$ y de la matriz generadora $G(D)$ es una serie causal de potencias en la variable D con coeficientes en el cuerpo base $\mathbb{F}_q$, o sea, una expresión de la forma $\sum_{i\geq 0} a_i D^i$, con $a_i \in \mathbb{F}_q$. El conjunto de todas estas expresiones formales constituye un

dominio de integridad conmutativo[3] con las operaciones usuales de suma y producto de polinomios en la indeterminada D (las operaciones que definen las ecuaciones (12.3)-(12.4)), dominio que obviamente contiene a $\mathbb{F}_q$ y que además se puede ampliar fácilmente a un cuerpo $\mathbb{F}_q(D)$ considerando las series formales de Laurent $\sum_{i\geq m} a_i D^i$ para un entero m arbitrario, positivo, negativo o nulo. Pero $\mathbb{F}_q(D)$ es un conjunto excesivamente amplio: contiene funciones transformadas que no son realizables por ningún codificador convolucional ya que, por definición, la matriz generadora $G(D)$ se compone de funciones racionales y toda función racional admite una expansión en serie ilimitada con una parte periódica. El subconjunto de $\mathbb{F}_q(D)$ constituido por todas aquellas series formales que sí admiten una representación racional, esto es, que son expresables como cociente de dos polinomios $p(D)/q(D)$, $q(D) \neq 0$, es un subcuerpo de $\mathbb{F}_q(D)$, que representaremos por $\mathbb{F}_q[D]$. Por lo tanto, en virtud de la ecuación (12.7), los espacios lineales de vectores y de matrices con elementos en $\mathbb{F}_q[D]$ son el álgebra que describe los códigos convolucionales; y de hecho se puede definir sin más un código convolucional como un subespacio lineal de $(\mathbb{F}_q[D])^n$.

Puesto que los elementos de la matriz generadora son funciones racionales en D, cualquier operación elemental con las filas de $G(D)$ (permutar dos de ellas, multiplicar o dividir una cualquiera por un polinomio no nulo) produce una matriz equivalente que genera el mismo código. Es así que cualquier código convolucional admite siempre una matriz generadora cuyos elementos son polinomios en D, que se obtiene sin más que multiplicar cada fila de $G(D)$ por el mínimo común múltiplo de los denominadores de los elementos de esa fila. Recíprocamente, dada una matriz $G(D)$ formada por funciones racionales en D, existe una matriz $G'(D)$ equivalente a ella formada por elementos causales en D (esto es, cuyos elementos son series causales). De acuerdo con un teorema general de la teoría de sistemas lineales, existe para $G'(D)$ una realización del modelo de variables de estado, es decir, $G(D)$ corresponde a un cierto codificador convolucional. Así, se concluye que *el estudio de los codificadores convolucionales es equivalente al estudio de las matrices* $G(D)$. Esa teoría algebraica, que sienta un marco formal riguroso para la comprensión de las propiedades estructurales de los códigos convolucionales, es una contribución original de G. D. Forney y se desarrolla ampliamente en [20, 22, 23] y en [53, cap. 11].

Desde un punto de vista práctico, una realización de un codificador convolucional es cualquier mecanismo material o lógico que reproduzca las operaciones (12.1)-(12.2). Físicamente, suele consistir en un circuito aritmético formado a partir de componentes sumadores, multiplicadores (innecesarios en el caso binario) y elementos de memoria en $\mathbb{F}_q$.

[3] Un dominio de integridad conmutativo es un anillo conmutativo sin divisores de cero; por ejemplo, el conjunto de los números enteros.

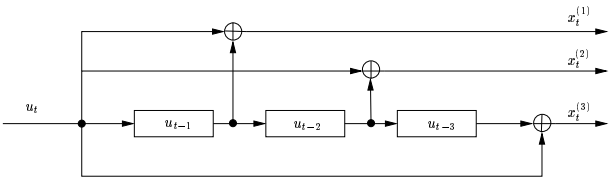


Figura 12.2. Codificador convolucional $G(D) = (1{+}D \;\; 1{+}D^2 \;\; 1{+}D^3)$ del ejemplo 12.1.

Ejemplo 12.1. Considérese el codificador convolucional binario $[3, 1]$ de grado 3 que responde a las ecuaciones de estado

$$\mathbf{s}_{t+1} = \mathbf{s}_t \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} + u_t \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}; \qquad \mathbf{x}_t = \mathbf{s}_t \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + u_t \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}.$$

Si en ellas se elimina la dependencia explícita del vector de estado, las ecuaciones de los símbolos de salida en el instante $t \geq 0$ son:

$$\begin{aligned} x_t^{(1)} &= u_t + u_{t-1} \\ x_t^{(2)} &= u_t + u_{t-2} \\ x_t^{(3)} &= u_t + u_{t-3}. \end{aligned}$$

Se ha representado en la figura 12.2 el esquema de una realización física de las mismas. El circuito sólo precisa un registro de memoria para contener los 3 símbolos anteriores de entrada. De acuerdo con las ecuaciones de generación que se acaban de escribir, la matriz generadora de este código es

$$G(D) = \begin{pmatrix} 1+D & 1+D^2 & 1+D^3 \end{pmatrix}.$$

El grado del codificador se puede reducir tomando la matriz generadora equivalente

$$G'(D) = \frac{1}{1+D} G(D) = \begin{pmatrix} 1 & 1+D & 1+D+D^2 \end{pmatrix}.$$

De vuelta al dominio del tiempo, las ecuaciones de entrada–salida del codificador dado por $G'(D)$ son

$$\begin{aligned} x_t^{(1)} &= u_t \\ x_t^{(2)} &= u_t + u_{t-1} \\ x_t^{(3)} &= u_t + u_{t-1} + u_{t-2} \end{aligned}$$

y la figura 12.3 muestra el esquema de un circuito de codificación para $G'(D)$, ahora con grado 2. Observe que ambas realizaciones del codificador,

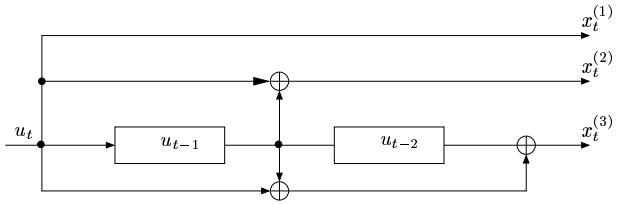


FIGURA 12.3. Codificador convolucional $G'(D) = (1 \;\; 1{+}D \;\; 1{+}D{+}D^2)$.

las de las figuras 12.2 y 12.3, son *no recurrentes* (no poseen conexiones por realimentación), siendo condición necesaria y suficiente para ello que su matriz generadora sea polinómica. Cuando un codificador $[n, 1]$ es no recurrente, el grado es el número máximo de símbolos futuros de cada una de las salidas que se ven afectados por un símbolo a la entrada.

Observe además que, con la segunda realización, el primer bit a la salida es una copia del de la entrada. En general, un codificador $[n, k]$ en el que, en cada instante de tiempo, k símbolos de salida son una réplica de los de entrada se denomina *sistemático*, y su matriz generadora es *sistemática*; y es obvio que, salvo por permutación de alguna de sus columnas, una matriz sistemática admitirá la forma

$$G(D) = \begin{pmatrix} I_{k\times k} & Q(D)_{k\times n-k} \end{pmatrix}$$

en donde $Q(D)$ es una matriz cuyos elementos son funciones racionales en la indeterminada D. ■

De entre todos los posibles codificadores de un mismo código convolucional, los que admiten una realización más simple son los no recurrentes con menor número de variables de estado.[4] Sus matrices generadoras son las *matrices canónicas*.

DEFINICIÓN 12.3 (GRADO EXTERNO). *Se llama grado del vector de polinomios $(x_1(D), x_2(D), \ldots, x_n(D))$ en la indeterminada D al máximo de los grados de sus elementos. El grado externo de la matriz de polinomios $G(D)$ es la suma de los grados de cada una de sus filas.*

DEFINICIÓN 12.4 (MATRIZ CANÓNICA). *Una matriz polinómica $G(D)$ es canónica si tiene el menor grado externo entre todas las matrices polinómicas generadoras del mismo código convolucional que ella.*

[4] Se toma como medida de la complejidad de un codificador convolucional el número de elementos de memoria que precisa. Además de las matrices canónicas, las matrices generadoras sistemáticas también son mínimas pero, en general, recurrentes.

Todo código convolucional posee al menos una matriz generadora canónica. Enunciado en un lenguaje matemático más general: cualquier subespacio de $(\mathbb{F}_q[D])^n$, el conjunto de vectores de funciones racionales en D, posee una base compuesta por polinomios cuyos grados suman una cantidad mínima. Este resultado, debido a G. D. Forney [23], posee importantes aplicaciones en campos como la teoría general de sistemas o la teoría de autómatas finitos y guarda íntima relación con algunos problemas algebraicos clásicos.

En el ejemplo 12.1, $(1 \;\; 1+D \;\; 1+D+D^2)$ es una matriz polinómica, canónica y sistemática de grado externo 2. Al haber un solo símbolo de entrada, el grado externo es también el grado del codificador. Es preciso señalar que, en un caso general, un código convolucional puede tener varias matrices generadoras canónicas. Se puede probar, sin embargo, que en todas ellas el conjunto de los grados externos de sus filas es invariante.

La relación entre los códigos convolucionales y los códigos de bloques lineales quedará un poco más clara si se presenta una caracterización de los primeros en el dominio del tiempo. Sin menoscabo alguno de la generalidad del planteamiento, supóngase, por conveniencia, que la memoria del codificador son los m bloques previos al bloque actual de entrada

$$\mathbf{s}_t = (\mathbf{u}_{t-1}, \mathbf{u}_{t-2}, \ldots, \mathbf{u}_{t-m})$$

y que el estado inicial es el estado nulo $\mathbf{s}_0 = (\mathbf{0}, \ldots, \mathbf{0})$.

En tal caso, la familia de ecuaciones

$$\mathbf{x}_t = \mathbf{u}_t\mathcal{C} + \mathbf{s}_t\mathcal{D}, \quad t \geq 0$$

se simplifica a una relación lineal entre la secuencia (infinita) de vectores de entrada $\mathbf{u} = (\mathbf{u}_0, \mathbf{u}_1, \ldots)$ y la secuencia código $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1, \ldots)$:

$$\mathbf{x} = \mathbf{u} \cdot G \tag{12.8}$$

para una cierta matriz G semi–infinita con la estructura

$$G = \begin{pmatrix} G_0 & G_1 & G_2 & \ldots & G_m & & & \\ & G_0 & G_1 & \ldots & G_{m-1} & G_m & & \\ & & G_0 & \ldots & G_{m-2} & G_{m-1} & G_m & \\ & & & \ddots & & \ddots & \ddots & \ddots \end{pmatrix}.$$

A la vista de (12.8), y por analogía con los códigos de bloques, también se suele denominar a G *matriz generadora* del código convolucional en el dominio del tiempo. En ella, $G_h = [g_{ij}^{(h)}]$, para $h = 0, \ldots, m$, son matrices de dimensiones $k \times n$ con elementos en el cuerpo $\mathbb{F}_q$; G_0 y G_m son matrices no nulas; y todos los bloques de G que no se indican valen 0. El elemento $g_{ij}^{(h)}$

de la matriz G_h es distinto de cero si y solamente si el j-ésimo símbolo de salida en el instante de tiempo t depende del i-ésimo símbolo de entrada en el instante de tiempo $t-h$. Así pues, si la entrada al codificador en reposo es el vector unitario $\mathbf{e}_i$ con un uno en la coordenada i-ésima, su secuencia de símbolos de salida será

$$\begin{bmatrix} g_{i1}^{(0)} & \dots & g_{i1}^{(m)} \\ \dots & \dots & \dots \\ g_{in}^{(0)} & \dots & g_{in}^{(m)} \end{bmatrix}$$

en donde las filas refieren cada una de las componentes de la salida y las columnas representan instantes de tiempo. Escribiendo la ecuación matricial (12.8) en forma explícita, el símbolo $1 \leq j \leq n$ de salida en el instante de tiempo $t \geq 0$ se calcula como

$$x_t^{(j)} = \sum_{h=0}^{m} \sum_{i=1}^{k} u_{t-h}^{(i)} g_{ij}^{(h)}.$$

O bien, en notación vectorial, como

$$\mathbf{x}_t = \sum_{h=0}^{m} \mathbf{u}_{t-h} \cdot G_h$$

que es, por supuesto, la operación de convolución que da nombre a esta clase de códigos.

G muestra una evidente estructura repetitiva, pues cada bloque de k filas son las k anteriores desplazadas n columnas hacia la derecha. Por convenio, se dice que la matriz de dimensiones finitas

$$\begin{pmatrix} G_0 & G_1 & \dots & G_m \end{pmatrix}$$

que, junto con el par $[n, k]$, identifica inequívocamente a G, es el *patrón generador* del código.

La semejanza estructural entre la matriz G y la matriz generadora de un código cíclico no es casual: un código cíclico binario $[n, k]$ puede verse como un código convolucional con un bit de entrada, un bit de salida y grado $n-k$, en el que la secuencia de salida se genera añadiendo, en la entrada, $n-k$ ceros tras cada grupo de k símbolos consecutivos.

Por sencillez, en el resto del capítulo se tratará sólo con códigos convolucionales binarios. Cuando se precise especificar que disponen de k símbolos de entrada, n de salida y grado m, se hará con la notación $C[n, k, m]$.

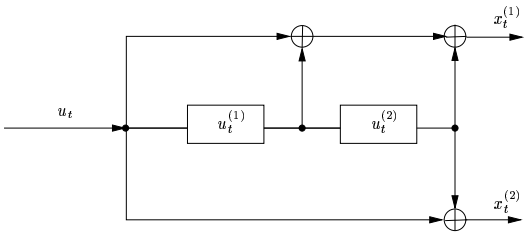


Figura 12.4. Codificador convolucional $G(D) = (1 + D + D^2, 1 + D^2)$ (ejemplo 12.2).

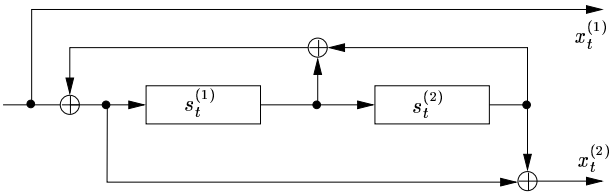


Figura 12.5. Esquema del codificador convolucional $[2, 1]$ con matriz generadora $G'(D) = (1 \quad (1 + D^2)/(1 + D + D^2))$.

Ejemplo 12.2. Las ecuaciones de estado que corresponden al codificador convolucional binario $[2, 1]$ de matriz generadora

$$G(D) = \begin{pmatrix} 1 + D + D^2 & 1 + D^2 \end{pmatrix}$$

son las siguientes:

$$\begin{aligned} \mathbf{s}_{t+1} &= (u_t, u_{t-1}) \\ \mathbf{x}_t &= (u_t + u_{t-1} + u_{t-2}, u_t + u_{t-2}). \end{aligned}$$

El grado de este codificador es obviamente 2 y el esquema del mismo es el de la figura 12.4.

Dividiendo las entradas de $G(D)$ por $1 + D + D^2$ resulta la matriz equivalente sistemática

$$G'(D) = \begin{pmatrix} 1 & \dfrac{1 + D^2}{1 + D + D^2} \end{pmatrix}$$

cuya realización física (figura 12.5) tiene como antes grado 2, pero memoria *infinita*; en efecto, el segundo bit de salida depende de un número ilimitado de bits de entrada anteriores. Veámoslo explícitamente: las ecuaciones de estado son, en este caso,

$$\begin{aligned} \mathbf{s}_{t+1} &= (u_t + s_t^{(1)} + s_t^{(2)}, s_t^{(1)}) \\ \mathbf{x}_t &= (u_t, u_t + s_t^{(1)}). \end{aligned}$$

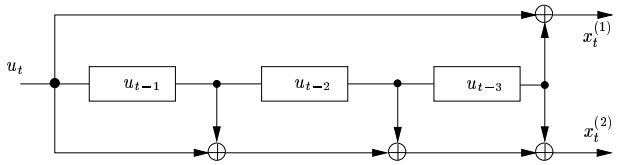


FIGURA 12.6. Esquema del codificador con matriz generadora catastrófica $G''(D) = (1 + D^3 \quad 1 + D + D^2 + D^3)$.

Si en este sistema intentamos eliminar $s_t^{(1)}$ en la segunda identidad utilizando para ello la identidad primera, resultaría

$$\mathbf{x}_{t+1} = (u_t, u_t + u_{t-1} + u_{t-2} + \cdots)$$

que muestra que $\mathbf{x}_t$ está afectado por entradas arbitrariamente distantes en el pasado. Por lo tanto, la memoria infinita es una característica del codificador elegido, no del código convolucional. Como es obvio, los codificadores asociados a matrices generadoras polinómicas tienen siempre memoria finita. Pero aun dentro de la clase de matrices generadoras polinómicas, no todas son igual de convenientes. Por ejemplo, si se multiplica $G(D)$ por $1 + D$ se obtiene la matriz

$$G''(D) = \left(1 + D^3 \quad 1 + D + D^2 + D^3\right)$$

cuyo codificador asociado aparece en la figura 12.6. Si suponemos la secuencia de entrada

$$\mathbf{u}(D) = 1 + D + D^2 + D^3 + \cdots = \frac{1}{1 + D}$$

o lo que es igual, el mensaje compuesto por una secuencia ilimitada de unos, entonces la secuencia del código que le asignaría el codificador $G''(D)$ sería

$$\mathbf{x}(D) = \mathbf{u}(D)G''(D) = \frac{1}{1 + D}G''(D) = \left(1 + D + D^2 \quad 1 + D^2\right).$$

Ahora bien, $\mathbf{x}(D)$ tiene peso Hamming 5; luego la regla de transformación elegida asigna a algunas secuencias de entrada de peso Hamming infinito secuencias de salida con un número finito de símbolos no nulos. Esto significa que existen secuencias del código que, difiriendo entre sí en un número finito de símbolos, han sido generadas por secuencias de entrada que difieren en un número infinito de posiciones. De ocurrir esta circunstancia, el decodificador podría asignar a vectores de error con peso Hamming finito una estimación de un número infinito de errores en la secuencia transmitida, algo claramente desaconsejable. Como habrá ocasión de ver posteriormente, el problema es que $G''(D)$ no posee una matriz pseudo–inversa polinómica, sino racional:

$$G''(D)\left(\frac{1}{D + D^2} \quad \frac{1}{D + D^2}\right)^t = 1.$$

Por esta razón, a los codificadores en que es posible que secuencias de entrada de peso infinito den secuencias de salida con peso finito se los conoce como catastróficos. Es muy importante observar que no tiene sentido hablar de códigos convolucionales catastróficos, sino sólo de codificadores catastróficos. Esto es, no se debe identificar jamás un código convolucional (que no es más que un conjunto de secuencias del código) con un codificador convolucional (que es una asignación particular entre secuencias del código y mensajes). ■

12.2. Representación reticular de un código convolucional

Un código (de bloques o convolucional) se puede representar por medio de un grafo dirigido etiquetado en el que cada camino se corresponde con una palabra del código. Este grafo surge de manera natural al considerar una realización de espacio de estados para el proceso de codificación y adopta a menudo una forma reticular[5] que pone claramente de manifiesto algunas de las características estructurales del código aparte de la linealidad. La regularidad de esta represantación gráfica permite, por ejemplo, desarrollar algoritmos de decodificación eficientes.

En el caso de los códigos convolucionales, el diagrama es una representación gráfica de la evolución dinámica de una máquina de estados finita. Consiste en representar su diagrama de estados y transiciones, extendido en el tiempo, por medio un grafo dirigido cuyos caminos señalan las posibles secuencias de estados que puede atravesar el sistema, así como los símbolos de entrada y de salida asociados a esos cambios de estado. La conveniente sugerencia de representar las características de un código convolucional a través de un diagrama reticular se debe también a G. D. Forney. Y si bien una retícula es una representación sobreabundante del código, resultará de gran utilidad más adelante para comprender el proceso de decodificación.

Supóngase que $\mathcal{C}$ es un codificador convolucional binario $[n, k]$ con grado m. En los términos a los que remite la definición 12.1, $\mathcal{C}$ es el autómata finito $(\Omega, \mathcal{L}_k, \mathcal{L}_n, \Gamma, \Xi)$. En esta quíntupla, Ω denota el espacio de estados

[5] En inglés, el término con el que se nombra a este diagrama es *trellis*, que nosotros hemos preferido traducir por *retícula* por la siguiente razón: los nodos de un diagrama *trellis* de un código convolucional se pueden poner en biyección con un subconjunto de los puntos de una retícula bidimensional

$$\mathcal{R} = \{\lambda_1 \mathbf{e}_1 + \lambda_2 \mathbf{e}_2 : \lambda_1, \lambda_2 \in \mathbb{Z}^+\}$$

en donde $\mathbf{e}_1 = (0, 1)$ y $\mathbf{e}_2 = (1, 0)$ son la base canónica de $\mathbb{R}^2$. $\mathcal{R}$ es un subgrupo aditivo de $\mathbb{R}^2$. Conway [13] trata extensamente la teoría de los códigos definidos sobre grafos.

del codificador, siendo el estado el conjunto de símbolos almacenados en los m elementos de memoria del dispositivo. Naturalmente $\Omega = \mathbb{F}_2^m$, el conjunto de todas las secuencias binarias de longitud m. Los conjuntos $\mathcal{L}_k = \mathbb{F}_2^k$ y $\mathcal{L}_n = \mathbb{F}_2^n$ son, respectivamente, el alfabeto de entrada y el alfabeto de salida del codificador.

Cuando, en el instante de tiempo t, el estado del sistema es $\mathbf{s}_t \in \Omega$ y la entrada es $\mathbf{u}_t \in \mathcal{L}_k$, la dinámica del codificador está gobernada por Γ y Ξ, que son las dos funciones que caracterizan a cualquier autómata:

a) Γ es la función de transición entre estados,

$$\begin{aligned} \Gamma : \mathcal{L}_k \times \Omega &\longrightarrow \Omega \\ (\mathbf{u}_t, \mathbf{s}_t) &\longrightarrow \mathbf{s}_{t+1} \end{aligned}$$

que indica el estado al que pasará el dispositivo tras introducir las entradas $\mathbf{u}_t$ en su memoria. Note, en consecuencia, que si la entrada es aleatoria entonces la secuencia de estados $\{\mathbf{s}_t\}_{t\geq 0}$ es un proceso de Markov de tiempo discreto con espacio de estados también discreto (una cadena de Markov y que si la secuencia de entrada es estacionaria, la secuencia de estados $\{\mathbf{s}_t\}_{t\geq 0}$ es una cadena de Markov homogénea. Por otra parte, cuando entre los elementos de memoria no existan conexiones por realimentación, algunos de los símbolos de entrada, posiblemente todos, entrarán directamente a los elementos de memoria del codificador, es decir, si

$$\mathbf{s}_t = (s_1, s_2, \ldots, s_m), \quad \mathbf{u}_t = (u_t^{(1)}, u_t^{(2)}, \ldots, u_t^{(k)})$$

entonces

$$\mathbf{s}_{t+1} = \Gamma(\mathbf{u}_t, \mathbf{s}_t) = (u_t^{(1)}, \ldots, u_t^{(j)}, s_1, \ldots, s_{m-j}), \quad 0 < j \leq k.$$

b) Ξ es la función de salida,

$$\begin{aligned} \Xi : \mathcal{L}_k \times \Omega &\longrightarrow \mathcal{L}_n \\ (\mathbf{u}_t, \mathbf{s}_t) &\longrightarrow \mathbf{x}_t \end{aligned}$$

que determina los símbolos de salida correspondientes la entrada $\mathbf{u}_t$ cuando el estado es $\mathbf{s}_t$; como un código convolucional admite un patrón generador lineal, entonces

$$\begin{aligned} \mathbf{x}_t &= \Xi(\mathbf{u}_t, \mathbf{s}_t) \\ &= \sum_{h=0}^{m} \mathbf{u}_{t-h} \cdot G_h = \mathbf{u}_t \cdot G_0 + \sum_{h=1}^{m} \mathbf{u}_{t-h} \cdot G_h \\ &= \mathbf{u}_t \cdot G_0 + f(\mathbf{s}_t). \end{aligned}$$

Las funciones de transición y salida son invariantes en el tiempo (no dependen del índice temporal t).

En lo sucesivo, se seguirá siempre el convenio de suponer que un codificador convolucional comienza en el estado inicial $\mathbf{0} = (0, 0, \ldots, 0)$ en el instante $t = 0$. En cualquier otro instante t, el codificador se encuentra en un estado $\mathbf{s}_t$ de entre un conjunto finito de estados permitidos. En el diagrama reticular asociado al codificador, un estado $\mathbf{s}_t$ permitido en el instante t se representa como un nodo de un grafo, y se etiqueta si es necesario con la secuencia de m símbolos que identifican al vector de variables de estado. El conjunto de nodos (estados) posibles en el tiempo t conforman la etapa t del diagrama reticular, $\mathcal{N}_t$.

Ante la entrada $\mathbf{u}_t$, el codificador realiza un cambio de estado o transición instantánea desde el estado $\mathbf{s}_t$ al estado $\mathbf{s}_{t+1} = \Gamma(\mathbf{u}_t, \mathbf{s}_t)$, transición que en el diagrama reticular se representa como un arco dirigido desde el nodo $\mathbf{s}_t$ en la etapa t al nodo $\mathbf{s}_{t+1}$ en la etapa $t+1$. Toda transición se marca con el par de valores $(\mathbf{u}_t, \mathbf{x}_t = \Xi(\mathbf{u}_t, \mathbf{s}_t))$, es decir, con los símbolos de entrada que originan la transición y con los símbolos de salida que produce el codificador. El arco o rama de una transición $(\mathbf{s}_t, \mathbf{s}_{t+1})$ es un arco de salida del estado $\mathbf{s}_t$ e incidente o de entrada en el estado $\mathbf{s}_{t+1}$. Dado un arco $a = (\mathbf{s}_t, \mathbf{s}_{t+1}, e)$, en donde e es su etiqueta, el nodo para el cual a es incidente se representará habitualmente por $a\circ$; y aquél para el que a es una rama de salida, por $\circ a$. Dos nodos conectados por una transición son adyacentes. Los símbolos de salida $\mathbf{x}_t$ se emiten instantáneamente durante la transición $(\mathbf{s}_t, \mathbf{s}_{t+1})$.

Un camino en la representación reticular entre los estados $\mathbf{s}_t$ y $\mathbf{s}_{t'}$, para $t < t'$, es cualquier secuencia ordenada de nodos adyacentes con comienzo en el nodo $\mathbf{s}_t$ y final en $\mathbf{s}_{t'}$. Un camino p queda completamente determinado si se dan el estado inicial y la secuencia de $t' - t$ entradas asociadas a las transiciones del camino, o lo que es igual, si se especifica su secuencia de arcos etiquetados: $p = a_1 a_2 \cdots a_n$.

Como se ha explicado, a cada camino del grafo reticular se le asigna además la secuencia ordenada de símbolos de salida de cada transición que lo constituye, de manera que los caminos de la retícula definen las secuencias de salida válidas del codificador (el código). La siguiente definición resume las características fundamentales de los diagramas reticulares.

Definición 12.5 (Diagrama reticular). *Un diagrama reticular o retícula es un grafo $\mathcal{T} = (\mathcal{N}, \mathcal{A}, \mathcal{E})$ dirigido, acíclico y etiquetado, cuyo conjunto de nodos $\mathcal{N}$ admite una partición (finita o numerable)*

$$\mathcal{N} = \mathcal{N}_0 \cup \mathcal{N}_1 \cup \mathcal{N}_2 \cup \ldots$$

tal que $\forall(\mathbf{n}, \mathbf{n}', e) \in \mathcal{A}$ $\exists i : \mathbf{n} \in \mathcal{N}_i$, $\mathbf{n}' \in \mathcal{N}_{i+1}$, $e \in \mathcal{E}$ y para todo i cualquier nodo de $\mathcal{N}_0 \cup \cdots \cup \mathcal{N}_i$ pertenece al menos a un camino con origen en algún

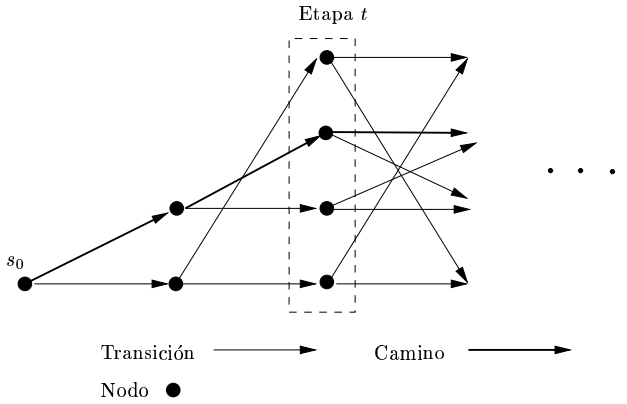


FIGURA 12.7. Elementos de un diagrama reticular.

nodo de $\mathcal{N}_0$ y final en alguno de $\mathcal{N}_i$.

Lo sustancial, por tanto, es que un nodo de una retícula puede ser ordenado según el número de arcos en cualquier camino desde un nodo raíz en $\mathcal{N}_0$ hasta él. Los caminos formados por i arcos finalizan en algún nodo de $\mathcal{N}_i$, y $\mathcal{N}_i$ se denomina etapa i de la retícula. El alfabeto de etiquetas $\mathcal{E}$ de los arcos es totalmente arbitrario y depende sólo del dominio de aplicación. En el caso de la teoría de códigos, $\mathcal{E}$ es el alfabeto de codificación y se dirá que la retícula $\mathcal{T}$ representa al código $\mathcal{C}$ si el conjunto formado por las secuencias de etiquetas de los caminos de $\mathcal{T}$ es $\mathcal{C}$.

En una retícula deducida de un codificador convolucional $\mathcal{C}[n,k,m]$ hay naturalmente un único nodo en la etapa inicial $\mathcal{N}_0 = \{\mathbf{0}\}$, el correspondiente al vector de estado nulo en el codificador. Los nodos de la etapa i representan los estados internos a los que es posible llegar en el instante de tiempo i, y cualquier nodo posee 2^k arcos de salida; asimismo, en cualquier nodo de una etapa $t > m$ convergen 2^k arcos provenientes de nodos de la etapa $t-1$. La etiqueta asociada a un arco a es $\alpha(a) = (\mathbf{u}; \Xi(\mathbf{u}, \mathbf{s}_i))$, el resultado de concatenar la secuencia de símbolos de entrada que causan la transición a con la secuencia de símbolos de salida que produce el codificador.

Tal como se ha definido, el diagrama reticular sirve, ante todo, para mostrar gráficamente la evolución del codificador y así revelar ciertas propiedades estructurales del código. Veamos algunas de ellas.

Se dice que un estado $\mathbf{s}_t$ en la etapa t es alcanzable si existe una secuencia de entradas que lleve al codificador del estado $\mathbf{s}_0$ al estado $\mathbf{s}_t$ en t transiciones, es decir, si existe un camino de longitud t en el diagrama reticular de $\mathbf{s}_0$ a $\mathbf{s}_t$. Por definición, todos los estados en una retícula correspondiente a un

código convolucional son alcanzables para $t \geq m$. Por otra parte, el que las funciones de transición y de salida sean invariantes en el tiempo implica que el propio diagrama reticular de un código convolucional también posee una estructura invariante. Cualquier estado $\mathbf{s} \in \Omega$ es alcanzable desde el estado inicial $\mathbf{s}_0$ en m transiciones o menos. Por tanto, los estados alcanzables en el tiempo $t \geq m$ son todos los de Ω. Más aún, también para los tiempos $t \geq m$, por la propiedad de invariancia temporal, las transiciones entre dos etapas sucesivas son siempre las mismas, y están asociadas a las mismas entradas. La retícula es, pues, a partir de la etapa $t \geq m$, una estructura regular, repetitiva, que se extiende indefinidamente en el tiempo.

La estructura de un diagrama reticular de un código convolucional presenta un patrón de interconexión entre etapas ciertamente regular: el conjunto de transiciones entre dos etapas se puede dividir en componentes paralelos idénticos. Para ver cómo tal cosa es posible, definamos la siguiente relación entre dos estados, $\mathbf{s}_t^{(1)}$ y $\mathbf{s}_t^{(2)}$, de una misma etapa

$$\begin{aligned} &\mathbf{s}_t^{(1)} \sim_F \mathbf{s}_t^{(2)} \Leftrightarrow \exists\, \mathbf{u}_t^{(1)}, \mathbf{u}_t^{(2)} \text{ y } \mathbf{s}_{t+1} \in \mathcal{N}_{t+1} : \\ &\mathbf{s}_{t+1} = \mathbf{u}_t^{(1)}\mathcal{A} + \mathbf{s}_t^{(1)}\mathcal{B} = \mathbf{u}_t^{(2)}\mathcal{A} + \mathbf{s}_t^{(2)}\mathcal{B}, \end{aligned}$$

en donde $\mathcal{A}$ y $\mathcal{B}$ son las matrices que, en las ecuaciones del modelo de estados, controlan la dinámica de la evolución interna del codificador. Es decir, dos estados estarán $\sim_F$-relacionados si existe algún estado en la etapa siguiente alcanzable desde ambos. Pues bien, esta relación binaria es una equivalencia. La clase de equivalencia del estado $\mathbf{s}_t$ consta de $\nu = |\Gamma(\mathbf{s}_t, \mathbf{u}_t) : \mathbf{u}_t \in \mathcal{L}_k|$ elementos, en donde $\Gamma(\cdot,\cdot)$ es la función de transición. Y puesto que en una retícula asociada a un código convolucional la función de transición es invariante en el tiempo, la relación $\sim_F$ divide a los 2^m estados de una etapa cualquiera en $2^m/\nu$ clases, de modo que, en cada clase, desde cualquiera de sus estados sólo existen transiciones hacia el mismo conjunto de estados en la etapa siguiente.

De manera análoga, podemos definir otra relación de equivalencia entre los estados de llegada, en la etapa $\mathcal{N}_{t+1}$, como

$$\begin{aligned} &\mathbf{s}_{t+1}^{(1)} \sim_B \mathbf{s}_{t+1}^{(2)} \Leftrightarrow \exists\, \mathbf{u}_t^{(1)}, \mathbf{u}_t^{(2)} \text{ y } \mathbf{s}_t \in \mathcal{N}_t : \\ &\mathbf{s}_{t+1}^{(1)} = \mathbf{u}_t^{(1)}\mathcal{A} + \mathbf{s}_t\mathcal{B}; \quad \mathbf{s}_{t+1}^{(2)} = \mathbf{u}_t^{(2)}\mathcal{A} + \mathbf{s}_t\mathcal{B} \end{aligned}$$

esto es, llamando a ambos estados $\sim_B$-equivalentes si son alcanzables desde un mismo estado en la etapa anterior. La relación de equivalencia $\sim_B$ define una partición del conjunto de estados en la etapa $\mathcal{N}_{t+1}$ con respecto a los estados de la etapa previa. Sin más que combinar las definiciones, es sencillo ver que existe una relación uno a uno entre las clases de equivalencia de $\sim_F$ y de $\sim_B$. En consecuencia, los estados de dos etapas consecutivas y las

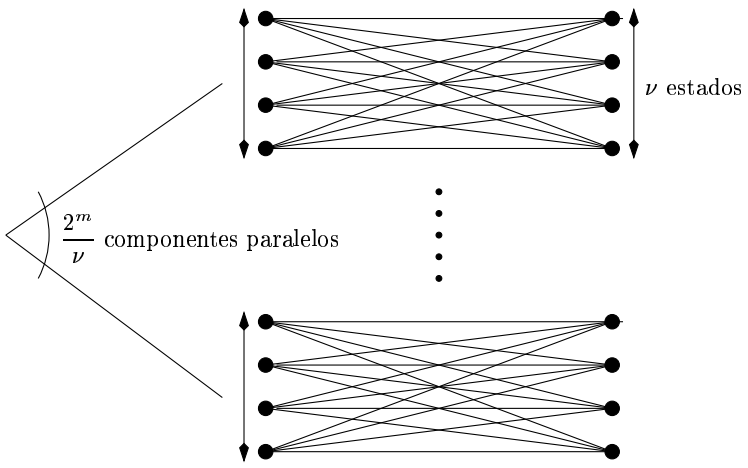


FIGURA 12.8. Estructura paralela de una retícula.

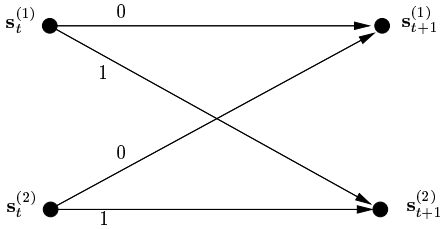


FIGURA 12.9. Propiedad de conectividad por emparejamiento de estados asociados.

transiciones entre ellos pueden ser divididos en $2^m/\nu$ componentes estructuralmente idénticas (paralelas), siendo cada componente un grafo completo bipartito. Las componentes difieren nada más que en las etiquetas asociadas a los nodos y a las transiciones. Esta estructura paralela genérica se ilustra en la figura 12.8.

En el caso particular de que el código sea $[n, 1]$, la descomposición en subgrafos paralelos implica las siguientes dos propiedades:

a) Propiedad de emparejamiento por pares de estados: si las transiciones de salida del estado $\mathbf{s}_t^{(1)}$ en la etapa t alcanzan los estados $\mathbf{s}_{t+1}^{(1)}$ y $\mathbf{s}_{t+1}^{(2)}$ en la etapa $t+1$, entonces estos dos estados también son alcanzables desde un mismo estado $\mathbf{s}_t^{(2)}$ en la etapa t.

b) Propiedad de accesibilidad restringida: el conjunto de 2^r estados alcanzables desde un estado de la etapa $t \geq m$ tras r transiciones también son alcanzables desde otros $2^r - 1$ estados en la etapa t.

La propiedad de accesibilidad restringida es una consecuencia simple de la aplicación iterativa de la propiedad de emparejamiento etapa por etapa. La propiedad de emparejamiento significa que, en una etapa cualquiera $t \geq m$, se pueden descomponer las transiciones del diagrama reticular en 2^{k-1} grupos de pares de transiciones independientes. Suponiendo, sin pérdida alguna de generalidad, que el codificador no es recurrente y que su vector de estado se compone sencillamente de los m últimos bits de entrada, la estructura de estos grupos de pares de transiciones es la indicada en la figura 12.9 con

$$\mathbf{s}_t^{(1)} = (u_{t-1}, u_{t-2}, \ldots, u_{t-m+1}, 0) \quad \mathbf{s}_{t+1}^{(1)} = (0, u_{t-1}, u_{t-2}, \ldots, u_{t-m+1})$$
$$\mathbf{s}_t^{(2)} = (u_{t-1}, u_{t-2}, \ldots, u_{t-m+1}, 1) \quad \mathbf{s}_{t+1}^{(2)} = (1, u_{t-1}, u_{t-2}, \ldots, u_{t-m+1})$$

y $(u_{t-1}, u_{t-2}, \ldots, u_{t-m+1})$ los $m-1$ últimos símbolos de entrada. Este paralelismo estructural en el diagrama reticular de un código convolucional redunda en algoritmos de decodificación más eficientes porque se puede aprovechar para realizar las operaciones de cálculo necesarias en paralelo.

Ejemplo 12.3. La figura 12.10, en la página siguiente, es el diagrama reticular del codificador convolucional binario $[3, 1]$ visto en el ejemplo 12.1. En cada etapa se han marcado sólo las transiciones que ocurren por primera vez, según el esquema u / $x_1x_2x_3$, con u el bit de entrada y $x_1x_2x_3$ los bits de salida. Observe que, en cada estado, a la entrada 0 le corresponde como salida la propia etiqueta del nodo del que diverge; y a la entrada 1, el complemento a 1 de esa etiqueta.

Un código convolucional de tasa $[n, 1]$ que, en cada estado, es tal que las secuencias de salida correspondientes a los símbolos de entrada 0 y 1 son

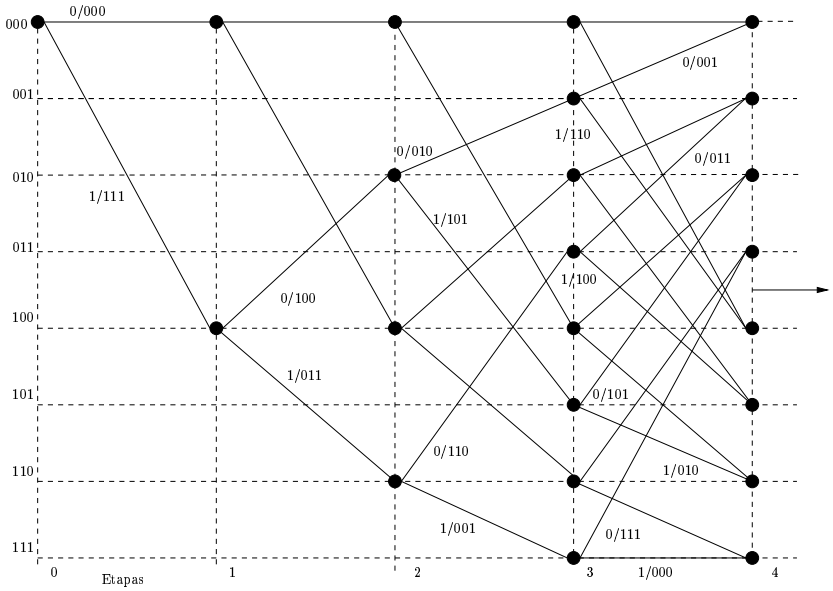


FIGURA 12.10. Diagrama reticular del código convolucional generado por $G(D) = (1 + D \;\; 1 + D^2 \;\; 1 + D^3)$.

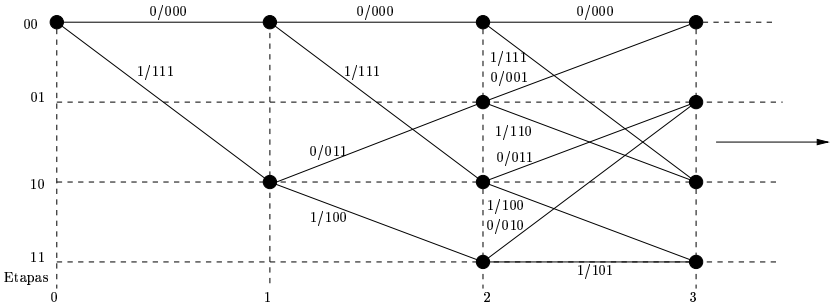


FIGURA 12.11. Diagrama reticular del codificador convolucional con implementación directa de $G(D) = (1 \;\; 1 + D \;\; 1 + D + D^2)$.

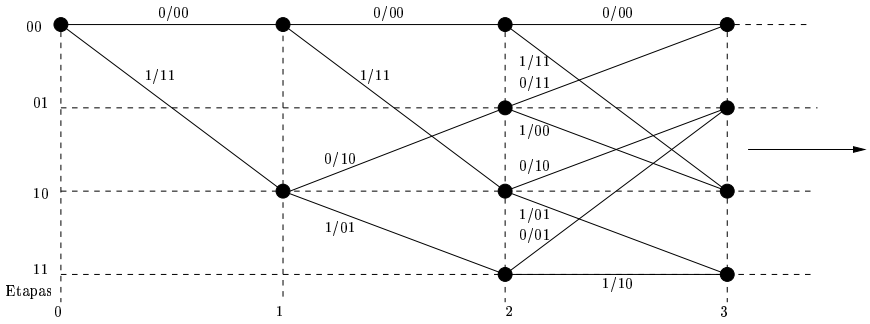


FIGURA 12.12. Diagrama reticular del codificador $G(D) = (1 + D + D^2 \;\; 1 + D^2)$.

una la inversa de la otra se llama antipodal. En términos algebraicos, si en el patrón generador en el dominio del tiempo es $G_0 = [1\ 1\ \ldots 1]$, entonces el código es antipodal. Los mejores códigos convolucionales $[n, 1]$ son antipodales, propiedad ésta que se justifica con los argumentos del apartado 12.6.

El mismo código admite como generador a

$$G'(D) = \begin{pmatrix} 1 & 1 + D & 1 + D + D^2 \end{pmatrix}.$$

El diagrama reticular correspondiente a $G'(D)$ (figura 12.11) es más simple, no tiene 8 sino 4 estados en cada etapa, porque el grado de $G'(D)$ es 2. Se verá en el apartado dedicado a la decodificación que el número de operaciones necesarias para decodificar es proporcional al número de estados del codificador, razón por la que conviene caracterizar un código convolucional con el diagrama reticular más sencillo posible, que es, como se sabe, el deducido de una matriz canónica. ■

EJEMPLO 12.4. El diagrama reticular del codificador convolucional dado en el ejemplo 12.2

$$G(D) = \begin{pmatrix} 1 + D + D^2 & 1 + D^2 \end{pmatrix}$$

consta de 4 estados y 8 transiciones, y se ha representado en la figura 12.12. Note que todas las ramas que terminan en un nodo están asociadas al mismo bit de entrada al codificador. En cuanto a las secuencias de salida, vea que cualquier par de bits aparece el mismo número de veces y que el código es antipodal.

Si se emplease el codificador equivalente sistemático y recurrente dado por

$$G'(D) = \begin{pmatrix} 1 & \dfrac{1 + D^2}{1 + D + D^2} \end{pmatrix}$$

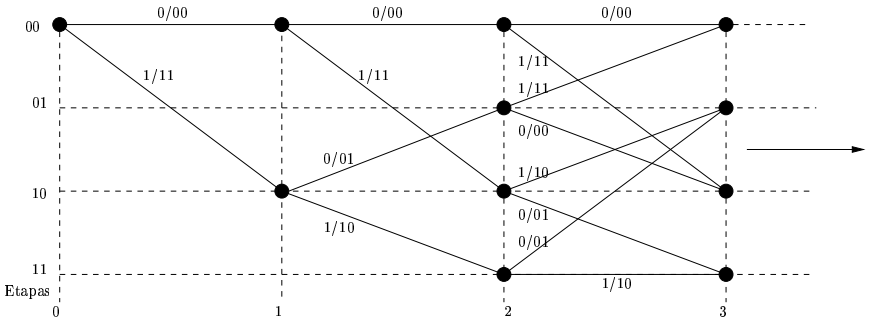


FIGURA 12.13. Diagrama reticular del codificador $G(D) = (1 \quad (1 + D^2)/(1 + D + D^2))$.

se tendría el diagrama reticular de la figura 12.13. La retícula es idéntica en estructura a la anterior pero, naturalmente, al haber cambiado la estructura interna del codificador varían también las etiquetas asignadas a los arcos. ■

12.3. Terminación de un código convolucional

En los sistemas de comunicaciones reales, la secuencia de símbolos de información que se introduce en un codificador convolucional es de longitud finita y produce una secuencia del código también finita al término del proceso de codificación. La correspondencia entre mensajes y secuencias del código de longitud finita es uno a uno, de manera que la terminación de un código convolucional convierte a éste en un código de bloques lineal.

Normalmente, al terminar de codificar una secuencia finita se desea que el codificador acabe en un estado interno conocido (a ser posible el estado nulo) para que la (de)codificación de las sucesivas secuencias finitas de símbolos de información sea independiente. Un método inmediato para finalizar un código convolucional es el de añadir al final de la secuencia de información tantos ceros como se precisen para devolver al codificador al estado $\mathbf{0}$. Así, en un codificador $[n, k]$ descrito por una matriz canónica en la que m sea el máximo grado de sus filas, si $\mathbf{u} = (\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{l-1})$ es la secuencia original de información, la entrada al codificador será $\mathbf{u}$ más la adición de m bloques de k ceros: $\mathbf{v} = (\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{l-1}, \mathbf{0}, \mathbf{0}, \ldots, \mathbf{0})$. La secuencia del código $\mathbf{x} = (\mathbf{x_0}, \ldots, \mathbf{x}_{l-1}, \mathbf{x}_l, \ldots, \mathbf{x}_{m+l-1})$ generada por $\mathbf{v}$ la forman $m+l$ grupos de n símbolos, siendo los m últimos los que corresponden a las m entradas nulas posteriores a $\mathbf{u}$. El código finito que resulta es, pues, un código lineal $[n(m + l), kl]$, cuya matriz generadora son los l primeros desplazamientos

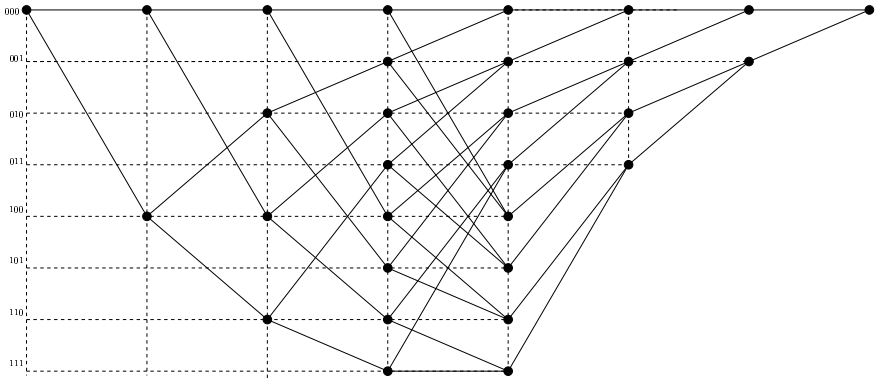


Figura 12.14. Diagrama reticular de un código convolucional terminado con ceros.

del patrón generador $G = [G_0\ G_1\ \ldots\ G_m]$ del código convolucional,

$$\mathcal{G} = \begin{pmatrix} G_0 & G_1 & \ldots & G_m & & & \\ & G_0 & G_1 & \ldots & G_m & & \\ & & \ddots & & & \ddots & \\ & & & G_0 & G_1 & \ldots & G_m \end{pmatrix}_{lk \times n(l+m)} .$$

Esta forma de terminar un código convolucional haciendo retornar al codificador al estado inicial conlleva una reducción de la tasa de codificación, que ahora pasará a valer $kl/n(m+l)$. El diagrama reticular de este código lineal es asimismo finito, consta de $m+l$ etapas y finaliza en el mismo estado de partida, el estado $\mathbf{0}$.

Ejemplo 12.5. El diagrama reticular de la figura 12.14 corresponde al código convolucional generado por el codificador de la figura 12.2 y terminado con 3 ceros tras 4 bits de información. El código terminado tiene, así, parámetros $[21, 4]$ y sus palabras del código son todas las secuencias de etiquetas de los 16 caminos existentes entre el estado inicial y el final. ■

Es posible evitar la reducción de tasa causada por la terminación con ceros utilizando la técnica de terminar un código convolucional por seccionamiento. El seccionamiento consiste en utilizar las secuencias del código definidas por $m+l$ secciones completas (con todos los estados y transiciones) consecutivas del diagrama reticular original. Más exactamente, en utilizar el código lineal definido por los caminos que comienzan en uno de los 2^s estados de la retícula original y terminan, después de $m+l$ transiciones, en el mismo estado que el inicial. La matriz generadora de este código, en el

dominio del tiempo, es [40]

$$\mathcal{G}^* = \begin{pmatrix} G_m & & & & G_0 & G_1 & \dots & G_{m-1} \\ G_{m-1} & G_m & & & & G_0 & G_1 & \dots \\ & \ddots & \ddots & & & & \ddots & \\ G_1 & G_2 & \dots & G_m & & & & G_0 \\ G_0 & G_1 & \dots & G_{m-1} & G_m & & & \\ & G_0 & G_1 & \dots & G_{m-1} & G_m & & \\ & & \ddots & & \ddots & \ddots & \ddots & \\ & & & G_0 & G_1 & \dots & G_{m-1} & G_m \end{pmatrix}$$

de dimensiones $(m+l)k \times (m+l)n$, siendo nulos todos los bloques no indicados. Si $\mathbf{u} = (\mathbf{u}_0, \mathbf{u}_2, \dots, \mathbf{u}_{m+l-1})$ es cualquier secuencia de $(m+l)k$ bits de entrada, la secuencia del código

$$\mathbf{x} = \mathbf{u} \cdot \mathcal{G}^* = \left(\sum_{i=0}^{m+l-1} \mathbf{u}_i G_{((m+j-i))} \right)_{j=0,\dots,m+l-1}$$

es la convolución periódica de $\mathbf{u}$ y $(G_0, G_1, \dots, G_m)$. Los paréntesis dobles en $((a))$ simbolizan el residuo módulo-$(m+1)$ del operando.

12.4. Decodificación

En un código convolucional, parece claro que una estimación óptima de la secuencia de símbolos emitida debe tener en cuenta todos los símbolos de la secuencia recibida, ya que por causa de la memoria del codificador cada uno de los símbolos de la secuencia que se observa a la salida del canal depende, posiblemente, de todos los anteriores a él. Así planteado, un método de decodificación óptimo consistirá, en esencia, en establecer una métrica probabilística con la que comparar cada una de las secuencias del código con la secuencia recibida. La secuencia del código más cercana (similar) a la detectada constituirá la elección del decodificador. Por tanto, el decodificador óptimo de un código convolucional es un estimador de secuencias de símbolos, no de símbolos individuales ni tampoco de n-tuplas de símbolos. Sin embargo, carece de sentido realizar una búsqueda exhaustiva en el espacio de secuencias del código con el fin de decidir la óptima, ya que sería preciso consumir un tiempo que aumenta de forma exponencial con la longitud de la secuencia. Interesan, por tanto, algoritmos de decodificación no sólo óptimos para minimizar la probabilidad de error, sino también eficientes. Entenderemos por eficientes aquellos algoritmos con una complejidad polinómica en lugar de exponencial.

Los primeros algoritmos de decodificación ideados para códigos convolucionales (el algoritmo de Wozencraft, el de Fano o el algoritmo de pila, ver [39]) sí eran de decodificación símbolo a símbolo, dado que hasta 1967 no se conocía ninguna manera eficiente de realizar la decodificación óptima. En este apartado se va a explicar precisamente esta realización eficiente, el denominado algoritmo de Viterbi. Veremos que la clave de un algoritmo con estas características es que la correlación presente entre los sucesivos grupos de n símbolos detectados va a permitir al decodificador desechar de inmediato un gran número de secuencias del código candidatas.

Las cuestiones relativas a la decodificación se van a descomponer, por claridad, en tres etapas:

a) Se va a describir el algoritmo de Viterbi. En su forma más general, éste es un algoritmo para calcular flujos en un grafo.

b) Se va a deducir el método óptimo de decodificación de secuencias de un código convolucional. El criterio de optimización elegido será el de maximizar la probabilidad de estimar sin error la secuencia del código emitida.

c) Se va a aplicar el algoritmo de Viterbi al problema de la decodificación de códigos convolucionales caracterizados por su diagrama reticular.

12.4.1. El algoritmo de Viterbi

A. J. Viterbi ideó en 1967 un algoritmo asintóticamente óptimo de decodificación de códigos convolucionales en canales ruidosos sin memoria. Fue G. D. Forney quien, más tarde (en 1973 [21]), consiguió probar que este algoritmo proporcionaba la decodificación óptima ML de sucesiones de símbolos y que, en su enunciado más amplio, era la solución al problema de la estimación ML de secuencias de estados en un proceso de Markov de estados finitos discreto en el tiempo alterado por un proceso de ruido sin memoria.

Pero el algoritmo de Viterbi se puede formular de un modo muy general, desprovisto del significado propio de un problema de decisión o decodificación. Es una aplicación del método matemático de la programación dinámica al problema del cálculo de flujos en un grafo reticular, y como tal va a ser descrito a continuación. La formulación abstracta que sigue es obra de R. J. McEliece.

Sea $\mathcal{T}$ una retícula cuyos arcos están etiquetados con elementos de un cierto alfabeto $\mathcal{E}$. La etiqueta de un arco a se representará por $\alpha(a)$. Supóngase que $\mathcal{E}$ es un conjunto que cuenta con dos operaciones binarias internas, que simbolizaremos con $\cdot$ (producto) y $+$ (suma), y que satisfacen los siguientes axiomas:

a) El producto es asociativo y existe un elemento identidad 1, tal que $1 \cdot a = a \cdot 1 = a$ para cualquier $a \in \mathcal{E}$. Dicho de otro modo, $(\mathcal{E}, \cdot)$ es un monoide o un semigrupo unitario.

b) La suma es asociativa, conmutativa y existe un elemento identidad 0 tal que $0 + a = a + 0 = a$ para todo $a \in \mathcal{E}$. $(\mathcal{E}, +)$ es un monoide conmutativo o semigrupo abeliano unitario.

c) El producto es distributivo por la derecha respecto de la suma: $\forall a, b, c \in \mathcal{E}, (a + b)c = ac + bc$.

La estructura algebraica $(\mathcal{E}, +, \cdot)$ es un *semianillo* o *dioide*. Tal y como se ha definido, la única diferencia entre un semianillo y un anillo es que en el primero no existe necesariamente operación inversa para la suma. Observe que el producto no siempre es conmutativo, por lo que el orden de los operandos puede ser significativo al multiplicar.

Definición 12.6 (Flujo). *Si $p = a_1 a_2 \cdots a_n$ es un camino en la retícula $\mathcal{T}$, compuesto por los arcos a_1, a_2, ..., a_n, se llama flujo de p al producto ordenado $\mu(p) = \alpha(a_1) \cdot \alpha(a_2) \cdots \alpha(a_n)$.*

Definición 12.7 (Flujo entre dos nodos). *En la retícula $\mathcal{T}$, se define el flujo entre dos nodos $\mathbf{s}_i$ y $\mathbf{s}_f$, $\mu(\mathbf{s}_i, \mathbf{s}_f)$, como la suma de los flujos de todos los caminos con origen en $\mathbf{s}_i$ y final en $\mathbf{s}_f$. Por convenio, si los nodos $\mathbf{s}_i$ y $\mathbf{s}_f$ no están conectados, $\mu(\mathbf{s}_i, \mathbf{s}_f) = 0$, y para cualquier nodo $\mathbf{s}$ se postula $\mu(\mathbf{s}, \mathbf{s}) = 1$.*

El concepto de flujo es abstracto, pero su significado se vuelve aparente si consideramos dos sencillos ejemplos.

Ejemplo 12.6. Identifiquemos $\mathcal{E} = \mathbb{F}_2$ y la suma y el producto con las operaciones booleanas or y and. Cada arco de una retícula $\mathcal{T}$ etiquetado con un 1 se puede interpretar como activo; y cada arco con etiqueta 0, como inactivo. Con este convenio, $\mu(p) = 1$ sólo y cuando el camino p no posee arcos inactivos; en otro caso, $\mu(p) = 0$; y $\mu(\mathbf{s}_i, \mathbf{s}_f) = 1$ si y sólo si existe algún camino desde $\mathbf{s}_i$ hasta $\mathbf{s}_f$ formado por arcos activos. El flujo, en este caso, es indicador de la conectividad entre pares ordenados de estados. ■

Ejemplo 12.7 (El álgebra $(\mathbb{R}^+ \cup \{\infty\}, \text{mín}, +)$). Sea $\mathcal{E}$ el conjunto de los números reales no negativos más el símbolo ∞. En $\mathcal{E}$ se define el producto de dos de sus elementos como la suma ordinaria de números reales, con la salvedad de reservar el símbolo 0 para el elemento identidad. La suma de dos elementos de $\mathcal{E}$ se define como el mínimo de ambos, e ∞ desempeña el papel de elemento identidad, es decir, $a + \infty = \text{mín}\{a, \infty\} = a$. Vea que

$(\mathbb{R}^+ \cup \{\infty\}, \text{mín}, +)$ es efectivamente un semianillo, en el que el producto (que es la suma ordinaria de números reales) es conmutativo. Si el número real que etiqueta a cada arco se ve como su coste, entonces $\mu(\mathbf{s}_i, \mathbf{s}_f)$ es el coste del camino de mínimo coste entre $\mathbf{s}_i$ y $\mathbf{s}_f$:

$$\mu(\mathbf{s}_i, \mathbf{s}_f) = \sum_{p \in P(\mathbf{s}_i, \mathbf{s}_f)} \mu(p) = \min_{p \in P(\mathbf{s}_i, \mathbf{s}_f)} \mu(p)$$

con $P(\mathbf{s}_i, \mathbf{s}_f)$ el conjunto de caminos desde $\mathbf{s}_i$ a $\mathbf{s}_f$. Si no existe ningún camino entre estos nodos, se conviene en identificar el flujo como el elemento identidad para la suma, es decir, $\mu(\mathbf{s}_i, \mathbf{s}_f) = \infty$.

El álgebra $(\mathbb{R}^+ \cup \{\infty\}, \text{mín}, +)$ juega un importante papel no sólo en la decodificación de códigos convolucionales, como enseguida veremos, sino también en áreas como la teoría de sistemas discretos. ■

Pues bien, el algoritmo de Viterbi es una simple ecuación recurrente para calcular los flujos $\mu(p)$ en una retícula sin tener que enumerar todos los caminos.

Teorema 12.1 (Algoritmo de Viterbi). *En cualquier retícula $\mathcal{T}$, el flujo desde el nodo raíz $\mathbf{0}$ hasta el nodo $\mathbf{s}$ es*

$$\mu(\mathbf{s}) = \sum_{a : a\circ = \mathbf{s}} \mu(\circ a)\alpha(a). \tag{12.9}$$

Demostración. El enunciado se prueba por inducción. Así, para el nodo raíz tenemos, por definición, $\mu(\mathbf{0}) = 1$; y para cualquier nodo $\mathbf{v}$ en la etapa 1 de $\mathcal{T}$,

$$\mu(\mathbf{v}) = \sum_{a : a\circ = \mathbf{v}} \mu(\circ a)\alpha(a) = \sum_{a : \mathbf{0} \to \mathbf{v}} \mu(\mathbf{0})\alpha(a) = \sum_{a : \mathbf{0} \to \mathbf{v}} 1 \cdot \alpha(a) = \sum_{a : \mathbf{0} \to \mathbf{v}} \alpha(a)$$

lo que evidentemente es cierto. Supongamos ahora que el teorema se cumple para cualquier nodo en la etapa t de la retícula. Sea $\mathbf{w}$ un nodo cualquiera en la etapa $t + 1$. En la fórmula

$$\mu = \sum_{a : a\circ = \mathbf{w}} \mu(\circ a) \cdot \alpha(a)$$

$\circ a$ es un nodo de la etapa t de la retícula. Haciendo uso de la hipótesis de inducción

$$\mu(\circ a) = \sum_{p \in P_{\circ a}} \mu(p)$$

en donde $P_{\circ a}$ es el conjunto de caminos que acaban en $\circ a$. Por tanto,

$$\mu = \sum_{a : a\circ = \mathbf{w}} \mu(\circ a) \cdot \alpha(a) = \sum_{a : a\circ = \mathbf{w}} \sum_{p \in P_{\circ a}} \mu(p) \cdot \alpha(a)$$

recurriendo a las propiedades conmutativa (de la suma) y distributiva (del producto). Pero $\mu(p) \cdot \alpha(a)$ es el flujo del camino p extendido por el arco a. Y como cualquier camino que finalice en el nodo $\mathbf{w}$ se puede construir añadiendo un arco a algún camino que termine en un nodo de la etapa t —por la propia definición de $\mathcal{T}$—

$$\mu = \sum_{p \in P_{\mathbf{w}}} \mu(p) = \mu(\mathbf{w})$$

que es precisamente la definición del flujo desde $\mathbf{0}$ a $\mathbf{w}$. ▶

La fórmula (12.9) del algoritmo de Viterbi toma la forma de una operación de convolución en el semianillo $(\mathcal{E}, +, \cdot)$. Expresa que para hallar el flujo hasta un nodo cualquiera de la etapa $t+1$ de una retícula es suficiente con conocer los flujos hasta todos y cada uno de los nodos de la etapa t y las etiquetas de los arcos entre la etapa t y la $t+1$. Es fácil generalizar este argumento y mostrar que para calcular un flujo se puede partir una retícula por una etapa intermedia $\mathcal{N}_j$ y calcular por separado los flujos de cada una de las dos retículas

$$\mu(\mathbf{s}_i, \mathbf{s}_f) = \sum_{\mathbf{s}_\star \in \mathcal{N}_j} \mu(\mathbf{s}_i, \mathbf{s}_\star)\mu(\mathbf{s}_\star, \mathbf{s}_f).$$

Asimismo, debe quedar claro que la naturaleza recurrente del algoritmo de Viterbi es una consecuencia de la estructura reticular (en etapas) de un grafo, y que la única condición necesaria para poder aplicarlo es que sea posible encontrar en el grafo una partición $\mathcal{N}_1, \ldots, \mathcal{N}_s, \ldots$ de sus nodos tal que todos los arcos que egresen de nodos en la clase $\mathcal{N}_i$ terminen en nodos de $\mathcal{N}_{i+1}$. La composición de cada etapa y el conjunto de arcos entre los nodos de etapas consecutivas pueden ser completamente arbitrarios.

Finalmente, note que el estado $\mathbf{0}$ sólo desempeña en el algoritmo de Viterbi el papel de estado de referencia, en relación al cual se calculan los flujos. Es decir, en realidad el algoritmo permite calcular el flujo entre dos estados cualesquiera del diagrama reticular. En particular, si la operación producto en $\mathcal{E}$ es conmutativa como en los ejemplos 12.6 y 12.7, la recurrencia de Viterbi también puede aplicarse hacia atrás.

Ejemplo 12.8. El algoritmo de Viterbi, caracterizado en la forma abstracta que se ha dado en este apartado, puede particularizarse para varios casos interesantes:

a) **Cálculo de la distancia de un código lineal**. Como se ha apuntado anteriormente, cualquier código de bloques puede representarse con un grafo reticular finito, y esto trae consigo como ventaja la posibilidad de emplear también con los códigos de bloques algoritmos

probabilísticos eficientes de decodificación basados en el algoritmo de Viterbi. A modo de curiosidad, el algoritmo de Viterbi permite obtener la distancia del código directamente a partir del grafo. Sea $\mathcal{T}$ una retícula que representa un código lineal $\mathcal{C}$, convolucional o de bloques. Utilizando el algoritmo de Viterbi sobre el álgebra $(\mathbb{R}_+^{\mathrm{n}}, \mathrm{mín}, +)$ —que es simplemente la extensión natural a vectores de longitud n del agebra $(\mathbb{R}^+ \cup \{\infty\}, \mathrm{mín}, +)$, sin más que realizar las operaciones por componentes de los vectores— se puede obtener la distancia del código. Sólo es necesario tener la precaución de excluir del cálculo del flujo el vector nulo, lo que puede lograrse con el siguiente reetiquetado:

- Cualquier arco de $\mathcal{T}$ que una dos estados nulos y finalice en la etapa i se etiqueta con el vector $(0, 0, \ldots, \infty, 0, \ldots, 0)$, donde ∞ se sitúa en la componente i.
- Cualquier otro arco se marca con el vector $\mathbf{0}$ o $\mathbf{1}$, según produzca el bit de salida 0 o 1, respectivamente.

El mínimo de las componentes del vector de flujo es la distancia del código.

b) **Enumeración de las palabras del código**. Es fácil observar que el algoritmo de Viterbi es en lo fundamental una técnica recurrente para enumerar los caminos de un grafo con estructura regular. Como aplicación inmediata de este principio, veamos cómo se adaptaría para enumerar las palabras de un código definido por la retícula $\mathcal{T}$. Aquí el álgebra a emplear sería $(\mathcal{F}^\infty, \cup, \circ)$, en donde $\mathcal{F}$ es el alfabeto de símbolos del código (que no precisa de ninguna estructura algebraica propia); $\mathcal{F}^\infty = \cup_{i=1}^{\infty} \mathcal{F}^i$ es el conjunto de secuencias que se pueden escribir con el alfabeto $\mathcal{F}$, completado con la cadena vacía; la operación de suma es la unión de conjuntos, con elemento neutro igual al conjunto vacío; y la operación de multiplicación, $\circ$, es la concatenación de dos elementos de $\mathcal{F}^\infty$, cuyo elemento identidad es la cadena vacía. Resulta claro en este ejemplo que el flujo de un camino es simplemente la palabra del código que el camino define. ■

12.4.2. Decodificación ML de secuencias de símbolos

Pasemos ahora a analizar qué criterio de elección debe utilizar el decodificador para conseguir hacer mínima la probabilidad de error.

Sea $\mathbf{u} = (\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1})$ una secuencia de N bloques de símbolos de entrada a un codificador convolucional, con N tan grande como se quiera, y $\mathbf{x} = (\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{N-1})$ la secuencia unívoca de vectores de salida que produce la excitación $\mathbf{u}$. Sea, por último, $\mathbf{r} = (\mathbf{r}_0, \mathbf{r}_1, \ldots, \mathbf{r}_{N-1})$ la secuencia

de símbolos que se observa a la salida del canal discreto.[6] Así planteado, el problema de la decodificación óptima consiste en decidir, para cada observación $\mathbf{r} = (\mathbf{r}_0, \mathbf{r}_1, \ldots, \mathbf{r}_{N-1})$, qué secuencia $\hat{\mathbf{x}}$ emitida hace mínima la probabilidad de error $\mathrm{P}(\hat{\mathbf{x}} \neq \mathbf{x})$. Aplicando directamente los resultados del apartado 5.A, es el caso que la estrategia que minimiza la probabilidad de equivocación $\mathrm{P}(\hat{\mathbf{x}} \neq \mathbf{x})$ en el receptor, cuando la secuencia recibida es $\mathbf{r}$, consiste en seleccionar aquella secuencia del código que hace máxima la probabilidad condicional $\mathrm{P}(\hat{\mathbf{x}} \,|\, \mathbf{r})$ de haber emitido $\hat{\mathbf{x}}$. Esto es, se minimiza el error si se toma la decisión MAP (máximo a posteriori):

$$\mathrm{MAP}(\mathbf{r}) = \arg \max_{\hat{\mathbf{x}}} \mathrm{P}(\hat{\mathbf{x}} \,|\, \mathbf{r}) = \arg \max_{\hat{\mathbf{x}}} \log \mathrm{P}(\hat{\mathbf{x}} \,|\, \mathbf{r}).$$

Si se supone que las secuencias de símbolos de información (y por tanto las secuencias del código) son equiprobables, el criterio MAP se simplifica y transforma en el criterio ML (de máxima verosimilitud[7]):

$$\mathrm{ML}(\mathbf{r}) = \arg \max_{\hat{\mathbf{x}}} \log \mathrm{P}(\mathbf{r} \,|\, \hat{\mathbf{x}})$$

de selección de la secuencia $\hat{\mathbf{x}}$ que, a priori (sin disponer de información acerca de los estadísticos de $\hat{\mathbf{x}}$), hace que la observación de la trayectoria $\mathbf{r}$ tenga probabilidad máxima.

La probabilidad de observar la secuencia $\mathbf{r}$ cuando se ha transmitido $\hat{\mathbf{x}}$ es, en un canal discreto y sin memoria,

$$\mathrm{P}(\mathbf{r} \,|\, \hat{\mathbf{x}}) = \prod_{i=0}^{N-1} \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i)$$

[6] Tratamos, por tanto, con observaciones discretas y con ruido de las secuencias del código. Es habitual hablar de decodificación *hard* cuando la entrada al decisor es una secuencia de símbolos; cuando la entrada al decisor es una sucesión de muestras de la señal que sirve de soporte a la transmisión del mensaje (o, con más propiedad, una secuencia de valores reales que constituyen un estadístico suficiente de la secuencia de símbolos emitida) se habla, en cambio, de decodificación *soft*. Las técnicas de decodificación *soft* consiguen menor probabilidad de error que las *hard*, aunque son ligeramente más complicadas de implementar, y no serán consideradas en este texto, en el que se parte de la idealización de canal discreto. En todo caso, el algoritmo óptimo de decodificación *hard* y *soft* de secuencias es el mismo, salvo por un cambio en la métrica.

[7] En cualquier experimento aleatorio, dadas dos alternativas posibles a priori, $\hat{\mathbf{x}}_\mathbf{1}$ y $\hat{\mathbf{x}}_\mathbf{2}$, y una observación ruidosa del resultado, $\mathbf{r}$, la función

$$f(\mathbf{r}) = \log \frac{\mathrm{P}(\mathbf{r} \,|\, \hat{\mathbf{x}}_\mathbf{1})}{\mathrm{P}(\mathbf{r} \,|\, \hat{\mathbf{x}}_\mathbf{2})}$$

es una medida de la posibilidad de que haya ocurrido $\hat{\mathbf{x}}_\mathbf{1}$ y no $\hat{\mathbf{x}}_\mathbf{2}$; más exactamente, si $f(\mathbf{r}) > 0$, es más probable la hipótesis $\hat{\mathbf{x}}_\mathbf{1}$ que la hipótesis $\hat{\mathbf{x}}_\mathbf{2}$. De ahí, por tanto, que la función $f(\mathbf{r})$ (cuyo signo permite discriminar entre dos alternativas excluyentes) se conozca también como función de verosimilitud.

en donde $\mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i)$ indica la probabilidad de observar a la salida del canal el vector $\mathbf{r}_i$ en respuesta a la entrada $\hat{\mathbf{x}}_i$. Tomando logaritmos en ambos miembros de esta ecuación, la regla de decodificación ML queda, por ser la función logaritmo una función estrictamente creciente, como

$$\begin{aligned} \arg\max_{\hat{\mathbf{x}}} \log \mathrm{P}(\mathbf{r} \,|\, \hat{\mathbf{x}}) &= \arg\max_{\hat{\mathbf{x}}} \sum_{i=0}^{N-1} \log \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i) \\ &= \arg\min_{\hat{\mathbf{x}}} \sum_{i=0}^{N-1} -\log \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i). \end{aligned} \tag{12.10}$$

La función $\sum_{i=0}^{N-1} -\log \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i)$ es claramente una métrica entre los caminos $\mathbf{r}$ y $\hat{\mathbf{x}}$, la resultante de acumular un coste individual de valor igual a $-\log \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i)$ por cada arco del camino. En consecuencia, puede enunciarse el principio de decodificación óptima ML diciendo que consiste en la elección del camino de coste mínimo con respecto a la secuencia observada a la salida del canal, siendo el coste de un camino la suma del coste relativo de cada uno de sus arcos, y éste el logaritmo cambiado de signo de cierta probabilidad de transición del canal.

Observe en la ecuación (12.10) que la minimización de la métrica no se hace en cada arco individual, sino que es global, a lo largo de todo el camino candidato hasta la etapa N. Es por este motivo por el que la técnica recibe el nombre de detección ML de secuencias o detección MLSE (*Maximum Likelihood Sequence Estimation*). Obsérvese, además, que la métrica es puramente probabilística, es decir, que en absoluto depende de ninguna propiedad algebraica o estructural de las secuencias que se pretenden estimar.

Cuando la transmisión se realiza por un canal binario simétrico, las ecuaciones (12.10) adoptan una forma particularmente simple. Supóngase que la probabilidad de transición es p y que, en la rama $i = 0, \ldots, N-1$, los símbolos de salida de la secuencia $\mathbf{r}$ (fija) y el camino $\hat{\mathbf{x}} \in \mathcal{T}$ (arbitrario) difieren en $d_H(\hat{\mathbf{x}}_i, \mathbf{r}_i)$ posiciones. Por lo tanto,

$$-\log \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i) = -\log p^{d_H(\mathbf{r}_i, \hat{\mathbf{x}}_i)} (1-p)^{n - d_H(\mathbf{r}_i, \hat{\mathbf{x}}_i)}$$

que es una función afín creciente de $d_H(\mathbf{r}_i, \hat{\mathbf{x}}_i)$. Así pues, en un canal binario simétrico se selecciona el camino más cercano en distancia Hamming a la secuencia de salida del canal.

12.4.3. Decodificación de códigos convolucionales

Fue la intuición de Viterbi la que le hizo percatarse de que la ecuación (12.10) podía escribirse en la forma recurrente dada por el teorema 12.1,

el algoritmo de Viterbi. En el diagrama reticular $\mathcal{T}$ del código convolucional, supongamos que cada arco está etiquetado con la secuencia de símbolos de salida que emite el decodificador al realizar esa transición. Para cualquiera de sus arcos, a, cambiemos su etiqueta aplicando la transformación

$$L : \alpha(a) \longrightarrow \beta(a) = -\log \mathrm{P}(\mathbf{r}_i \,|\, \hat{\mathbf{x}}_i).$$

Esta transformación tiene como efecto convertir el alfabeto de etiquetas de $\mathcal{T}$ en el semianillo $\mathcal{S} = (\mathbb{R}^+ \cup \{\infty\}, \text{mín}, +)$ presentado en el ejemplo 12.7. De forma que la regla de decodificación ML queda como

$$\arg \min_{p=a_1 \ldots a_N \in \mathcal{T}_N} \sum_{i=0}^{N-1} \beta(a_i),$$

la minimización de $\sum_i \beta(a_i)$ sobre todos los caminos definidos por las N primeras secciones de la retícula $\mathcal{T}$ modificada. Recordando que en $\mathcal{S}$ la operación de sumar números reales se denominaba producto, lo anterior corresponde simbólicamente a

$$\arg \min_{p=a_1 \ldots a_N \in \mathcal{T}_N} \mu(p),$$

es decir, a elegir el camino de mínimo flujo (coste) de la retícula. La secuencia del código que define este camino se puede reconstruir concatenando los símbolos de salida de cada uno de sus arcos.

Consideremos todos los caminos que terminan en un estado $\mathbf{s}_i$ de la etapa i. A la hora de calcular caminos de coste mínimo hasta la etapa $i+1$ que pasen por el estado $\mathbf{s}_i$, es suficiente con memorizar sólo el camino p^* de menor medida de todos los que terminan en $\mathbf{s}_i$, porque es obvio que si se sumase el coste de una rama de salida de $\mathbf{s}_i$ a otro camino que no fuese el óptimo se obtendría siempre un camino de mayor longitud que si se hubiese sumado el coste de la misma rama a p^*. De donde se deduce, por lo tanto, que para cada etapa del diagrama reticular es suficiente con almacenar un solo camino óptimo desde $\mathbf{0}$ a cada uno de sus estados. Llamamos a este camino el *superviviente*, siendo así que el número de supervivientes en cada etapa será igual al número de estados posibles en la misma. En caso de que la retícula conste de un número finito de etapas y finalice en un único estado, sólo sobrevive un camino. Pero conviene advertir que los caminos candidatos $\hat{\mathbf{x}}$ y el camino recibido $\mathbf{r}$ no tienen por qué terminar en el mismo estado del diagrama reticular si éste, en la etapa N, consta de más de un estado alcanzable.

Formalicemos el algoritmo de Viterbi, con el propósito de mostrar toda su simplicidad. A tal efecto, supongamos una retícula $\mathcal{T}$ de N etapas, con origen en el estado $\mathbf{0}$ y fin en el estado $\mathbf{s}_\infty$. Sea $\mathcal{N}_i$ el conjunto de nodos de la etapa i. El algoritmo de Viterbi consta de los siguientes pasos:

1. Inicialización: fijar $\mu(\mathbf{0}) = 0$.

2. Iteración: para $i = 1, \ldots, N$ y para cualquier nodo $\mathbf{v} \in \mathcal{N}_i$

$$\mu(\mathbf{v}) = \min_{a:a\circ=\mathbf{v}} \{\mu(\circ a) + \beta(a)\}\,; \tag{12.11}$$

$$s(\mathbf{v}) = \arg \min_{a:a\circ=\mathbf{v}} \{\mu(\circ a) + \beta(a)\}\,. \tag{12.12}$$

3. Reconstruir la secuencia del código; para ello, fijar $\mathbf{v} = \mathbf{s}_\infty$ y, para $i = N, \ldots, 1$,

$$c_i = \alpha(s(\mathbf{v})); \qquad \mathbf{v} = \circ s(\mathbf{v}).$$

4. Estimar como secuencia del código transmitida $(c_1, c_2, \ldots, c_N)$.

La operación de suma que aparece en las ecuaciones (12.11)-(12.12) es la suma ordinaria entre números reales. Por otro lado, para cada nodo $\mathbf{v}$, la variable $s(\mathbf{v})$ sólo tiene como propósito guardar el último arco del camino superviviente hasta $\mathbf{v}$.

En caso de que el mínimo en la expresión (12.11) se alcanzase para dos o más arcos, note que la elección entre ellos es indistinta desde el punto de vista de la detección de la secuencia del código transmitida, pero no en lo referente al mensaje; pues los símbolos de entrada al codificador que generan los caminos respectivos pueden diferenciarse en un número variable de posiciones de los símbolos de información transmitidos. Es decir, caminos en la retícula con el mismo coste relativo respecto de la secuencia recibida pueden haber sido generados por mensajes diferentes. Por lo tanto, la decodificación ML de secuencias del código no minimiza necesariamente la probabilidad de error en los símbolos de información. Sin embargo, las simulaciones numéricas muestran que únicamente aparece una diferencia apreciable cuando la probabilidad de error en la transmisión de un símbolo es elevada.

La evaluación de las ecuaciones (12.11) puede dividirse, para cada estado de una etapa t, en tres fases: (1) suma del coste de las ramas incidentes en ese estado a los caminos supervivientes que terminan en los estados predecesores; (2) comparación del coste de los caminos supervivientes así extendidos; (3) selección del camino extendido de menor coste. La unidad básica de cómputo de un decodificador Viterbi es, entonces, un circuito aritmético–lógico de suma, comparación y selección.

La complejidad del algoritmo de Viterbi es absolutamente independiente del marco de aplicación. La ecuación (12.11) se ejecuta para cada nodo de la retícula (excepto el nodo raíz), y requiere tantas multiplicaciones en $\mathcal{S}$ como arcos de entrada al nodo; la selección del superviviente hasta ese nodo es equivalente a la realización de $g-1$ sumas en $\mathcal{S}$, siendo g el número de arcos incidentes en el nodo. Por lo tanto, la ejecución completa del algoritmo de

Viterbi sobre un grafo reticular con P nodos (en total) y A arcos (en total) precisa de A multiplicaciones y $A-P+1$ sumas en el conjunto algebraico $\mathcal{S}$. En consecuencia, su complejidad aumenta sólo linealmente con el número de etapas (estados) de la retícula, en lugar de incrementarse de manera exponencial como lo hace el número total de caminos.

En un sistema real de comunicaciones, N, la longitud en bloques del mensaje, puede alcanzar valores muy grandes, de forma que, en primer lugar, la decisión en el receptor se retrasaría hasta ser inaceptable y, en segundo lugar, la memoria necesaria para almacenar los caminos supervivientes, que es directamente proporcional a N, crecería también hasta hacer poco económica o incluso inviable la construcción del decodificador. Existen varias alternativas para solucionar estos inconvenientes:

a) Se puede utilizar un código convolucional truncado de longitud nN símbolos, por ejemplo recurriendo a la terminación con ceros o a la técnica de seccionamiento. En el diagrama reticular finito que corresponde al código terminado con ceros, en la última etapa sólo hay un estado y, por lo tanto, sobrevive un único camino hasta él.

b) Si todos los caminos supervivientes hasta la etapa t coinciden en sus q primeras transiciones (es decir, si los q primeros estados intermedios de cada camino son idénticos), entonces el decodificador puede tomar una decisión sobre los q primeros bloques de símbolos del mensaje. Parece claro, sin embargo, que una condición así puede no suceder con suficiente frecuencia y exige además que el decodificador disponga de la capacidad de explorar prefijos en los caminos supervivientes hasta una profundidad arbitraria.

c) Se puede forzar periódicamente, cada I intervalos de tiempo, una decisión sobre los kI símbolos de información transmitidos; basta para ello con elegir el camino superviviente de menor coste cada I transiciones. Se reconocerá, en cualquier caso, que éste es un procedimiento de decodificación subóptimo.

d) Se puede purgar periódicamente el conjunto de caminos supervivientes memorizados si se desechan todos aquéllos cuya medida difiera de la del mejor camino superviviente en una cantidad mayor que un umbral prefijado. Aunque mejor que el método anterior, porque las decisiones no se realizan a intervalos regulares, también éste es un procedimiento subóptimo.

Ejemplo 12.9. Supongamos que se utiliza sobre cierto canal binario simétrico el codificador convolucional de cuatro estados de la figura 12.15.

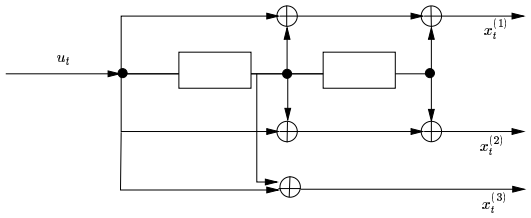


Figura 12.15. Codificador convolucional $[3, 1, 2]$.

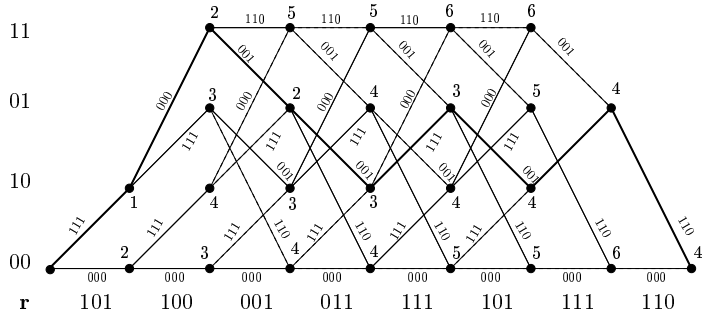


Figura 12.16. Algoritmo de Viterbi para el ejemplo 12.9.

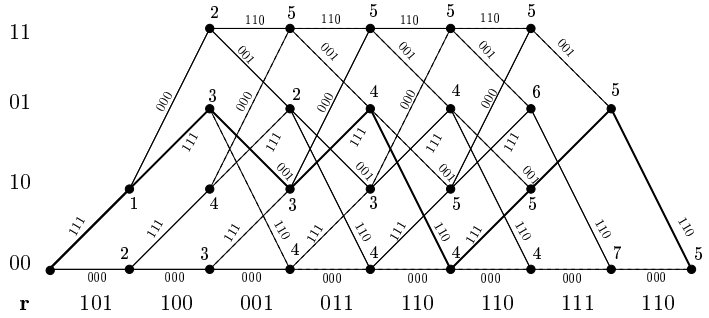


Figura 12.17. Algoritmo de Viterbi para el ejemplo 12.9.

La secuencia de información $\mathbf{u} = 110101$ produce la secuencia del código

$$\mathbf{x} = 111\ 000\ 001\ 001\ 111\ 001\ 111\ 110.$$

Imaginemos que se recibe la secuencia

$$\mathbf{r}_1 = 1\mathbf{0}1\ \mathbf{1}00\ 001\ 0\mathbf{1}1\ 111\ \mathbf{1}01\ 111\ 110$$

que contiene 4 símbolos erróneos, marcados con letra negrita.

En la figura 12.16 se muestra el resultado del algoritmo de decodificación de Viterbi. Cada nodo se ha etiquetado con el coste del camino superviviente hasta él, utilizando como medida de cada rama la distancia Hamming entre la secuencia recibida y los símbolos de salida del diagrama reticular. Los caminos que no sobreviven se han señalado con líneas discontinuas, y el camino final seleccionado es el marcado con trazo grueso. El camino ML coincide con la secuencia del código transmitida.

Si la secuencia recibida hubiese sido

$$\mathbf{r}_2 = 1\mathbf{0}1\ \mathbf{1}00\ 001\ 0\mathbf{1}1\ 11\mathbf{0}\ \mathbf{11}0\ 111\ 110$$

la ejecución del algoritmo de Viterbi habría seguido la trayectoria de caminos supervivientes representada ahora en la figura 12.17. En este caso, en la etapa 6 no sobrevive el camino correcto y la decisión ML del decodificador no coincide con la secuencia del código transmitida. ■

12.5. Enumeración de caminos

Tomando como punto de partida el algoritmo de decodificación de secuencias, parece lógico pensar que la capacidad de detectar y corregir errores con un código convolucional va a depender de la distancia entre dos secuencias del código cualesquiera o, lo que es igual, del coste relativo entre dos caminos en el diagrama reticular.

En diagramas simples, con pocos estados y transiciones, el coste relativo de un camino puede hallarse sin dificultad por observación. Pero, desde luego, no es éste un método aconsejable cuando se manejan diagramas reticulares complejos. En todos los casos, la obtención del camino a distancia mínima de uno dado puede lograrse aplicando el algoritmo de Viterbi.

Se va a dar a continuación otra técnica general para enumerar el coste de los caminos en un diagrama reticular arbitrario de un código convolucional. Aunque no es necesariamente más sencilla que la aplicación directa del algoritmo de Viterbi, tiene la ventaja añadida de que se puede extender

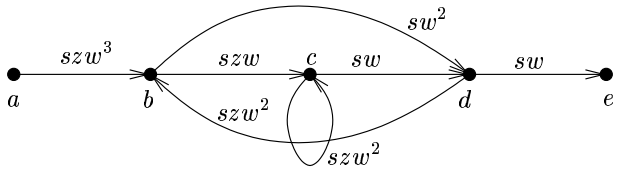


Figura 12.18.

de inmediato para enumerar también la longitud de las trayectorias y para contar el número de bits de información en cada una.

El fundamento de esta técnica es el hecho de que la codificación convolucional es lineal. Así, en cualquier retícula existe siempre una trayectoria fija bien definida: el camino generado por una secuencia de ceros a la entrada y que, en respuesta, produce otra secuencia ilimitada de ceros. Ahora bien, por causa de la linealidad, sucede que el coste entre dos caminos $\mathbf{r}_1$ y $\mathbf{r}_2$ (producidos, respectivamente, por las secuencias de entrada $\mathbf{u}_1$ y $\mathbf{u}_2$) es igual al coste del camino $\mathbf{r}_1 - \mathbf{r}_2$ respecto del camino nulo. Pensemos, por tanto, en el estado $(0, \ldots, 0)$ como principio y fin de todos los caminos erróneos, y veamos la manera de medir todas las trayectorias que divergen del estado cero en el instante inicial y regresan a él en algún instante futuro.

Podemos, para ello, transformar el diagrama de estados y transiciones del codificador en otro grafo aumentado de acuerdo con estas reglas:

a) Eliminar del diagrama de estados original el arco del estado cero a sí mismo.

b) Separar el nodo inicial $(0, \ldots, 0)$ en dos nodos, $(0, \ldots, 0)_1$ y $(0, \ldots, 0)_2$.

c) Sustituir todos los arcos con origen en el estado $(0, \ldots, 0)$ por arcos con origen en el estado $(0, \ldots, 0)_1$.

d) Sustituir todos los arcos incidentes en el estado $(0, \ldots, 0)$ por arcos incidentes en el estado $(0, \ldots, 0)_2$.

e) Sustituir las etiquetas de todos los arcos por el polinomio z^i, en donde i es el peso Hamming de la secuencia de símbolos de salida asociada a la transición.

Una vez hecho esto, pasemos a calcular la función de transferencia del nuevo grafo, interpretando los arcos como elementos de proceso de señal y los nodos como sumadores, y suponiendo que la entrada tiene lugar en el nodo $(0, \ldots, 0)_1$ y la salida en el $(0, \ldots, 0)_2$.

Un ejemplo permitirá aclarar mejor todo el proceso.

EJEMPLO 12.10. La figura 12.18 presenta el grafo transformado del diagrama de estados y transiciones del codificador convolucional

$$G(D) = \begin{pmatrix} 1 & 1+D & 1+D+D^2 \end{pmatrix}.$$

El grafo puede ser dibujado directamente a partir de una etapa de la retícula del codificador, con la siguiente correspondencia entre nodos del grafo y estados del codificador

NODO	a	b	c	d	e
ESTADO	00	10	11	01	00

Cada arco se ha etiquetado simbólicamente con tres variables, $sz^x w^y$, cuyo significado pasamos a aclarar. La variable s aparece en todos los arcos y su exponente siempre vale 1, sirviendo como contador del número de transiciones. La variable x, el exponente de la variable z, es el peso Hamming de los símbolos de entrada que producen la transición; en este caso sólo existe un símbolo de entrada, por lo que x valdrá 0 o 1 según corresponda. Por último, y, el exponente de la variable w, es el peso Hamming de los símbolos de salida del codificador en esa transición.

Si el grafo se interpreta como un esquema de procesado de señal en el dominio transformado (s, z, w), y se consideran sus nodos como operaciones de suma y las etiquetas de cada arco como operaciones de multiplicación, entonces la función de transferencia del diagrama se obtiene resolviendo el sistema de ecuaciones:

$$\begin{aligned}
T_b(s,z,w) &= szw^3 T_a(s,z,w) + szw^2 T_d(s,z,w) \\
T_c(s,z,w) &= szw T_b(s,z,w) + szw^2 T_c(s,z,w) \\
T_d(s,z,w) &= sw^2 T_b(s,z,w) + sw T_c(s,z,w) \\
T_e(s,z,w) &= sw T_d(s,z,w)
\end{aligned}$$

en donde $T_a(\cdot), \ldots, T_e(\cdot)$ son las funciones transformadas en cada nodo del grafo. Tal función de transferencia es

$$H(s,z,w) = \frac{T_e(z,s,w)}{T_a(s,z,w)}.$$

Utilizando las dos primeras ecuaciones del sistema anterior para sustituir en la tercera, se tiene que

$$T_d(s,z,w) = \left(sw^2 + \frac{sw}{1 - szw^2} \right) \left(szw^3 T_a(s,z,w) + szw^2 T_d(s,z,w) \right)$$

de donde, despejando $T_d(s,z,w)$ e insertando su expresión en la cuarta ecuación, se llega a

$$\begin{aligned} H(s,z,w) &= sw\frac{2s^2zw^5 - s^3z^2w^7}{1 - szw^2 - 2s^2zw^4 + s^3z^2w^6} \\ &= sw\left(2s^2zw^5 - 3s^3z^2w^7z^2 + \cdots\right) \\ &= 2s^3zw^6 - 3s^4z^2w^8 + \cdots \end{aligned}$$

La expansión en serie infinita de la función de transferencia $H(s,z,w)$ produce una sucesión de términos cuyos exponentes enumeran, para todas las posibles secuencias, el número de transiciones (el exponente de s), el peso Hamming de la secuencia de salida en esa trayectoria (el exponente de w) y el peso Hamming de la secuencia de entrada (el exponente de z). El coeficiente de cada término señala el número de caminos diferentes con las mismas características. Así por ejemplo, los dos primeros términos de la expansión en serie ilimitada de $H(s,z,w)$ nos indican que, para este código, existen dos caminos diferentes a distancia 6 del camino nulo, compuestos por tres transiciones y generados por dos bits de entrada nulos y otro a uno; y que existen 3 caminos a distancia 8 del camino nulo, con cuatro transiciones y dos bits de entrada no nulos.

Si solamente se quisiera obtener la enumeración de la distancia de los caminos, bastaría fijar $s = z = 1$ en la función de transferencia o, de manera equivalente, etiquetar el grafo transformado sólo con la variable w:

$$T(w) = H(s,z,w)|_{z=s=1} = w\frac{2w^5 - w^7}{1 - w^2 - w + w^6}.$$

Obsérvese que la función general de transferencia $H(s,z,w)$ depende tanto del código como del codificador, mientras que $T(w)$ depende únicamente del código. ■

12.6. Distancia mínima y codificadores catastróficos

El algoritmo de Viterbi que hemos presentado emplea como métrica de comparación la distancia Hamming entre dos posibles secuencias del código. Suponiendo que $\mathbf{x}_1$ y $\mathbf{x}_2$ son dos secuencias del código distintas con distancia Hamming mínima entre ellas

$$d_H(\mathbf{x}_1, \mathbf{x}_2) = \text{mín}\left\{d_H(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C}[n,k]\right\}$$

la secuencia $\mathbf{x}_1 - \mathbf{x}_2$ representa la secuencia de error en el canal con menor número de símbolos no nulos capaz de convertir una secuencia del código en

otra y, con ello, capaz de inducir al decodificador a un error indetectable. Parece lógico pensar que tal distancia mínima va a tener una influencia significativa en la capacidad de detección de errores del código.

DEFINICIÓN 12.8 (DISTANCIA MÍNIMA LIBRE). *La distancia mínima libre de un código convolucional es la mínima distancia Hamming entre dos secuencias del código completas,*

$$d_l = \text{mín}\{d_H(\mathbf{r}, \mathbf{r}') \,|\, \mathbf{r} \neq \mathbf{r}'\} = \text{mín}\{p_H(\mathbf{r}) \,|\, \mathbf{r} \neq \mathbf{0}\}.$$

La definición concuerda con la dada para los códigos de bloques lineales, salvo porque, como las secuencias del código son de longitud infinita, puede ocurrir $p_H(\mathbf{r}) = \infty$. Por citar un caso, la distancia mínima libre del código convolucional utilizado en el ejemplo 12.6 es 3, el peso Hamming de la secuencia del código $(10, 11, 00, 00, \dots)$.

Pero, a diferencia de lo que sucede con los códigos de bloques, la determinación de la distancia mínima libre de un código convolucional dado puede resolverse con facilidad, haciendo uso de la técnica analítica de enumeración de caminos expuesta en el apartado anterior. No obstante, queda por resolver el problema, más amplio, de hallar la máxima distancia mínima libre de un código convolucional $[n, k]$, que es previo a la búsqueda de códigos con tal distancia máxima. Veremos a continuación que del estudio de cierta clase de subcódigos se deducen algunas cotas de distancia útiles que responden parcialmente a esta cuestión. Básicamente, estas cotas extraen las consecuencias de la conexión entre códigos de bloques lineales y códigos convolucionales.

Establezcamos antes el contexto y la notación precisos. Sea $\mathcal{C}$ un código convolucional $[n, k]$, y consideremos el conjunto $\mathcal{C}_L$ formado por todas las secuencias de $\mathcal{C}$ con grado menor o igual que L. Claramente, $\mathcal{C}_L$ es un subespacio vectorial cuya dimensión se representará por δ_L. Por otro lado, sea $G(D)$ una matriz generadora canónica de $\mathcal{C}$, y sean $e_1, e_2, \dots, e_k$ los grados de sus filas (recuérdese que el grado de un vector de polinomios es el mayor grado de cualquiera de sus componentes). El siguiente teorema muestra que la dimensión δ_L se puede calcular a partir de $e_1, \dots, e_k$.

TEOREMA 12.2.

a) $\displaystyle\sum_{L \geq 0} \delta_L t^L = \frac{t^{e_1} + \cdots + t^{e_k}}{(1-t)^2}.$

b) $\displaystyle\delta_L = \sum_{i=1}^{k} \text{máx}\,(L + 1 - e_i, 0).$

Demostración. Para la primera parte, tomemos una matriz generadora canónica $G(D)$ de $\mathcal{C}$ y, sin pérdida de generalidad, supongamos sus filas ordenadas por grados $e_1 \leq \cdots \leq e_k$. Sean precisamente $g_1(D), \ldots, g_k(D)$ las filas, y sea $\mathbf{x}(D)$ una palabra del código de grado menor o igual que L. La secuencia $\mathbf{x}(D)$ será la respuesta del codificador a una entrada $\mathbf{u}(D) = (u_1(D), \ldots, u_k(D))$ formada por polinomios tales que grado$(u_i(D))+e_i \leq L$, para $i = 1, \ldots, k$. Así pues, el conjunto $\{D^j g_i(D) : j + e_i \leq L\}$ constituye una base del subespacio $\mathcal{C}_L$. De esta manera, se puede escribir que

$$\begin{aligned}\sum_{L\geq 0} \delta_L t^L &= \sum_{i=1}^{k} \sum_{j\geq 0} \left(t^{e_i+j} + t^{e_i+j+1} + \cdots\right) \\ &= \sum_{i=1}^{k} \sum_{j\geq 0} \frac{t^{e_i+j}}{1-t} \\ &= \sum_{i=1}^{k} \frac{t^{e_i}}{(1-t)^2}.\end{aligned}$$

Para la segunda parte, introduciendo el desarrollo en serie $(1-t)^{-2} = \sum_{i\geq 0}(i+1)t^i$ en la fórmula anterior, se tiene que

$$\begin{aligned}\sum_{L\geq 0} \delta_L t^L &= \sum_{i=1}^{k} \frac{t^{e_i}}{(1-t)^2} \\ &= \sum_{i=1}^{k} \sum_{j\geq 0} (j+1) t^{e_i+j} \\ &= \sum_{i=1}^{k} \sum_{j\geq e_i} (j+1-e_i) t^j.\end{aligned}$$

Igualando coeficientes en ambos miembros, queda

$$\delta_L = \sum_{i=1}^{k} \text{máx}(L+1-e_i, 0). \qquad \blacktriangleright$$

En resumen, la primera parte del teorema da la expresión analítica de una función generadora de la sucesión de dimensiones $\delta_1, \delta_2, \ldots$, mientras que la segunda proporciona una fórmula explícita para cada término de la sucesión.

Puesto que máx$(x, 0) \geq x$, el siguiente enunciado es un corolario inmediato que acota inferiormente δ_L.

TEOREMA 12.3. *Si $\mathcal{C}$ es un código $[n,k,m]$, para cualquier $L \geq 0$,*

$$\delta_L \geq \text{máx}\big(k(L+1)-m, 0\big).$$

DEMOSTRACIÓN.

$$\delta_L \geq \text{máx}\left(\sum_{i=1}^{k} L+1-e_i, 0\right) = \text{máx}\big(k(L+1)-m, 0\big)$$

ya que $\sum_{i=1}^{k} e_i = m$. ▶

No presenta excesiva dificultad mostrar que la igualdad estricta se logra si y solamente si los grados $e_1, \ldots, e_k$ toman únicamente los valores $\lfloor m/k \rfloor$ y $\lceil m/k \rceil$, habiendo exactamente $m \mod k$ de ellos que valen $\lceil m/k \rceil$, y $k-(m \mod k)$ que valen $\lfloor m/k \rfloor$.

Pues bien, dado que la distancia mínima libre de $\mathcal{C}$ no puede ser mayor que la de ninguno de sus subcódigos $\mathcal{C}_L$, se llega al teorema 12.4.

TEOREMA 12.4. *Sea $\mathcal{C}$ un código convolucional $[n,k,m]$ sobre $\mathbb{F}_q$.*

a) $d_l \leq \min\limits_{L \geq 0} \Delta_q(n(L+1), \delta_L)$, *en donde $\Delta_q(a,b)$ representa la máxima distancia de un código lineal sobre $\mathbb{F}_q$ con longitud a y dimensión b.*

b) $d_l \leq \min\limits_{L \geq 0} \Delta_q(n(L+1), k(L+1)-m)$.

DEMOSTRACIÓN. La primera parte es obvia. La segunda se sigue de combinar la primera con la cota inferior para δ_L. ▶

Una distancia mínima libre elevada reduce la probabilidad de error, cuestión ésta que será analizada en el apartado 12.7. Pero un valor dado de distancia no es suficiente para garantizar que el número de errores de decodificación que se cometen sea finito. En algunos codificadores convolucionales, una pequeña cantidad de errores en la secuencia recibida puede llevar al decodificador a emitir un número ilimitado de símbolos de información erróneos. Intuitivamente, se entiende que esto ocurre cuando un error de decisión en el receptor se propaga y afecta a las decisiones acerca de los símbolos siguientes. En el diagrama reticular, esta situación equivale a la existencia de un camino debido a entradas no nulas que genera ceros a la salida del codificador.

DEFINICIÓN 12.9 (CODIFICADOR CATASTRÓFICO). *Un codificador convolucional es catastrófico cuando alguna secuencia de entrada con un número infinito de símbolos no nulos produce una secuencia del código con un número finito de símbolos no nulos.*

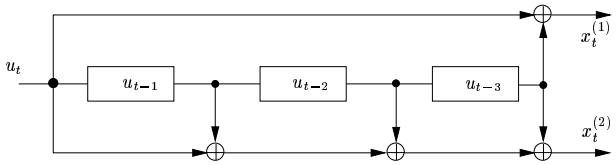


Figura 12.19. Un codificador convolucional catastrófico.

Ejemplo 12.11. El codificador de la figura 12.19 es catastrófico. La secuencia del código que corresponde al mensaje $(1+D)^{-1} = (1,1,1,\dots)$ es $(11,10,11,00,00,\dots)$. Si el ruido en el canal provocase la detección en el receptor de la secuencia $(10,00,00,\dots)$ (con cuatro errores), entonces el algoritmo de decodificación ML estimaría como secuencia del código transmitida la secuencia cero, lo que resultaría en un número ilimitado de errores de decodificación en el mensaje. ■

Es posible demostrar que un codificador convolucional es catastrófico si, y solamente si, no existe un circuito secuencial sin realimentación (es decir, un filtro lineal no recurrente) que invierta las operaciones del codificador. El siguiente teorema establece, en lenguaje algebraico, este hecho, además de varias condiciones equivalentes para discriminar cuándo una matriz generadora polinómica es catastrófica.

Teorema 12.5 (Massey–Sain). *Sea $G(D)$ una matriz generadora polinómica de un código convolucional $\mathcal{C}[n,k]$. $G(D)$ no es un codificador catastrófico si se cumple alguna de las siguientes condiciones:*

a) No existe ninguna entrada de peso infinito $\mathbf{u}(D)$ que produzca una salida $\mathbf{x}(D) = \mathbf{u}(D)G(D)$ de peso finito.

b) El máximo común divisor de los menores de orden k de $G(D)$ es una potencia de D.

c) $G(D)$ tiene una matriz pseudo–inversa de peso finito (con todos los elementos de peso finito). Esto es, existe una matriz polinómica $n \times k$, $H(D)$, tal que $G(D)H(D) = I_k$.

Demostración. Probaremos en primer lugar que $b) \Rightarrow c)$. Sean $G_i(D)$, $i = 1,\dots,\binom{n}{k}$, las submatrices $k \times k$ de $G(D)$ y $\Delta_i(D)$ los menores correspondientes (los determinantes de $G_i(D)$). Sea, además, $C_i(D)$ la traspuesta de la matriz de cofactores de $G_i(D)$. Es un resultado bien conocido del álgebra lineal que $G_i(D)C_i(D) = \Delta_i(D)I_k$. Por lo tanto, situando adecuadamente $n-k$ columnas de ceros en la matriz de cofactores, siempre se puede conseguir una matriz polinómica $n \times k$, $C_i'(D)$, tal que $G(D)C_i'(D) = \Delta_i(D)I_k$.

Ahora bien, puesto que el máximo común divisor de los $\Delta_i(D)$ es D^L para algún $L \geq 0$, por el algoritmo de Euclides se tiene que existen polinomios $g_i(D)$ tales que

$$\sum_i g_i(D)\Delta_i(D) = D^L.$$

Luego, definiendo $H(D) = \sum_i g_i(D)C'_i(D)$, se tiene que

$$\begin{aligned} G(D)H(D) &= \sum_i g_i(D)G(D)C'_i(D) \\ &= \sum_i g_i(D)\Delta_i(D)I_k \\ &= \left(\sum_i g_i(D)\Delta_i(D)\right)I_k = D^L I_k. \end{aligned}$$

En consecuencia, la matriz $D^{-L}H(D)$ es una pseudo–inversa polinómica por la derecha de $G(D)$.

Para probar que $c) \Rightarrow a)$, supóngase que $\mathbf{x}(D) = \mathbf{u}(D)G(D)$ y que $H(D)$ es una matriz pseudo–inversa polinómica de $G(D)$. En tal caso se tiene que $\mathbf{x}(D)H(D) = \mathbf{u}(D)G(D)H(D) = \mathbf{u}(D)$, y si $\mathbf{x}(D)$ es de peso finito entonces $\mathbf{u}(D)$ también lo es, porque $H(D)$ es de peso finito.

Por último, supongamos que $p(D)$ es un polinomio irreducible (que además no es una potencia de D) que divide a todos los menores de orden k de $G(D)$. Sea α una raíz de $p(D)$ en algún cuerpo y $G(\alpha)$ la matriz que resulta de evaluar cada uno de los elementos de $G(D)$ en α. En estas condiciones, cualquier menor de orden k de $G(\alpha)$ es nulo, por lo que el rango de $G(\alpha)$ es menor que k. Por lo tanto, aplicando una secuencia de operaciones elementales a $G(\alpha)$ se puede obtener una matriz $G'(\alpha)$ con una última fila de ceros. Esta secuencia de operaciones elementales significa que existe una matriz elemental $k \times k$, $U(\alpha)$, tal que $G'(\alpha) = U(\alpha)G(\alpha)$. Si el elemento α se sustituye por la indeterminada D, entonces $G'(D) = U(D)G(D)$ con $U(D)$ una matriz unimodular (polinómica, con determinante 1) y con todos los elementos de la última fila de $G'(D)$ divisibles por $p(D)$. Ahora, si se toma el vector

$$\mathbf{u}'(D) = \begin{pmatrix} 0 & 0 & \dots & \dfrac{1}{p(D)} \end{pmatrix}$$

se tiene que $\mathbf{u}'(D)G'(D)$ es un vector de polinomios. Y si se define $\mathbf{w}(D) = \mathbf{u}'(D)U(D)$, entonces $\mathbf{w}(D)G(D) = \mathbf{u}'(D)G'(D)$. En consecuencia, $\mathbf{w}(D)$ es una secuencia de entrada de peso infinito que produce una salida finita y el codificador es catastrófico, lo que prueba que $a) \Rightarrow b)$. ▶

Ejemplo 12.12. Consideremos el codificador convolucional $[4, 2]$ generado

por

$$G(D) = \begin{pmatrix} 1+D & 0 & 1 & D \\ D & 1+D+D^2 & D^2 & 1 \end{pmatrix}.$$

Los menores 2×2 de esta matriz son

$$\begin{aligned} \Delta_{12} &= (1+D)(1+D+D^2) & \Delta_{23} &= 1+D+D^2 \\ \Delta_{13} &= D(1+D+D^2) & \Delta_{24} &= D(1+D+D^2) \\ \Delta_{14} &= 1+D+D^2 & \Delta_{34} &= 1+D+D^2 \end{aligned}$$

en donde los subíndices se refieren a las columnas de $G(D)$ que definen el menor. Como es obvio, el máximo común divisor de los menores es $1+D+D^2$, de modo que la matriz es catastrófica. Sea α una raíz del polinomio $1+D+D^2$ en $\mathbb{F}_4$. Sustituyendo α por D en $G(D)$ tenemos la matriz de rango 1

$$G(\alpha) = \begin{pmatrix} \alpha^2 & 0 & 1 & \alpha \\ \alpha & 0 & \alpha^2 & 1 \end{pmatrix}.$$

Multiplicando la primera fila por α^2 y sumándola a la segunda fila

$$G'(\alpha) = \begin{pmatrix} 1 & 0 \\ \alpha^2 & 1 \end{pmatrix} G(\alpha) = U(\alpha)G(\alpha) = \begin{pmatrix} \alpha^2 & 0 & 1 & \alpha \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Deshaciendo el cambio entre α y D en la matriz $U(\alpha)$

$$\begin{aligned} G'(D) &= \begin{pmatrix} 1 & 0 \\ D^2 & 1 \end{pmatrix} \begin{pmatrix} 1+D & 0 & 1 & D \\ D & 1+D+D^2 & D^2 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 1+D & 0 & 1 & D \\ D(1+D+D^2) & 1+D+D^2 & 0 & 0 \end{pmatrix}. \end{aligned}$$

La segunda fila de $G'(D)$ es divisible por $1+D+D^2$, de modo que

$$\begin{pmatrix} 0 & \dfrac{1}{1+D+D^2} \end{pmatrix} G'(D) = \begin{pmatrix} D & 1 & 0 & 0 \end{pmatrix}.$$

Así pues, la entrada

$$\mathbf{u}(D) = \begin{pmatrix} 0 & \dfrac{1}{1+D+D^2} \end{pmatrix} U(D) = \begin{pmatrix} \dfrac{D^2}{1+D+D^2} & \dfrac{1}{1+D+D^2} \end{pmatrix}$$

de peso infinito produce una salida de peso finito. Ídem para la entrada

$$\mathbf{u}'(D) = \begin{pmatrix} 0 & \dfrac{1}{1+D+D^2} \end{pmatrix} U'(D) = \begin{pmatrix} \dfrac{1+D}{1+D+D^2} & \dfrac{1}{1+D+D^2} \end{pmatrix}$$

si se toma como matriz unimodular a

$$U'(D) = \begin{pmatrix} 1 & 0 \\ 1+D & 1 \end{pmatrix}.$$

■

Este teorema de Massey–Sain no se puede aplicar directamente para matrices generadoras no polinómicas, aunque admite la siguiente generalización, cuya demostración, disponible en [53], se omite.

TEOREMA 12.6. *Sea $G(D)$ una matriz generadora racional de un código convolucional $\mathcal{C}[n,k]$. $G(D)$ es un codificador no catastrófico si se cumple alguna de las siguientes condiciones:*

a) No existe ninguna entrada de peso infinito $\mathbf{u}(D)$ que produzca una salida $\mathbf{x}(D) = \mathbf{u}(D)G(D)$ de peso finito.

b) Sea $\beta(D)$ el mínimo común múltiplo de los denominadores de los elementos de $G(D)$, y $G'(D)$ la matriz obtenida al multiplicar cada elemento de $G(D)$ por $\beta(D)$. Sea $\alpha(D)$ el máximo común divisor de los menores de orden k de $G'(D)$. En el cociente $\alpha(D)/\beta(D)$, expresado como una fracción irreducible, $\alpha(D)$ es una potencia de D.

c) $G(D)$ tiene una matriz pseudo–inversa de peso finito.

Un corolario de este resultado es que ninguna matriz generadora sistemática es catastrófica, ya que para ellas $\alpha(D) = 1 = D^0$.

12.7. Probabilidad de error

Al contrario de lo que sucede con los códigos de bloques, la probabilidad de recuperar sin error un mensaje codificado no constituye una medida útil de las características de control de errores con un código convolucional. El intercambio de información entre emisor y receptor consiste ahora en la transmisión de una secuencia del código de longitud ilimitada. La probabilidad de error al decodificar una secuencia equivale, en este caso, a la probabilidad de que el camino seleccionado en el diagrama reticular por el receptor difiera en uno o más estados de la secuencia transmitida. Pero como la secuencia del código es infinitamente larga, aunque la tasa de error por cada símbolo del canal esté acotada, se comprende que la probabilidad de error al estimar la secuencia transmitida tiende a uno. Se explica así por qué, en el análisis de la probabilidad de error de los códigos convolucionales, el interés se centra en calcular la tasa de error *por bit* a la salida del decodificador; esto es, en calcular el promedio entre el número de bits de información erróneos que produce el decodificador y el número de bits de información que genera.

Supongamos que el decodificador realiza el algoritmo de Viterbi[8] y, en

[8] La decodificación de máxima verosimilitud (ML) de secuencias del código no es necesariamente la óptima para minimizar la probabilidad de error *por bit de información*. Pero su implementación es computacionalmente mucho más sencilla que la de un al-

virtud del principio de linealidad, que se transmite el mensaje (la secuencia del código) $(0,0,0,\dots)$. Si el camino $\hat{\mathbf{x}}$ que estima el decodificador al observar la secuencia recibida es distinto del camino nulo, entonces $\hat{\mathbf{x}}$ necesariamente debe diverger en algún momento del estado $(0,0,\dots,0)$, para volver a converger a él más adelante (si no volviese a converger y el código no fuese catastrófico, la medida de $\hat{\mathbf{x}}$ sería infinita, lo que es imposible si el número de errores en el canal es finito). Cada una de las formas en que sucede esto (cada uno de los distintos caminos que, partiendo del estado cero, retornan luego a él tras pasar por una sucesión de estados intermedios distintos del estado cero) representa un suceso de error, un subintervalo de tiempo en el que el decodificador comete posiblemente errores al estimar la secuencia de símbolos de información.

Definición 12.10 (Secuencia de error). *Una secuencia de error de longitud N en un diagrama reticular de un código convolucional es cualquier camino $\mathbf{s}_0 \to \mathbf{s}_1 \to \dots \mathbf{s}_{N-1} \to \mathbf{s}_0$ que parte del estado nulo y regresa por primera vez a él tras N transiciones: $\mathbf{s}_1 \neq \mathbf{s}_0$, $\mathbf{s}_2 \neq \mathbf{s}_0, \dots, \mathbf{s}_{N-1} \neq \mathbf{s}_0$.*

Una secuencia de error concreta, $\hat{\mathbf{e}}$, da lugar a uno o varios errores de decodificación, tantos como el peso Hamming de la secuencia de símbolos de información asociada a la trayectoria de estados $\hat{\mathbf{e}}$. Definamos E como el conjunto de secuencias de error que comienzan en el instante t del diagrama reticular. Una manera directa de caracterizar los errores de decodificación que origina un miembro cualquiera de E es por medio de una función indicadora

$$I_j(\hat{\mathbf{e}}) = \begin{cases} r & \text{si } \hat{\mathbf{e}} \text{ tiene } r \text{ símbolos de información erróneos} \\ & \text{en el instante } t+j; \\ 0 & \text{si } \hat{\mathbf{e}} \text{ no contiene símbolos de información erróneos} \\ & \text{en el instante } t+j. \end{cases}$$

Así, la probabilidad de que el decodificador elija una determinada secuencia de error con comienzo en t y un error en el instante $t+j$ es sin más

$$1\left(I_j(\hat{\mathbf{e}}) > 0\right) \mathrm{P}(\hat{\mathbf{e}})$$

donde $\mathrm{P}(\hat{\mathbf{e}})$ representa la probabilidad de decidir en favor de la secuencia de error $\hat{\mathbf{e}}$ en lugar de optar por la secuencia $\mathbf{0}$.

La probabilidad de que en el instante t ocurra algún error de decodificación será

$$\mathrm{P}_t = \frac{1}{k} \sum_{j=0}^{t} \sum_{\hat{\mathbf{e}} \in E} I_{t-j}(\hat{\mathbf{e}}) \, \mathrm{P}(\hat{\mathbf{e}}) \tag{12.13}$$

goritmo óptimo y, además, si las probabilidades de error en el canal son pequeñas, la discrepancia en el resultado final es despreciable.

la suma de la contribución de todas las secuencias de error que comienzan en el instante $j = 0, \ldots, t$, puesto que las distintas secuencias de error son eventos mutuamente excluyentes. Vea que en (12.13) se supone implícitamente que $\mathrm{P}(\hat{\mathbf{e}})$ es independiente del tiempo, es decir, se introduce un pequeño error al despreciar la parte inicial transitoria del diagrama reticular.

Intercambiando en (12.13) el orden de las sumas se tiene que

$$\mathrm{P}_t = \frac{1}{k} \sum_{\hat{\mathbf{e}} \in E} \mathrm{P}(\hat{\mathbf{e}}) \sum_{j=0}^{t} I_{t-j}(\hat{\mathbf{e}}) = \frac{1}{k} \sum_{\hat{\mathbf{e}} \in E} f(\hat{\mathbf{e}})\, \mathrm{P}(\hat{\mathbf{e}})$$

con

$$f(\hat{\mathbf{e}}) = \sum_{j=0}^{t} I_{t-j}(\hat{\mathbf{e}})$$

el número de etapas con errores o fallos de decodificación en los símbolos de información de $\hat{\mathbf{e}}$. Si hacemos en estas expresiones $t \to \infty$, considerando que el sistema lleva largo tiempo en operación y que la secuencia del código es de longitud ilimitada, entonces

$$f(\hat{\mathbf{e}}) = \sum_{j=0}^{\infty} I_{t-j}(\hat{\mathbf{e}}) = \sum_{j=0}^{\infty} I_j(\hat{\mathbf{e}})$$

es el número total de errores en los símbolos de información correspondiente a la secuencia de error $\hat{\mathbf{e}}$, y

$$\mathrm{P}_\infty = \frac{1}{k} \sum_{\hat{\mathbf{e}} \in E} f(\hat{\mathbf{e}})\, \mathrm{P}(\hat{\mathbf{e}}) \tag{12.14}$$

es independiente de t. La ecuación (12.14) significa que la probabilidad de cometer algún error de decodificación en régimen permanente es igual al número medio de etapas con errores de decodificación, promediando sobre todas las secuencias de error.

En general, es extremadamente difícil dar una expresión de la probabilidad $\mathrm{P}(\hat{\mathbf{e}})$ de que el decodificador seleccione la secuencia $\hat{\mathbf{e}}$ en lugar de la secuencia nula emitida. Es posible, en cambio, hallar una cota superior de $\mathrm{P}(\hat{\mathbf{e}})$ razonando como sigue. Cuando $p_H(\hat{\mathbf{e}}) = d$, es claro que el decodificador decide en favor de alguna secuencia de error de peso d o menor si la secuencia de símbolos que detecta a la salida del canal tiene peso Hamming mayor o igual que

$$d^* = \left\lfloor \frac{d}{2} \right\rfloor .$$

Por consiguiente, $\mathrm{P}(\hat{\mathbf{e}}) \leq \mathrm{P}_d$ cuando $p_H(\hat{\mathbf{e}}) = d$, con P_d designando la probabilidad de que el ruido en el canal haya transformado la secuencia

nula en una secuencia con d^* o más símbolos no nulos. Supongamos, para simplificar la exposición, que el canal es un canal binario simétrico con probabilidad de equivocación p. En estas condiciones,

$$\mathrm{P}_d = \begin{cases} \displaystyle\sum_{k=(d+1)/2}^{d} \binom{d}{k} p^k (1-p)^{d-k}, & d \text{ impar} \\ \displaystyle\frac{1}{2}\binom{d}{d/2} p^{d/2}(1-p)^{d-d/2} + \sum_{k=d/2+1}^{d} \binom{d}{k} p^k (1-p)^{d-k}, & d \text{ par.} \end{cases}$$

La expresión combinatoria P_d para el caso en que d es par supone que la mitad de las veces que se observa una secuencia de peso $d/2$ se toma la decisión correcta.

Cuando d es impar (si fuese par se razonaría de manera análoga), P_d se puede acotar aplicando el teorema de Chernoff

$$\mathrm{P}_d = \sum_{k=(d+1)/2}^{d} \binom{d}{k} p^k (1-p)^{d-k} \leq \left(2\sqrt{p(1-p)}\right)^d.$$

Si se introduce esta fórmula en (12.14), resulta

$$\mathrm{P}_\infty = \sum_{\hat{\mathbf{e}} \in E} \frac{f(\hat{\mathbf{e}})}{k} \mathrm{P}(\hat{\mathbf{e}}) \leq \frac{1}{k} \sum_{d=1}^{\infty} \sum_{i=1}^{\infty} i \cdot a_{id} w^d \Big|_{w=2(p(1-p))^{1/2}}$$

en donde a_{id} es el número de secuencias de error a distancia d que divergen del estado cero y poseen i símbolos de información erróneos. Para ver de dónde procede esta expresión, note que

$$\frac{1}{k} \sum_{\hat{\mathbf{e}} \in E} f(\hat{\mathbf{e}}) \mathrm{P}(\hat{\mathbf{e}}) \leq \frac{1}{k} \sum_{d=1}^{\infty} w^d \sum_{\hat{\mathbf{e}} \in E : p_H(\hat{\mathbf{e}}) = d} f(\hat{\mathbf{e}})$$

y que, por definición,

$$\sum_{\hat{\mathbf{e}} \in E : p_H(\hat{\mathbf{e}}) = d} f(\hat{\mathbf{e}}) = \sum_{i=1}^{\infty} i \cdot a_{id}.$$

Es un sencilla transformación algebraica verificar que

$$\sum_{d=1}^{\infty} \sum_{i=1}^{\infty} i \cdot a_{id} w^d = \frac{\partial}{\partial z} \left(\sum_{d=1}^{\infty} \sum_{i=1}^{\infty} a_{id} w^d z^i \right) \Big|_{z=1}.$$

Pero

$$T(z, w) = \sum_{d=1}^{\infty} \sum_{i=1}^{\infty} a_{id} w^d z^i$$

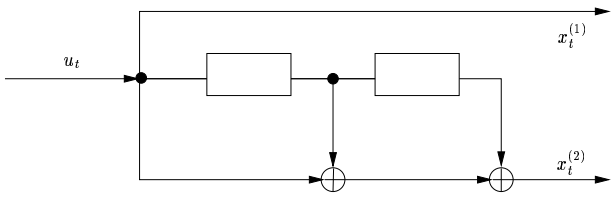


FIGURA 12.20. Codificador del ejemplo 12.13.

es la función enumeradora de distancias (en w) y errores en los bits de información (en z) del código convolucional, y por lo tanto se puede escribir finalmente para la probabilidad de error por bit de información

$$\mathrm{P}_\infty < \frac{1}{k} \cdot \left. \frac{\partial T(z,w)}{\partial z} \right|_{w=2\sqrt{p(1-p)},z=1} . \tag{12.15}$$

Cuando p es pequeña, la cota (12.15) está dominada por la magnitud del primer término, y entonces

$$\mathrm{P}_\infty \approx \frac{1}{k} A_{d_l} \left(2\sqrt{p(1-p)}\right)^{d_l} \approx \frac{1}{k} A_{d_l} 2^{d_l} p^{d_l/2} \tag{12.16}$$

siendo A_{d_l} la suma del número de símbolos de información erróneos en todas las secuencias de error a la distancia mínima libre d_l.

EJEMPLO 12.13. La función de transferencia del codificador convolucional $[2,1]$ de la figura 12.20 es

$$T(z,w) = \frac{w^4 z^2 + w^4 z - w^6 z^2}{1 - w^2 z - w^2 z^2 - w^2 z + w^4 z^2}.$$

Tras realizar algunos cálculos se obtiene

$$\left. \frac{\partial T(z,w)}{\partial z} \right|_{z=1} = \frac{3w^4 - 3w^6 + w^8}{(1 - 3w^2 + w^4)^2}$$

La gráfica logarítmica de esta función se ha representado en la figura 12.21. En el eje de abscisas se indica el valor de $\log(p)$ y en el de ordenadas el de

$$\log \left. \frac{3w^4 - 3w^6 + 4w^8}{(1 - 3w^2 + w^4)^2} \right|_{w=2\sqrt{p(1-p)}}$$

que, a tenor de (12.15), es una cota superior del logaritmo de la probabilidad de error por bit. ■

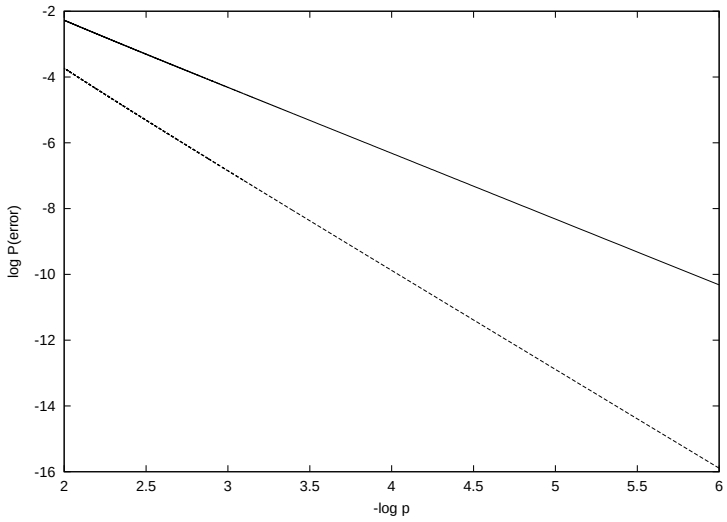


Figura 12.21. Gráfica de la cota de la probabilidad de error de bit (ejemplo 12.13). Por comparación, se muestra con trazo discontinuo la cota de la probabilidad de error de un código convolucional óptimo $[2, 1, 2]$.

12.8. Punción de códigos convolucionales

En el diagrama reticular de un código convolucional binario $\mathcal{C}[n, k, m]$ aparecen 2^m estados y 2^k transiciones de salida (*idem* de entrada) en cada estado. Cuando k o m o ambos son elevados, se dificulta la realización del algoritmo de Viterbi, que en cada etapa debe procesar las medidas de 2^{m+k} ramas. Se presenta entonces un compromiso entre la conveniencia de utilizar un codificador con una tasa y un orden de memoria elevados (un número elevado de estados), para incrementar la fiabilidad y la eficiencia, y la inconveniencia de una retícula densamente conectada para el proceso de decodificación.

El procedimiento de punción o perforación de un código convolucional consiste en eliminar periódicamente ciertos símbolos de salida de un codificador $[n, k]$ para obtener un código de tasa mayor, al tiempo que se preserva la estructura de la retícula del primero. Lo habitual en la práctica es que $k = 1$, pero haremos la descripción para un caso general. Consideremos, en primer lugar, el código convolucional que resulta al agrupar las secuencias de símbolos de entrada de M en M; esto es, el código convolucional $\mathcal{C}^{[M]}[nM, kM]$. El código perforado se obtiene eliminando en cada conjunto de nM símbolos de salida de $\mathcal{C}^{[M]}$ un cierto número de símbolos, que se

especifican por medio de una *matriz de perforación* $P = \{p_{ij}\}_{n\times M}$, en la que $p_{ij} = 0$ significa que el i-ésimo símbolo de salida del bloque j del grupo de M bloques se borra, mientras que $p_{ij} = 1$ si se mantiene. Si la matriz de perforación tiene un total de α elementos nulos, el código perforado es de parámetros $[nM - \alpha, kM]$, es decir, de tasa $kM/(nM - \alpha)$.

El diagrama reticular del código perforado se obtiene a partir de la retícula del código $[n, k]$ original simplemente prescindiendo en las etiquetas de salida de cada rama de los símbolos que se han eliminado. El proceso de perforación produce, por tanto, un diagrama reticular con 2^k transiciones de salida (*idem* de entrada) en cada estado, diagrama que no es uniforme en cada etapa,[9] aunque es periódico con periodo M.

El código obtenido por perforación es un código convolucional cuya matriz generadora se puede obtener con el siguiente procedimiento. Sea $G(D)$ una matriz generadora canónica del código de partida $[n, k, m]$, que se podrá escribir como

$$G(D) = G_0 + G_1 D + G_2 D^2 + \cdots + G_m D^m$$

expresión en la cual el conjunto de matrices

$$G = \begin{pmatrix} G_0 & G_1 & \ldots & G_m \end{pmatrix}$$

es el patrón generador del código en el dominio del tiempo. El patrón generador del código convolucional extendido M veces será

$$G^{[M]} = \begin{pmatrix} G_0 & G_1 & \ldots\ldots\ldots & G_m & & \\ & G_0 & G_1 & \ldots\ldots & G_{m-1} & G_m \\ & & \ldots\ldots\ldots\ldots\ldots\ldots\ldots & & & \\ & & & \ldots\ldots\ldots\ldots\ldots\ldots & & \\ & & & G_0 & G_1 & \ldots\ldots\ldots & G_m \end{pmatrix} \begin{matrix} 1 \\ 2 \\ \vdots \\ \\ M \end{matrix}$$

Las dimensiones de $G^{[M]}$ son $kM \times n(M + m)$. Dentro de cada bloque de nM columnas consecutivas $[1, nM]$, $[nM + 1, 2nM]$ ... de esta matriz, eliminemos las columnas $i+(j-1)M$ para los índices i, j tales que $p_{ij} = 0$. En la matriz resultante, $G_p^{[M]}$, agrupemos las columnas en bloques de $nM-\alpha$. El patrón generador en el dominio del tiempo del código perforado es, entonces,

$$\begin{aligned} G_p &= \left(G_p^{[M]}(1, nM - \alpha) \quad G_p^{[M]}(nM - \alpha + 1, 2(nM - \alpha)) \quad \ldots \right) \\ &= \begin{pmatrix} G_{0,p} & G_{1,p} & \ldots & G_{r,p} \end{pmatrix}, \qquad r = \left\lceil \frac{m + M}{nM - \alpha} \right\rceil \end{aligned}$$

[9] No son constantes las secuencias de salida de las transiciones en etapas consecutivas; la estructura de transiciones sí es regular.

en donde $G_p^{[M]}(i,j)$ es la submatriz de $G_p^{[M]}$ compuesta por las filas $i, i+1, \ldots, j$. La matriz generadora, en el dominio transformado D, del código obtenido por perforación se obtiene directamente a partir del generador G_p como

$$G_p(D) = G_{0,p} + G_{1,p}D + \cdots + G_{r,p}D^r.$$

$G_p(D)$ es una matriz canónica si $G(D)$ lo es, y el código perforado es antipodal cuando el de partida es antipodal.

En general, la punción es un método con el que construir códigos convolucionales de tasa $kM/(nM-\alpha)$ a partir de uno de tasa k/n, de tal manera que su diagrama reticular tenga la misma estructura de estados e igual complejidad que la del código matriz y, gracias a ello, la decodificación sea más simple. Pero no se da la circunstancia, en general, de que los códigos perforados sean los óptimos (bajo el criterio de minimización de la probabilidad de error) de entre la clase de códigos $[nM-\alpha, kM]$. Por otro lado, eligiendo adecuadamente la matriz de perforación, es posible cambiar la tasa de un dispositivo de codificación, en función de las propiedades del canal por ejemplo, sin variar en nada su estructura física.

Ejemplo 12.14. Para el código $\mathcal{C}[3,1,3]$ del ejemplo 12.1, con matriz generadora

$$\begin{aligned} G(D) &= \begin{pmatrix} 1+D & 1+D^2 & 1+D^3 \end{pmatrix} \\ &= (1,1,1) + (1,0,0)D + (0,1,0)D^2 + (0,0,1)D^3 \end{aligned}$$

elijamos la matriz de perforación

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

P da lugar a un código $[4,3]$ que se obtiene al borrar de $\mathcal{C}$ el primer bit del primer grupo de tres bits a la salida, los bits primero y segundo de la segunda terna de bits a la salida y los bits segundo y tercero de la siguiente terna de salida.

El patrón generador de $\mathcal{C}$ extendido 3 intervalos de tiempo es

$$G^{[3]} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & & & & & \\ & & & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & & \\ & & & & & & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

El patrón generador del código perforado se obtiene tras borrar las columnas 1, 4, 5, 8 y 9 de cada grupo de 9 columnas de $G^{[3]}$

$$G_p^{[3]} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

que, una vez agrupadas sus columnas de 4 en 4 y convertida al dominio D, da como matriz generadora del código perforado a

$$G_p(D) = \begin{pmatrix} 1 & 1+D & 0 & 0 \\ D & 0 & 1+D & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Obsérvese que la perforación no preserva el grado: $G_p(D)$ es un código convolucional $[4, 3, 1]$. ■

Notas bibliográficas

Los códigos convolucionales fueron propuestos originalmente en 1955 [16]. Pronto se desarrollaron los primeros métodos de decodificación secuencial ([78] y [42] constituyen dos buenas referencias tempranas), pero la falta de un algoritmo eficiente ML de decodificación y la inexistencia de una teoría formal apropiada hicieron decaer el interés inicial. La solución por parte de Viterbi [71] al problema de la decodificación eficiente reactivó este campo de estudio. El principal impulso por dotar a los códigos convolucionales de una fundamentación formal sólida es de Forney [20, 21]. La exposición más completa de la teoría algebraica de los códigos convolucionales es de Piret [51], aunque McEliece [53, cap. 12] también ofrece una cobertura extensa.

El empleo de códigos convolucionales como códigos interiores en sistemas de codificación concatenada o multinivel es una técnica bien conocida, y se aplica en numerosos sistemas reales. La propuesta original de codificación concatenada con decodificación *soft* es de G. D. Forney [19], pero se puede hallar un tratamiento más uniforme en [53].

El empleo integrado de códigos convolucionales y modulaciones digitales redundantes (llamado modulación codificada [67]) constituye el avance más importante de las técnicas de codificación durante la década de los ochenta. Permitió mejorar notablemente la fiabilidad de los canales limitados en banda manteniendo una eficiencia espectral elevada. Una de las ideas centrales de la modulación codificada es la observación por parte de G. Ungerboeck de que la distancia euclídea entre las señales, y no la distancia Hamming entre las palabras del código, es el parámetro más importante para conseguir una buena protección frente a los errores. Los textos de transmisión digital, como [55], dedican análisis pormenorizados a esta solución.

Las relaciones entre los códigos de bloques, la representación reticular y los códigos convolucionales, con particular énfasis en la decodificación Viterbi de códigos de bloques, se exploran en [40].

En los últimos años, se ha producido un avance práctico muy significativo, al descubrirse que el empleo de codificadores convolucionales recurrentes

en paralelo y de un procedimiento de decodificación *soft* MAP iterativo consiguen resultados prácticamente idénticos al límite de Shannon. El uso de códigos convolucionales resulta esencial en este esquema, llamado decodificación turbo [32, 72]. Muy recientemente se ha dado forma a una teoría que unifica ciertos códigos de bloques y los códigos convolucionales como códigos definidos sobre grafos. Esta descripción revela conexiones sorprendentes entre ambos, y permite explicar en parte el funcionamiento de los algoritmos MAP de decodificación iterativos. Resulta insólito descubrir que una familia de códigos de esta clase ya había sido propuesta por R. G. Gallager en 1962 [24].

Bibliografía

[1] N. Abramson. *Information theory and coding*. McGraw-Hill, 1963. Existe traducción al castellano, *Teoría de la información y la codificación*. Editorial Paraninfo, 1986. 4.7.3

[2] E. F. Asmuss, Jr., H. F. Mattson, Jr. y R. J. Turyn. «Research to develop the algebraic theory of codes». *Informe AFCRL-67-0365*, Air Force Cambridge Res. Labs., 1967. 7.9

[3] E. R. Berlekamp. *Algebraic coding theory*. McGraw-Hill, 1968. También en: Aegean Park Press, Laguna Hills, edición revisada, 1984. 9.6, 10.4.3, 11.7

[4] E. R. Berlekamp. *Key papers in the development of coding theory*. IEEE Press, 1974. 6.10, 10, 28, 52

[5] D. Bertsekas, R. G. Gallager. *Data networks*. Prentice Hall, 2.ª edición, 1992. 8.2.5

[6] U. D. Black. *Data link protocols*. Prentice Hall, 1993. 8.2.5

[7] R. E. Blahut. *Theory and practice of error control codes*. Addison-Wesley, 1983. 7.9

[8] R. E. Blahut. *Algebraic codes for data transmission*. Cambridge Univ. Press, 2002. 7.9

[9] T. C. Bell, J. G. Cleary e I. H. Witten. *Text compression*. Prentice-Hall, 1990. 3.B.2

[10] R. C. Bose, D. K. Ray–Chaudhuri, «On a class of error correcting binary group codes». *Inform. and Control*, vol. 3, págs. 68–79, 1960. Reimpreso en [4], págs. 78–81. 7.9, 10.4.3

[11] A. R. Calderbank. «The art of signaling: fifty years of coding theory». En *Information theory: 50 years of discovery*, S. Verdú y S. McLaughlin (editores), págs. 517–551, IEEE Press, 2000. 6.10

[12] W. Cary Huffman, V. Pless. *Fundamentals of error-correcting Codes*. Cambridge University Press, 2003. 11.5

[13] J. H. Conway, N. J. A. Sloane. *Sphere packings, lattices and groups*. Springer–Verlag, 1988. 5

[14] D. J. Costello, Jr., J. Hagenauer, H. Imai y S. B. Wicker. «Applications of error–control coding». En *Information theory: 50 years of discovery*, S. Verdú y S. McLaughlin (editores), págs. 487–516, IEEE Press, 2000. 6.10

[15] T. M. Cover, J. A. Thomas. *Elements of information theory*. Wiley, 1991. 2.4.1, 3.A, 3.B.2

[16] P. Elias. «Coding for noisy channels». *1955 International Convention Record*, págs. 37–46, 1955. 12.8

[17] R. M. Fano. «Class notes for course 6574: transmission of information». MIT, 1952. 4.7.3

[18] A. Feinstein. «A new basic theorem of information theory». *IRE Trans. Inf. Theory*, vol. 4, págs. 2–22, 1954. 5.3

[19] G. D. Forney, Jr. *Concatenated codes*. MIT Press, 1966. 12.8

[20] G. D. Forney, Jr. «Convolutional codes I: algebraic structure». *IEEE Trans. Inf. Theory*, vol. 16, págs. 720–738, 1970. 12.1, 12.8

[21] G. D. Forney, Jr. «The Viterbi algorithm». *Proc. IEEE*, vol. 61, págs. 268–276, 1973. 12.4.1, 12.8

[22] G. D. Forney, Jr. «Structural analysis of convolutional codes via dual codes». *IEEE Trans. Inf. Theory*, vol. 19, págs. 512–518, 1973. 12.1

[23] G. D. Forney, Jr. «Minimal bases of rational vector spaces, with applications to multivariable linear systems». *SIAM Journal on Control*, vol. 13, n° 3, págs. 493–520, 1975. 12.1, 12.1

[24] R. G. Gallager. *Low density parity-check codes*. MIT Press, 1962. 12.8

[25] R. G. Gallager. «A simple derivation of the coding theorem and some applications». *IEEE Trans. Inf. Theory*, vol. 11, págs. 3–18, 1965. 5.3

[26] R. G. Gallager. *Information theory and reliable communication*. Wiley, 1968. 2.4.1

[27] R. G. Gallager. «Claude E. Shannon: a retrospective of his Life, work and impact». *IEEE Trans. Inf. Theory*, vol. 47, núm. 7, págs. 2681–2695, 2001. 11

[28] D. C. Gorenstein, N. Zierler. «A class of error-correcting codes in p^m symbols». *J. SIAM*, vol. 9, págs. 207–214, 1961. Reimpreso en [4], págs. 87–89. 11.7

[29] M. G. Gouda. *Elements of network protocol design*. Wiley, 1998. 8.2.5

[30] V. Guruswami, M. Sudan. «Improved decoding of Reed-Solomon and algebraic-geometric codes». *IEEE Trans. Inf. Theory*, vol. 45, págs. 1757–1767, 1999. 11.5, 11.7

[31] R. V. L. Hartley. «Transmission of information». *Bell Systems Technical Journal*, vol. 7, págs. 535–563, julio 1928. 2

[32] C. Heegard, S. B. Wicker. *Turbo coding*. Kluwer Academic Publishers, 1998. 12.8

[33] G. J. Holzmann. *Design and validation of computer protocols*. Prentice Hall, 1991. 8.1.1, 8.2.5

[34] D. Huffman. «A method for the construction of minimum redundancy codes». *Proc. IRE*, vol. 40, págs. 1098–1101, 1952. 3.6

[35] J. Karush. «A simple proof of an inequality of McMillan». *IEEE Trans. Inf. Theory*, vol. 7, pág. 118, 1961. 3.6

[36] R. Koetter, A. Vardy. «Algebraic soft-decision decoding of Reed-Solomon codes». *IEEE Trans. Inf. Theory*, vol. 49, núm. 11, págs. 2809–2825, 2003. 11.5

[37] L. Kraft. «A device for quantizing, grouping and coding amplitude modulated pulses». M. S. Thesis, Dept. Elec. Eng., MIT, 1949. 3.6

[38] A. Lempel, J. Ziv. «On the complexity of an individual sequence». *IEEE Trans. Inf. Theory*, vol. 22, págs. 75–81, 1976. 3.B.2

[39] S. Lin, D. J. Costello. *Error control coding: fundamentals and applications*. Prentice Hall, 1983. 7.9, 8.2.5, 12.4

[40] S. Lin, T. Kasami, T. Fujiwara y M. Fossorier. *Trellises and trellis-based decoding algorithms for linear block codes*. Kluwer Academic Publishers, 1998. 12.3, 12.8

[41] F. J. MacWilliams, N. J. A. Sloane. *The theory of error correcting codes*. North Holland, 1977. 6.9, 6.10, 6.A, 6.A, 7.9, 10.1, 10.3, 3, 5

[42] J. L. Massey. *Threshold decoding*. MIT Press, 1963. 12.8

[43] J. L. Massey. «Shift-register synthesis and BCH decoding». *IEEE Trans. Inf. Theory*, vol. 15, págs. 122–127, 1969. 10.4.3

[44] H. F. Mattson, Jr., G. Solomon. «A new treatment of Bose–Chaudhuri codes». *J. SIAM*, vol. 9, págs. 654–669, 1961. 7.9

[45] R. J. McEliece. *The theory of information and coding*. Encyclopedia of Math. and its applications, Vol. 3. Addison Wesley, 1977. También en Cambridge Univ. Press, 2001, 2.ª edición. 2.4.1, 6.A

[46] R. J. McEliece, E. R. Rodemich, H. C. Rumsey y L. R. Welch. «New upper bounds on the rate of a code via the Delsarte-MacWilliams inequalities». *IEEE Trans. Inf. Theory*, vol. 23, págs. 157–166, 1977.

[47] R. J. McEliece. *Finite fields for computer scientists and engineers*. Kluwer Academic Publishers, 1987. 9.6

[48] B. McMillan. «The basic theorems of information theory». *Ann. Math. Statist.*, vol. 24, págs. 196–219, 1953. 3.6

[49] B. McMillan. «Two inequalities implied by unique decipherability». *IRE Trans. Inf. Theory*, vol. 2, págs. 115–116, 1956. 3.6

[50] M. S. Pinsker. *Information and information stability of random variables and processes*. Holden Day, 1964. Publicado originalmente en ruso en 1960. 4.7.3

[51] Ph. Piret. *Convolutional codes, an algebraic approach*. MIT Press, 1988. 12.8

[52] V. S. Pless. «Power moment identities on weight distributions in error correcting codes». *Inform. and Control*, vol. 6, págs. 147–152, 1963. Reimpreso en [4], págs. 266–267. 6.10

[53] V. S. Pless, W. C. Huffman. *Handbook of coding theory. Vols. I y II*. Elsevier, 1998. 7, 6.10, 6.A, 7.9, 10.4.3, 11.7, 12.1, 12.6, 12.8

[54] V. S. Pless. *Introduction to the theory of error-correcting codes*. Wiley, 3.ª edición, 1998. 6.10

[55] J. G. Proakis. *Digital communications*. McGraw-Hill, 4.ª edición, 2001. 12.8

[56] I. S. Reed, G. Solomon. «Polinomial codes over certain finite fields». *J. SIAM*, vol. 8, págs. 300–304, 1960. 11.7

[57] I. S. Reed, X. Cheng. *Error-control coding for data networks*. Kluwer Academic Publishers, 1999. 11.7

[58] D. Salomon. *Data compression*. Springer, 2.ª edición, 2000. 3.B.1

[59] C. E. Shannon. «A mathematical theory of communication». *Bell Systems Technical Journal*, vol. 27, págs. 327–423. Julio, 1948. Reimpreso en: C. E. Shannon y W. Weaver (editores), *A mathematical theory of communication*. Univ. of Illinois Press, 1949 (primera reimpresión), 1963 (segunda reimpresión). 12, 15, 2, 2.4.1, 3.6, 3.A, 4.7.3, 5.3, 6.10

[60] C. E. Shannon. «A mathematical theory of communication». *Bell Systems Technical Journal*, vol. 27, págs. 623–656. Octubre, 1948. Reimpreso en: C. E. Shannon y W. Weaver (editores), *A mathematical theory of communication*. Univ. of Illinois Press, 1949 (primera reimpresión), 1963 (segunda reimpresión). 12, 2.4.1, 4.7.3

[61] C. E. Shannon. «Certain results in coding theory for noisy channels». *Inform. Contr.*, vol. 1, págs. 6–25, 1957. 5.3

[62] D. Slepian. «A class of binary signaling alphabets». *Bell Systems Technical Journal*, vol. 39, págs. 203–234, 1956. 6.10

[63] D. Slepian. «Some further theory of group codes». *Bell Systems Technical Journal*, vol. 39, págs. 1219–1252, 1960. 6.10

[64] D. Slepian. *Key papers in the development of coding theory*. IEEE Press, 1974. 6.10

[65] N. J. A. Sloane, A. D. Wyner. *Claude Elwood Shannon: collected papers*. IEEE Press, 1993. 11

[66] Y. Sugiyama, M. Kasahara, S. Hirasawa y T. Namekawa. «A method for solving key equation for decoding Goppa codes». *Infor. and Control*, vol. 27, págs. 87–99, 1975. 11.4.3, 11.7

[67] G. Ungerboeck. «Channel coding with multilevel/phase signals». *IEEE Trans. Inf. Theory*, vol. 28, págs. 55–67, 1982. 12.8

[68] J. H. van Lint, R. M. Wilson. «On the minimum distance of cyclic codes». *IEEE Trans. Inf. Theory*, vol. 32, págs. 23–40, 1986. 7.9, 10.4.3

[69] J. H. van Lint. *Introduction to coding theory*. Springer, 3.ª edición, 1998. 6.10, 10.1

[70] S. Verdú, S. McLaughlin (editores). *Information theory: 50 years of discovery*. IEEE Press, 2000. 6.10

[71] A. J. Viterbi. «Error bounds for convolutional codes and an asymptotically optimum decoding algorithm». *IEEE Trans. Inf. Theory*, vol. 13, págs. 260–269, abril 1967. 12.8

[72] B. Vucetic, J. Yuan. *Turbo codes. principles and applications*. Kluwer Academic Publishers, 2000. 12.8

[73] T. A. Welch. «A technique for high performance data compression». *Computer*, vol. 17, págs. 8–19, 1984. 3.B.2

[74] S. B. Wicker, B. K. Barghawa. *Reed-Solomon codes and their applications*. IEEE Press, 1994. 11.7

[75] S. B. Wicker. *Error control systems for digital communications and storage*. Prentice Hall, 1995. 8.2.5, 11.7

[76] A. D. Wyner, J. Ziv. «The sliding-window Lempel-Ziv algorithm is asymptotically optimal». *Proc. IEEE*, vol. 82, págs. 872–877, 1994. 3.B.2

[77] J. Wolfowitz. «The coding of messages subject to chance errors». *Illinois Journal of Mathematics*, vol. 1, págs. 591–606, 1957. 5.3

[78] J. M. Wozencraft, B. Reiffen. *Sequential decoding*. MIT Press, 1961. 12.8

[79] J. Ziv, A. Lempel. «A universal algorithm for sequential data compression». *IEEE Trans. Inf. Theory*, vol. 23, págs. 337–343, 1977. 3.B.2

[80] J. Ziv. «Coding theories for individual sequences». *IEEE Trans. Inf. Theory*, vol. 24, págs. 405–412, 1978. 3.B.2

[81] J. Ziv, A. Lempel. «Compression of individual sequences via variable-rate coding». *IEEE Trans. Inf. Theory*, vol. 24, págs. 530–536, 1978.

3.B.2

Índice alfabético

D

Este libro ha sido compuesto utilizando tipos EC a 10pt y el sistema de edición LaTeX 2_ε. Se terminó de imprimir en la imprenta Tórculo Artes Gráficas, Santiago de Compostela, España, el 25 de septiembre de 2003.

Este libro contiene las ideas básicas de la Teoría de la Información y la Teoría de la Codificación, dos campos esenciales para la comprensión cabal de la ingeniería de los modernos sistemas de transmisión digital.

Los autores han querido dar:

- una presentación autocontenida y actualizada, en un solo volumen, de la Teoría de la Información, las técnicas de codificación de fuente, las de codificación de canal y los fundamentos de los protocolos de retransmisión;
- un estudio detallado de los códigos algebraicos de control de errores más utilizados en las aplicaciones: los códigos lineales, los códigos cíclicos, los códigos BCH y los códigos Reed–Solomon, con abundantes ejemplos de los métodos de decodificación más apropiados para cada clase;
- una exposición precisa de los códigos convolucionales en cuanto que técnicas de control de errores; la mayoría de textos los tratan sólo desde la perspectiva de las modulaciones digitales.

Cándido López García es Dr. Ingeniero de Telecomunicación por la Universidad Politécnica de Madrid (1995). Actualmente es Profesor Titular de Universidad del Departamento de Ingeniería Telemática de la Universidad de Vigo e imparte clases de *Fundamentos de Telemática* y *Redes y Servicios Telemáticos* en la E.T.S.I. de Telecomunicación.

Manuel Fernández Veiga es Dr. Ingeniero de Telecomunicación por la Universidad de Vigo (2001). Actualmente es Profesor Titular de Universidad del Departamento de Ingeniería Telemática en la misma Universidad e imparte clases de *Fundamentos de Telemática* y *Redes y Servicios Telemáticos* en la E.T.S.I. de Telecomunicación.