Teoría de la Información y la Codificación

Ejercicios – Tema 3

Curso 2017/2018

Francisco Javier Caracuel Beltrán

caracuel@correo.ugr.es

4º - Grado en Ingeniería Informática – CCIA – ETSIIT

Índice

1. Explica el Teorema Fundamental de Shannon para canales sin ruido y expón sus consecuencias más inmediatas.

Si se tiene una fuente con entropía H (bits por símbolo) y un canal con capacidad C (bits por segundo), es posible codificar la salida de la fuente de tal modo que se puede transmitir a un régimen de C/H – ε símbolos por segundo sobre el canal. No se puede transmitir a un régimen promedio mayor que C/H.

Este teorema prueba la existencia de un límite a la eficiencia de lo que se denomina consecuencias: cuanto menor sea la entropía de la fuente, menos información se debe enviar.

2. Explica qué es un código uniforme. Pon un ejemplo de código uniforme para codificar 5 mensajes {A, B, C, D, E}.

Un código uniforme es aquel cuyos símbolos tienen la misma longitud. Por ejemplo, el código ASCII.

Ejemplo:

- A: 000
- B: 001
- C: 010
- D: 011
- E: 100

3. Explica qué es un código no uniforme. Pon un ejemplo de código no uniforme para codificar 5 mensajes {A, B, C, D, E}.

Un código no uniforme es aquel en el que algunos símbolos tienen una longitud y otros tienen otra. Por ejemplo, el código Morse.

Ejemplo:

- A: 0
- B: 1
- C: 00
- D: 01
- E: 10

4. Explica qué es un código de traducción única. Pon un ejemplo de un código que permita codificar 5 mensajes {A, B, C, D, E} que sea de traducción única. Pon un ejemplo de código para el mismo conjunto de mensajes que no sea de traducción única y explica por qué no lo es. Indica, mediante un ejemplo que codifique la secuencia de mensajes {AACAD}, cuál es la principal desventaja de códigos que no son de traducción única.

Un código de traducción única es aquel en el que para cualquier sucesión de símbolos que se reciba, solo hay una manera de decodificarlo.

Ejemplo:

- A: 000
- B: 001
- C: 010
- D: 011
- E: 100

Ejemplo de código que no es de traducción única:

- A: 0
- B: 1
- C: 00
- D: 01
- E: 10

No es de traducción única porque si se trasmite el mensaje codificado 0010 no se puede decodificar unívocamente. Puede ser "AABA", "CE", etc.

Un ejemplo que codifique la secuencia "AACAD" con el código de traducción única es "000000010000011". Con el código que no es de traducción única es "0000001".

La principal desventaja de los códigos que no son de traducción única es que no se puede garantizar cuál es el mensaje que envió el emisor (sin utilizar elementos de control).

5. Explica qué es un código instantáneo. Indica un procedimiento general para generar códigos instantáneos y utilízalo para generar un código instantáneo que permita codificar 5 mensajes {A, B, C, D, E}. Indica cómo se codificaría y decodificaría la secuencia de mensajes {AACAD}. Pon otro ejemplo de código que no sea instantáneo para el mismo conjunto de mensajes. Indica cómo sería el proceso para decodificar con este nuevo código y exponga un ejemplo de decodificación para la cadena de mensajes {AACAD}.

Un código instantáneo es un tipo de código de traducción única en el que ningún símbolo del código es el comienzo de otro. No hay que señalizar el fin de símbolo.

Se puede utilizar un algoritmo de generación de códigos instantáneos. Sea M el conjunto de mensajes a enviar ({A, B, C, D, E}). Sea A el conjunto de símbolos transmisibles ({0,1}).

1. Se divide M en D subconjuntos. A cada elemento de cada subconjunto se le asigna el símbolo correspondiente de los símbolos transmisibles.
2. Se repite el paso 1 hasta que cada subconjunto solo tenga 1 elemento.

Codificación de 5 mensajes: M = {A, B, C, D, E}.

- Se divide M en dos subconjuntos: M1 = {A, B, C} → 0; M2 = {D, E} → 1
- Se divide M1 en dos subconjuntos: M11 = {A, B} → 00; M12 = **{C} → 01**
- Se divide M11 en dos subconjuntos: M111 = **{A} → 000**; M112 = **{B} → 001**
- Se divide M2 en dos subconjuntos: M21 = **{D} → 10**; M22 = **{E} → 11**

El código instantáneo generado para codificar M es:

- A: 000
- B: 001
- C: 01
- D: 10
- E: 11

La secuencia de mensajes AACAD se codificaría como 0000000100010. Habría que sustituir cada mensaje por su valor codificado.

Para decodificar 0000000100010 habría que ir recorriendo cada bit. Si el bit en cuestión aparece en la tabla de codificación se sustituye por su mensaje y se continúa con el siguiente. Si ese bit no aparece, se continúa por el siguiente manteniendo un subconjunto con ambos bits. Este proceso se repite hasta que el conjunto de bits sí coincide con alguno que se encuentra en la tabla de codificación:

- **0**000000100010 → 0 no existe en la tabla de codificación.
- **00**000000100010 → 00 no existe en la tabla de codificación.
- **000**0000100010 → 000 si existe en la tabla de codificación. Se sustituye por A.
- …

000 000 01 000 10 → A A C A D

Un ejemplo de código no instantáneo es:

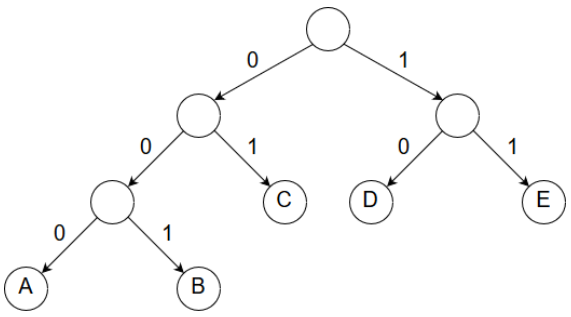
- A: 000
- B: 001
- C: 010
- D: 011
- E: 100

Al no ser un código instantáneo se puede crear un código que codifique los mensajes con el mismo número de bits. El código AACAD codificado es: 000000010000011.

Con este código, se obtiene una codificación que requiere más bits que con el código instantáneo. Para decodificarlo es suficiente con agrupar los bits en subconjuntos de 3 bits y buscar en la tabla el mensaje al que corresponden: 000 000 010 000 011 → A A C A D.

6. Indica qué es un árbol de codificación para un código instantáneo. Pon un ejemplo de árbol de codificación para el código instantáneo desarrollado en el ejercicio anterior. Explica los procedimientos para codificar y para decodificar la cadena de mensajes {AACAD} utilizando el árbol de codificación.

Un árbol de codificación para un código instantáneo es un árbol que representa todas las palabras de un alfabeto. En los nodos hoja se encuentran los símbolos del alfabeto. El código de cada símbolo del alfabeto se representa por la secuencia que se genera al recorrer desde la raíz hasta el nodo hoja correspondiente.



Para codificar la cadena de mensajes AACAD se debe recorrer cada mensaje, guardando el conjunto de bits hasta llegar a la raíz. Cuando se termine, el mensaje codificado se debe invertir y ése será el resultado.

Para decodificar una serie de bits, se recorre uno a uno, avanzando por el árbol hasta llegar a un nodo hoja. Cuando se llega a un nodo hoja, se guarda el valor correspondiente y se continúa con los siguientes bits desde el nodo raíz.

7. Explica la desigualdad de Kraft y qué implicaciones tiene en códigos instantáneos. Pon un ejemplo de código que permita codificar 5 mensajes {A, B, C, D, E} y que cumpla la desigualdad de Kraft y, explica qué propiedades tiene. Pon otro ejemplo de código que permita codificar 5 mensajes {A, B, C, D, E} y que no cumpla la desigualdad de Kraft y, explica que propiedades tiene.

La desigualdad de Kraft expresa las condiciones suficientes para que un código sea instantáneo. Se calcula como la suma del número de símbolos que codifica el traductor elevado a la longitud en negativo de cada palabra:

$$\sum_{i=1}^{N} D^{-ni}$$

Si el resultado es menor o igual a 1, el código existe.

Si es mayor de 1, el código no puede ser de decodificación instantánea.

Ejemplo: código desarrollado en el ejercicio 5:

- A: 000
- B: 001
- C: 01
- D: 10
- E: 11

$$\sum_{i=1}^{N} D^{-ni} = 2^{-3} + 2^{-3} + 2^{-2} + 2^{-2} + 2^{-2} = 1$$

El código anterior existe (ya se ha comprobado anteriormente).

Ejemplo: código que no cumple la desigualdad de Kraft podría ser uno cuya longitud M (mensajes a codificar) fuera 5 (A, B, C, D, E), A (símbolos a utilizar por el codificador fuera {0, 1} y las longitudes de los mensajes fueran: |c(A)|=3, |c(B)|=2, |c(C)|=2, |c(D)|=2, |c(E)|=2:

$$\sum_{i=1}^{N} D^{-ni} = 2^{-3} + 2^{-2} + 2^{-2} + 2^{-2} + 2^{-2} = 1{,}125$$

Como el resultado es mayor de 1, el código no existe.

8. Explica la igualdad de Kraft y qué implicaciones tiene. Pon un ejemplo de código que permita codificar 5 mensajes {A, B, C, D, E} y que cumpla la igualdad de Kraft y, explica qué propiedades tienes.

La igualdad de Kraft es similar a la desigualdad de Kraft, a diferencia que cuando el resultado es exactamente 1, se puede decir que existe un código completo con las características utilizadas.

Un ejemplo de código que cumple la igualdad de Kraft es el utilizado en los ejercicios anteriores (5, 6 y 7):

- A: 000
- B: 001
- C: 01
- D: 10
- E: 11

El código es instantáneo, por lo que ningún símbolo del código es el comienzo de otro. Además, es completo, ya que no se pueden representar los símbolos con menos bits de los utilizados, es decir, no se desaprovecha ningún bit para la codificación.

9. Indica qué es un código completo y en qué se diferencia de un código instantáneo, apoyándote en los ejemplos de los dos ejercicios anteriores.

Un código es completo si la suma de las probabilidades de los símbolos existentes es 1.

Se diferencia de un código instantáneo en que el código instantáneo puede no cumplir con el requisito de que la suma de las probabilidades de los símbolos sea 1, pero ningún símbolo de ambos códigos es el comienzo de otro de ese mismo código.

10. Explica detalladamente qué es un código óptimo y qué propiedades cumple. Relaciona los códigos óptimos con el primer teorema fundamental para la codificación sin ruido.

Un código óptimo cumple la condición de que la información que se transmite es igual a la capacidad del canal. Según el Teorema de Shannon para la codificación con ruido, no se puede transmitir a una tasa mayor que C/H. Si esta tasa es 1, el código es óptimo.

Cuando se habla de código óptimo, se refiere al número de elementos por símbolo, es decir, la longitud de los símbolos que se ha diseñado. Es la media del número de elementos por símbolo de todos los símbolos que se envían o, dicho de otra manera, la media de todas las longitudes de los símbolos. Este planteamiento es válido si la probabilidad de que aparezca cada símbolo es igual en todos.

Un código óptimo hace que el número promedio de elementos por símbolos sea mínimo en el tiempo. Esto beneficia en que los símbolos que más se usan tengan longitud menor que los que se utilizan menos.

Relacionándolo con el primer teorema fundamental para la codificación sin ruido, se puede decir que indica el mínimo número de bits que se tienen que utilizar para enviar símbolos por un canal.

11. Considera los siguientes códigos para codificar los mensajes {A, B, C, D, E}:

a) 110, 1110, 0, 100, 1111
b) 111,100,0,101,110
c) 10,110,01,111,00
d) 10,0,110,111,101

Indique qué propiedades cumplen los códigos anteriores y, también cuáles de ellos han podido ser generados mediante los métodos de Shannon-Fano o de Huffman. En caso de encontrar anomalías, indícalas y explica qué repercusiones tienen.

a) Es un código no uniforme, de traducción única, instantáneo, no es completo ($\sum_{i=1}^{N} D^{-ni} = 7/8$). No ha sido generado mediante los métodos de Shannon-Fano ni de Huffman.

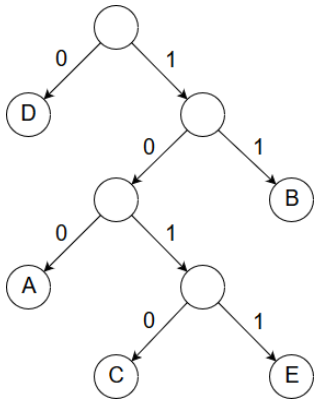
b) Es un código no uniforme, de traducción única, instantáneo, completo ($\sum_{i=1}^{N} D^{-ni} = 1$). No ha sido generado mediante los métodos de Shannon-Fano ni de Huffman.

c) Es un código no uniforme, de traducción única, instantáneo, completo ($\sum_{i=1}^{N} D^{-ni} = 1$). Ha podido ser generado mediante Shannon-Fano o Huffman.

d) Es un código no uniforme, no es de traducción única (10, 10 = 101, 0), no es completo $\sum_{i=1}^{N} D^{-ni} = \frac{9}{8}$). No ha sido generado mediante los métodos de Shannon-Fano ni de Huffman.

12. Se sabe que las longitudes de los códigos para los mensajes m_i en el conjunto {A, B, C, D, E}, tienen longitudes n_i = {3, 2, 4, 1, 4}, respectivamente. Halle un código Huffman compatible con este hecho y, dibuje el árbol de codificación.

Teniendo en cuenta que no se tienen las probabilidades de que aparezca un mensaje concreto, los mensajes que se codifiquen con menor longitud serán más probables, por lo que se encontrarán más cerca de la raíz del árbol de codificación. Se crea un árbol que cumpla con las características dadas:



El código generado a partir del árbol es:

- A: 100
- B: 11
- C: 1010
- D: 0
- E: 1011

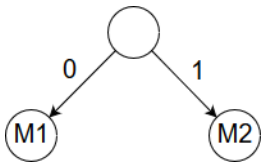
13. Considerando las probabilidades de ocurrencia de los mensajes siguientes, desarrolle un código Shannon-Fano. Explique el algoritmo según construye el código y, dibuje el árbol de codificación:

A	B	C	D	E	F	G	H
1/2	1/4	1/8	1/16	1/32	1/64	1/128	1/128

Finalmente, exponga un ejemplo para codificar la cadena de mensajes {HACED}. Explique también, apoyándose con un ejemplo, cómo decodificar la secuencia codificada.

1. Se ordena por probabilidad decreciente. Ya se encuentra ordenado de esta manera.

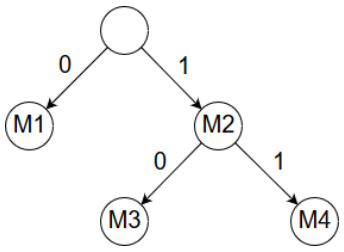
2. Se dividen los mensajes en dos subgrupos, haciendo que la suma de la probabilidad en ambos grupos sea lo más similar posible:



M1 = {A}

M2 = {B, C, D, E, F, G, H}

3. Se repite el proceso 2 con todos los subgrupos que contengan más de un elemento:

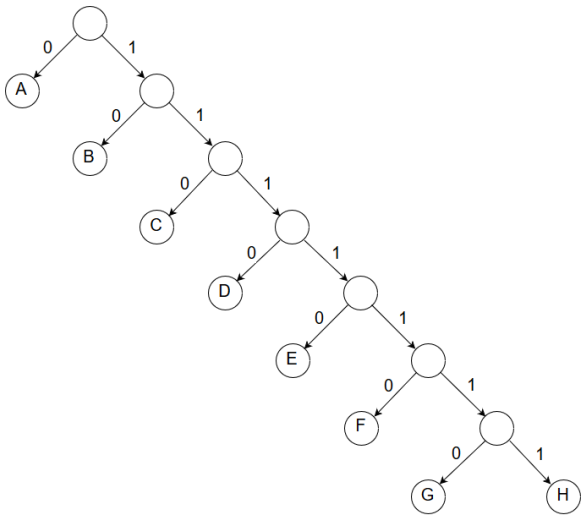


M1 = {A}

M3 = {B}

M4 = {C, D, E, F, G, H}

De la tabla de probabilidades se puede deducir que sea cual sea el subgrupo generado, la probabilidad del primer mensaje del subgrupo es similar a la probabilidad del resto de mensajes del subgrupo, por lo que el árbol de codificación resultante es:



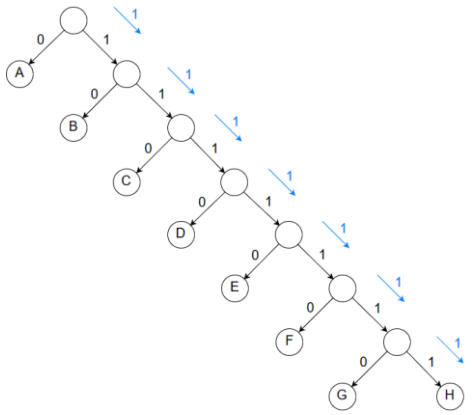
Los mensajes codificados son:

- A: 0
- B: 10
- C: 110
- D: 1110
- E: 11110
- F: 111110
- G: 1111110
- H: 1111111

Para codificar la secuencia de mensajes "HACED" se recorrería cada nodo hoja correspondiente al mensaje que se quiere codificar y el resultado codificado es la concatenación de cada nodo por el que ha pasado hasta llegar al raíz pero invertido:

HACED =11111110110111101110

Para decodificar 11111110110111101110 se comienza desde la raíz y se recorren los nodos dependiendo de lo que indique cada bit. Cuando se llegue a un nodo hoja, se guarda el mensaje decodificado y se comienza desde la raíz de nuevo.



1111111 0110111101110 = **H** 0110111101110; …

14. Considerando las probabilidades de ocurrencia dadas en el ejercicio anterior, desarrolla un código Huffman. Explica el algoritmo según construye el código y, dibuja el árbol de codificación.

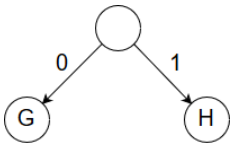
El proceso es iterativo y consiste en dos pasos:

- Ordenar los elementos por probabilidades.
- Agrupar los dos últimos símbolos en un subgrupo donde se suman sus probabilidades.

A	B	C	D	E	F	G	H
1/2	1/4	1/8	1/16	1/32	1/64	1/128	1/128

Dos últimos mensajes/subgrupos: G → 0; H → 1

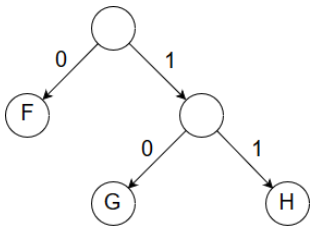
Se agrupan en X_{GH} con probabilidad 1/64.



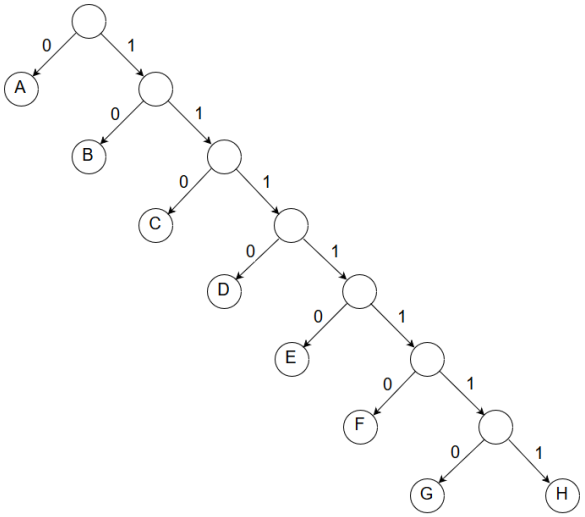
A	B	C	D	E	F	X_{GH}
1/2	1/4	1/8	1/16	1/32	1/64	1/64

Dos últimos mensajes/subgrupos: F → 0; X_{GH} → 1

Se agrupan en X_{FGH} con probabilidad 1/32.



Por las características de la tabla de codificación, el árbol resultante mantiene este mismo patrón, por lo que su resultado final es similar al árbol del ejercicio anterior:



15. Exponga un ejemplo de distribución de probabilidades para los mensajes {A, B, C, D, E, F} que, aplicando el método de Shannon-Fano, no proporcionen un código óptimo. Exponga también el código óptimo que se generaría utilizando codificación Huffman.

16. Atendiendo a los siguientes mensajes y a sus probabilidades de generación por la fuente:

a	b	c	d	e	f	g
0.01	0.24	0.05	0.20	0.47	0.01	0.02

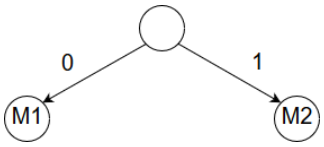
Explica cómo generar un código de Shannon-Fano y crea su árbol de codificación asociado. Explica cómo generar un código Huffman y crea su árbol de codificación asociado. ¿Son igualmente óptimos ambos códigos? Justifica la respuesta. Codifica y decodifica la secuencia de mensajes {bacafeg} con ambos métodos, explicando el procedimiento.

- Shannon-Fano:

- Se ordenan las probabilidades:

e	b	d	c	g	a	f
0.47	0.24	0.20	0.05	0.02	0.01	0.01

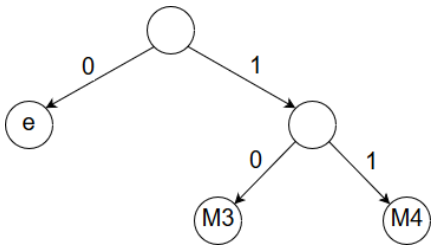
- Se hacen dos subgrupos de probabilidad equiprobables:



M1 = {e}

M2 = {b, d, c, g, a, f}

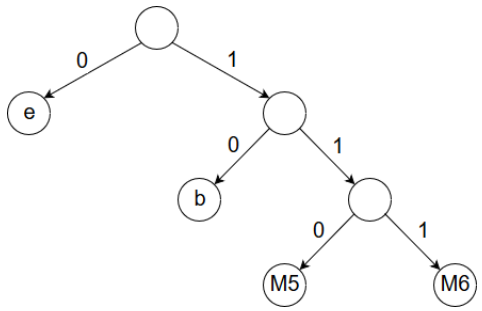
- Se hacen dos subgrupos de M2 de probabilidad equiprobables:



M3 = {b}

M4 = {d, c, g, a, f}

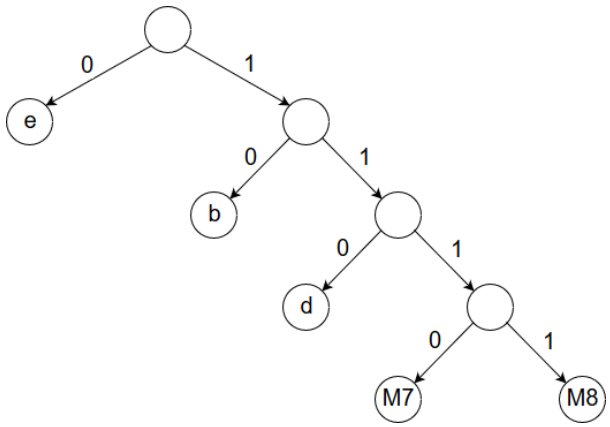
- Se hacen dos subgrupos de M4 de probabilidad equiprobables:



M5 = {d}

M6 = {c, g, a, f}

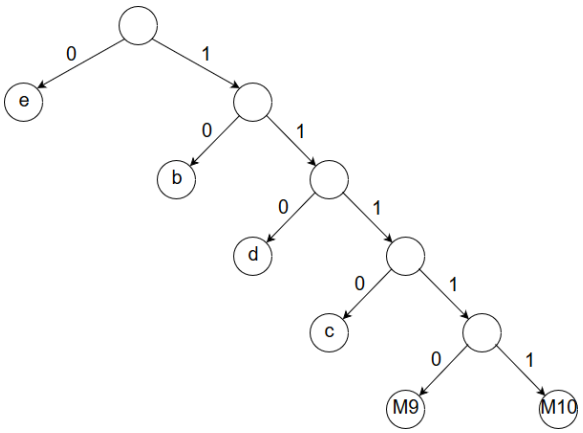
- Se hacen dos subgrupos de M6 de probabilidad equiprobables:



M7 = {c}

M8 = {g, a, f}

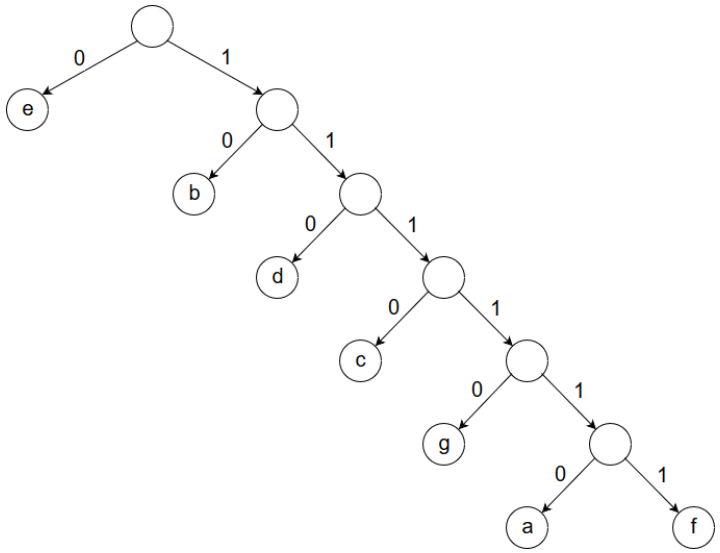
- Se hacen dos subgrupos de M8 de probabilidad equiprobables:



M9 = {g}

M10 = {a, f}

- Se hacen dos subgrupos de M10 de probabilidad equiprobables y se finaliza el árbol de codificación:



M11 = {a}

M12 = {f}

Codificación:

- e: 0
- b: 10
- d: 110
- c: 1110
- g: 11110
- a: 111110
- f: 111111

- Huffman:

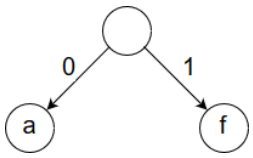
- Se ordenan las probabilidades:

e	b	d	c	g	a	f
0.47	0.24	0.20	0.05	0.02	0.01	0.01

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{af} = 0.02$$

- El árbol resultante es:



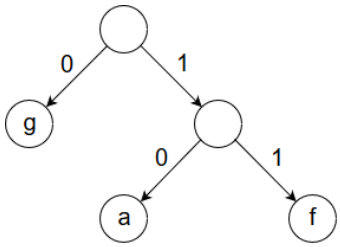
- Se ordenan las probabilidades:

e	b	d	c	g	X_{af}
0.47	0.24	0.20	0.05	0.02	0.02

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{gaf} = 0.04$$

- El árbol resultante es:



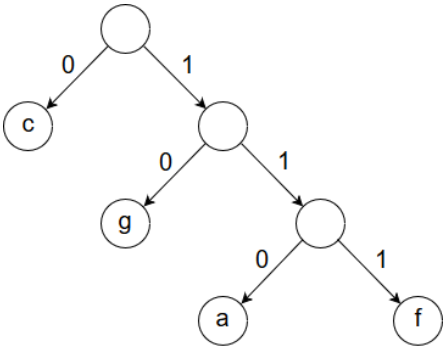
- Se ordenan las probabilidades:

e	b	d	c	X_{gaf}
0.47	0.24	0.20	0.05	0.04

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{cgaf} = 0.09$$

- El árbol resultante es:



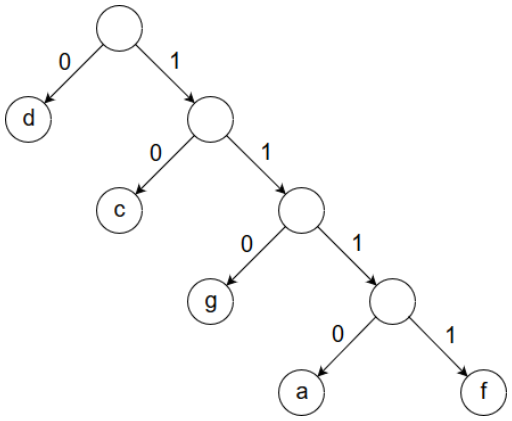
- Se ordenan las probabilidades:

e	b	d	X_{cgaf}
0.47	0.24	0.20	0.09

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{dcgaf} = 0.29$$

- El árbol resultante es:



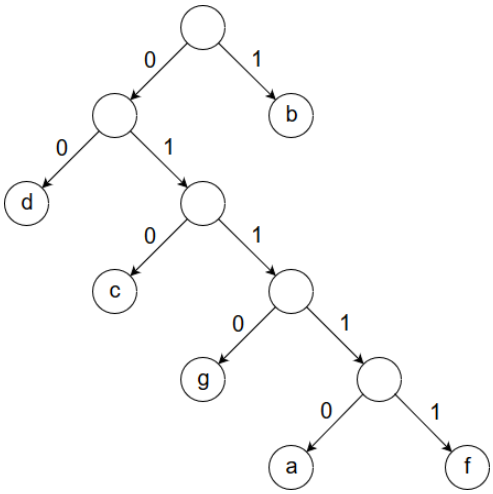
- Se ordenan las probabilidades:

e	X_{dcgaf}	b
0.47	0.29	0.24

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{bdcgaf} = 0.53$$

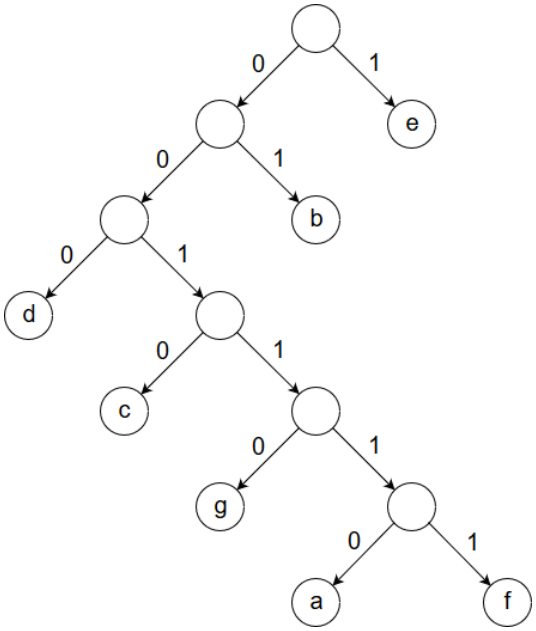
- El árbol resultante es:



- Se ordenan las probabilidades y se termina el ciclo al quedar solo dos grupos:

X_{bdcgaf}	e
0.53	0.47

- El árbol resultante es:



Codificación:

- e: 1
- b: 01
- d: 000
- c: 0010
- g: 00110
- a: 001110
- f: 001111

Los árboles resultantes de hacer Shannon-Fano y Huffman son similares. Pese a que la codificación de los mensajes sea diferente, la estructura de los árboles es similar. Como Huffman asegura que se obtiene un código óptimo y la estructura de ambos árboles es similar, se puede decir que ambos son igualmente óptimos.

- Codificar "bacafeg" con Shannon-Fano:

  Como se ha explicado en el ejercicio 13, al tener el valor codificado de cada mensaje, se sustituye cada uno por el correspondiente:

  bacafeg = 10111110111011111101111111011110

- Decodificar 10111110111011111101111111011110:

  Al igual que en el ejercicio 13, se recorre el árbol desde el nodo raíz hasta un nodo hoja. En ese momento, se guarda el mensaje y se continúa desde el nodo raíz de nuevo.

- Codificar "bacafeg" con Huffman:

  Como se ha explicado en el apartado anterior, al tener el valor codificado de cada mensaje, se sustituye cada uno por el correspondiente:

  bacafeg = 0100111000100011100011111100110

- Decodificar 0100111000100011100011111100110:

  Al igual que en el apartado anterior, se recorre el árbol desde el nodo raíz hasta un nodo hoja. En ese momento, se guarda el mensaje y se continúa desde el nodo raíz de nuevo.

17. Sea una fuente S capaz de generar 4 símbolos, con las siguientes probabilidades:

S = [P(A) = 0.4; P(B) = 0.3; P(C) = 0.2; P(D) = 0.1]

Indica, entre los siguientes códigos, cuáles son instantáneos, cuáles unívocamente decodificables, cuáles completos y cuáles tienen mejor rendimiento (H/longitud promedia):

a) Código 1: A = 001; B = 01; C = 11; D = 010
b) Código 2: A = 0; B = 01; C = 011; D = 111
c) Código 3: A = 1; B = 01; C = 001; D = 0001

a) No es instantáneo (B, D), no es unívocamente decodificable (DB = 01001, BA = 01001), no es completo ($\sum_{i=1}^{N} D^{-ni} = 3/4$).
b) No es instantáneo (A, B, C), es unívocamente decodificable, es completo ($\sum_{i=1}^{N} D^{-ni} = 1$).
c) Es instantáneo, es unívocamente decodificable, no es completo ($\sum_{i=1}^{N} D^{-ni} = 15/16$).

El que mejor rendimiento tiene es el "b", después el "c" y finalmente el "a".

18. Explica en qué consiste en algoritmo de compresión Run-Length. Explica cómo se comprimiría la siguiente secuencia binaria: 00100001011000000000000100111111000101.

Consiste en buscar patrones de datos que sean frecuentes. A esos patrones frecuentes se le asigna un código.

Run-Length busca patrones a nivel de bit. Busca cuantos bits iguales hay juntos y crea una tupla con el número de bits iguales y si es 0 o 1.

Para descomprimir la secuencia 00100001011000000000000100111111000101 se recorre bit a bit y los que son iguales y se encuentran juntos se sustituye por una tupla indicando el bit que es y cuantas veces se repite:

00100001011000000000000100111111000101: (2,0), (1,1), (4,0), (1,1), (1,0), (2,1), (13,0), (1,1), (2,0), (6,1), (3,0), (1,1), (1,0), (1,1).

19. Explica cuáles son los fundamentos de los métodos de compresión de datos basados en diccionario.

Se utiliza una tabla Hash, asociando una clave a cada valor. La clave es el patrón conseguido, el valor es el valor que le sigue.

La idea de los métodos de compresión de datos basados en diccionario consiste en dividir la cadena de mensajes en frases. Cada frase se forma por otra frase que ya ha sido utilizada anteriormente más el símbolo correspondiente.

20. Explica, utilizando un ejemplo, cómo funciona el algoritmo de compresión de datos LZ78 para comprimir y descomprimir.

Cadena: 100100010

Para comprimir se divide en frases: 1 | 0 | 01 | 00 | 010

Para descomprimir se numeran las frases comenzando en 1, ya que el símbolo vacío es el 0.

El código anterior se puede convertir en: (0,1), (0,0), (2,1), (2,0), (3,0).

- (0,1): se utiliza en símbolo vacío y se le añade 1:

Entrada	Código
1	1

La cadena codificada de momento es: 1.

- (0,0): se utiliza el símbolo vacío y se le añade 0:

Entrada	Código
1	1
2	0

La cadena codificada de momento es: 10.

- (2,1): se utiliza la segunda entrada y se le añade 1:

Entrada	Código
1	1
2	0
3	01

La cadena codificada de momento es: 1001.

- (2,0): se utiliza la segunda entrada y se le añade 0:

Entrada	Código
1	1
2	0
3	01
4	00

La cadena codificada de momento es: 100100.

- (3,0): se utiliza la tercera entrada y se le añade 0:

Entrada	Código
1	1
2	0
3	01
4	00
5	010

La cadena codificada finalmente es: 100100010.

21. Sea una fuente S capaz de generar 4 símbolos, con las siguientes probabilidades:

S = [P(A) = 0.4; P(B) = 0.3; P(C) = 0.2; P(D) = 0.1]

Esta fuente genera la siguiente secuencia de mensajes: {AABAAADADDAAAAAB}.

a) Desarrolla un código uniforme para codificar los mensajes de S. Demuestra si el código es completo. Explica como codificar la secuencia de mensajes dada con este código y, exponlo como ejemplo. Explica también el método de decodificación, haciendo uso del mensaje codificado para su decodificación.

b) Desarrolla un código Huffman para codificar los mensajes de S. Demuestra si el código es completo. Explica cómo codificar la secuencia de mensajes dada con este código y, exponlo como ejemplo. Explica también el método de decodificación, haciendo uso del mensaje codificado para su decodificación. Indica, en comparación con el método de codificación uniforme, cuál es la variación de eficiencia de ambos códigos.

a) Código uniforme:
   - A: 00
   - B: 01
   - C: 10
   - D: 11

El código es completo: $\sum_{i=1}^{N} D^{-ni} = 2^{-2} + 2^{-2} + 2^{-2} + 2^{-2} = 1$.

Codificación:
Como ya se tiene la codificación correspondiente a cada mensaje, solo es necesario sustituir cada mensaje por su correspondiente codificado. El resultado es:

00000100000011001111000000000001

Decodificación:
Al conocer que es un código uniforme, se sabe que todos los mensajes están codificados con 2 bits, por lo que el mensaje completo codificado se recorre de dos en dos y se sustituye cada valor codificado por el mensaje que le corresponde:

00(A)00(A)01(B)00(A)00(A)00(A)11(D)00(A)11(D)11(D)00(A)00(A)00(A)00(A)0
0(A)01(B)

b) Código Huffman:

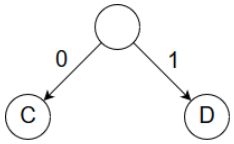
- Se ordenan las probabilidades:

A	B	C	D
0.4	0.3	0.2	0.1

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{cd} = 0.3$$

- El árbol resultante es:



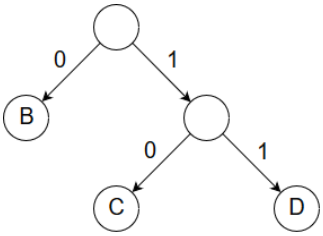
- Se ordenan las probabilidades:

A	B	X_{cd}
0.4	0.3	0.3

- Se agrupan los dos últimos y su probabilidad será la suma de ambos:

$$X_{bcd} = 0.6$$

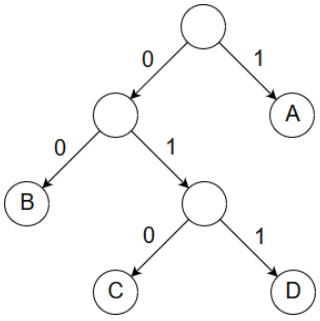
- El árbol resultante es:



- Se ordenan las probabilidades y se finaliza al quedar solo dos mensajes/grupos:

X_{bcd}	A
0.6	0.4

- El árbol resultante es:



Código resultante:

- A: 1
- B: 00
- C: 010
- D: 011

El código es completo, ya que los códigos Huffman son completos. Pese a eso, se puede comprobar: $\sum_{i=1}^{N} D^{-ni} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-3} = 1$.

Codificación:

Para codificar se debe recorrer cada mensaje de la secuencia que se quiere codificar, comenzando por el nodo hoja del mensaje en cuestión hasta llegar a la raíz. El resultado será la concatenación de todas las ramas por las que ha pasado invertidas. Como ya se ha representado el código resultante, se sustituye cada mensaje por su correspondiente codificado. El resultado es:

110011101110110111111100

Decodificación:

Al ser un código completo, se sabe que es un código instantáneo y se sabe que es unívocamente decodificable, por lo que el proceso para su decodificación es recorrer bit a bit el mensaje codificado y desplazarse por el lugar correspondiente del árbol. Cuando se llegue a un nodo hoja, se guarda el mensaje y se comienza de nuevo por la raíz.

- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 0: Rama izquierda; 0: Rama izquierda llega a nodo hoja → B
- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 0: Rama izquierda; 1: Rama derecha; 1: Rama derecha: D
- 1: Rama derecha llega a nodo hoja → A
- 0: Rama izquierda; 1: Rama derecha; 1: Rama derecha: D
- 0: Rama izquierda; 1: Rama derecha; 1: Rama derecha: D
- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 1: Rama derecha llega a nodo hoja → A
- 0: Rama izquierda; 0: Rama izquierda llega a nodo hoja → B

Eficiencia:

- Código uniforme:

$$\bar{n} = \sum_{i=1}^{n} p(mi) * ni = 0.4 * 2 + 0.3 * 2 + 0.2 * 2 + 0.1 * 2 = 2$$

- Código Huffman:

$$\bar{n} = \sum_{i=1}^{n} p(mi) * ni = 0.4 * 1 + 0.3 * 2 + 0.2 * 3 + 0.1 * 3 = 1.9$$

22. Desarrolla el método de compresión LZ78 para comprimir la secuencia de mensajes dada en el ejercicio anterior. Descomprime el mensaje, explicando cada paso. Indica una estimación de la eficiencia (en número de bits que ocupa el total de la secuencia de mensajes) con respecto al método de Huffman desarrollado en el ejercicio anterior.

Código: 1100111011101101111111100

Para comprimir se divide en frases: 1 | 10 | 0 | 11 | 101 | 110 | 1101 | 111 | 1110 | 0

El código anterior se comprime en: (0,1), (1,0), (0,0), (1,1), (2,1), (4,0), (6,1), (4,1), (8,0), (0,0).

Para descomprimir se numeran las frases comenzando en 1, ya que el símbolo vacío es el 0:

- (0,1): se utiliza en símbolo vacío y se le añade 1:

Entrada	Código
1	1

La cadena codificada de momento es: 1.

- (1,0): se utiliza el primero y se le añade 0:

Entrada	Código
1	1
2	10

La cadena codificada de momento es: 110.

- (0,0): se utiliza el símbolo vacío y se le añade 0:

Entrada	Código
1	1
2	10
3	0

La cadena codificada de momento es: 1100.

- (1,1): se utiliza la primera entrada y se le añade 1:

Entrada	Código
1	1
2	10
3	0
4	11

La cadena codificada de momento es: 110011.

- (2,1): se utiliza la segunda entrada y se le añade 1:

Entrada	Código
1	1
2	10
3	0
4	11
5	101

La cadena codificada de momento es: 110011101.

- (4,0): se utiliza la cuarta entrada y se le añade 0:

Entrada	Código
1	1
2	10
3	0
4	11
5	101
6	110

La cadena codificada de momento es: 110011101110.

- (6,1): se utiliza la sexta entrada y se le añade 1:

Entrada	Código
1	1
2	10
3	0
4	11
5	101
6	110
7	1101

La cadena codificada de momento es: 1100111011101101.

- (4,1): se utiliza la cuarta entrada y se le añade 1:

Entrada	Código
1	1
2	10
3	0
4	11
5	101
6	110
7	1101
8	111

La cadena codificada de momento es: 11001110111011011111.

- (8,0): se utiliza la octava entrada y se le añade 0:

Entrada	Código
1	1
2	10
3	0
4	11
5	101
6	110
7	1101
8	111
9	1110

La cadena codificada de momento es: 1100111011110110111111110.

- (0,0): se utiliza el símbolo vacío y se le añade 0. Como ya está en el diccionario, no se inserta:

Entrada	Código
1	1
2	10
3	0
4	11
5	101
6	110
7	1101
8	111
9	1110

La cadena codificada de momento es: 110011101110110111111100.

Se termina la descompresión y la cadena codificada por el código Huffman es:

110011101110110111111100

El número de símbolos que se envía con el código Huffman es 24.

El número de símbolos que se envía si se comprime con LZ78 es 20.

Si se comprime con LZ78 se consigue una reducción del 17% de los símbolos enviados.