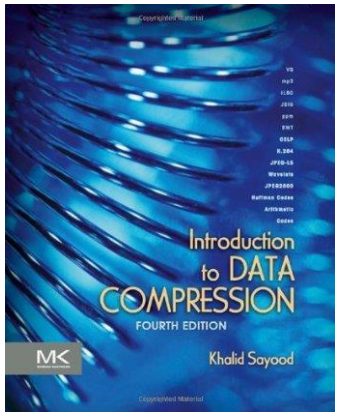
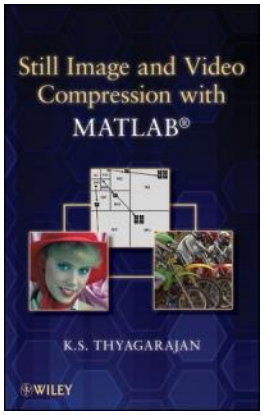
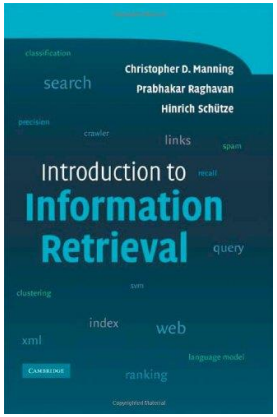
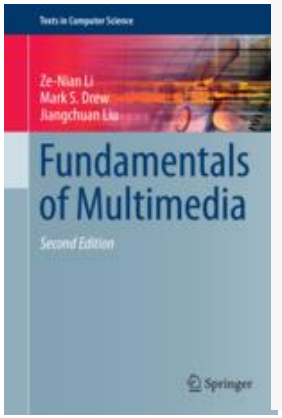








Tema 3: Codificación por Longitud de Ráfagas, Run Length Encoding (RLE)

Rafael Molina

Depto de Ciencias de la Computación

e Inteligencia Artificial

Universidad de Granada

Contenidos

I. Codificación RLE

Se utiliza para fuentes que emiten "ráfagas" de símbolos (letras del alfabeto) idénticos.

Ejemplo: 00000000111000000011111111000000011111100000

Ráfagas: 00000000, 111, 0000000, 11111111, 0000000, 1111111, 00000

¿Podrías poner un ejemplo donde las ráfagas aparecen de una forma natural?

- Para fijar conceptos, vamos a suponer, salvo que establezcamos lo contrario, que queremos codificar símbolos de un alfabeto que no tiene más de 256 elementos distintos.

- Suponemos que nuestra fuente genera ráfagas, de una cierta longitud, de símbolos. También por simplicidad suponemos que la longitud de la ráfaga no es superior a 256.

- La codificación RLE (run length encoding) cuenta el número de veces que un símbolo aparece repetido de forma sucesiva.
- Cada ráfaga codificada tiene dos partes
  - Run o Run Byte: indica el número de repeticiones consecutivas
  - Símbolo: valor que se repite
- Así pues tendremos pares de longitud fija de la forma

{número de repeticiones, símbolo}

Ejemplo

Queremos codificar la secuencia

AAABBCCAAA

Su codificación sería

{3,A}{2,B}{2,C}{3,A}

I.1 Ejemplo: El formato Windows BMP

BitMap (BMP) es un formato estándar de imágenes de Microsoft Windows. Soporta diferentes formatos de imágenes: color indexado (8 bits por pixel), e imágenes en color con 16, 24 y 32 bits.

Utiliza RLE y puede comprimir con pérdida imágenes en color de 24 bits de forma eficiente.

BMP admite también almacenamiento sin pérdida. En particular, las imágenes de 16 y 32 bits (con el canal alfa) son siempre sin pérdida.

Veamos una breve descripción de BMP.

El formato BMP de 8 bits utiliza una forma de RLE en la que los datos son representados en dos modos: codificados y absolutos, ambos pueden ocurrir en una imagen

- Modo codificado: utiliza una representación RLE de 2 bytes, el primero especifica el número de veces consecutivas que tienen el mismo índice de color. El índice de color se almacena en el segundo byte.
- Modo absoluto: el primer byte es cero y el segundo señala una de las cuatros condiciones siguientes

Valor del segundo Byte	Condición	Valor del segundo Byte	Condición
0	Fin de línea	2	Desplazamiento a nueva posición
1	Fin de imagen	3-255	Especifica píxeles individualmente

- Si el segundo byte es dos, los dos bytes siguientes contienen desplazamientos horizontales y verticales a una nueva posición espacial en la imagen
- Si el segundo byte está en [3,255], especifica el número de píxeles sin comprimir que siguen (los siguientes bytes contienen los índices de color)

La codificación de ráfagas con un solo elemento es costosa ya que usar codificaciones de la forma

{1,símbolo_del_alfabeto}

no produce ninguna compresión, al revés expansión.

¿cómo codificamos eficientemente secuencias de la forma

abcddddddddcbbbbbbabcdef ?

Su codificación usando RLE sería

{1,a}{1,b}{1,c}{9,d}{1,c}{6,b}{1,a}{1,b}{1,c}{1,d}{1,e}{1,f}

que casi no se comprime si usamos RLE: pasa de 25 bytes a 24 bytes. ¿Cómo solucionamos el problema?

II RLE usando PackBits

Vamos a usar el primer bit para saber si tenemos que codificar una longitud de ráfaga (símbolo comprimido) o una secuencia de símbolos distintos que sólo aparecen una vez.

Usaremos una representación signo-magnitud:

11111111=-1111111=-127
10000001=-1000000=-1
10000000=-0000000=-0
01111111=+1111111=+127
00000000=+0000000=+0

Algoritmo PackBits (unpack)

Lee el siguiente byte de la entrada y almacénalo en n

- Si +0≤n≤+127 (símbolos no comprimidos)
  - Copia los n+1 bytes literales que siguen
- Si -127≤n≤-1 (símbolo comprimido contiene la longitud de la ráfaga)
  - Copia el siguiente byte 1-n veces
- Si n= -0
  - No hagas nada

Ejemplo

La cadena

abcddddddddddcbbbbbbbabcdef

Se codificaría usando Packbits en

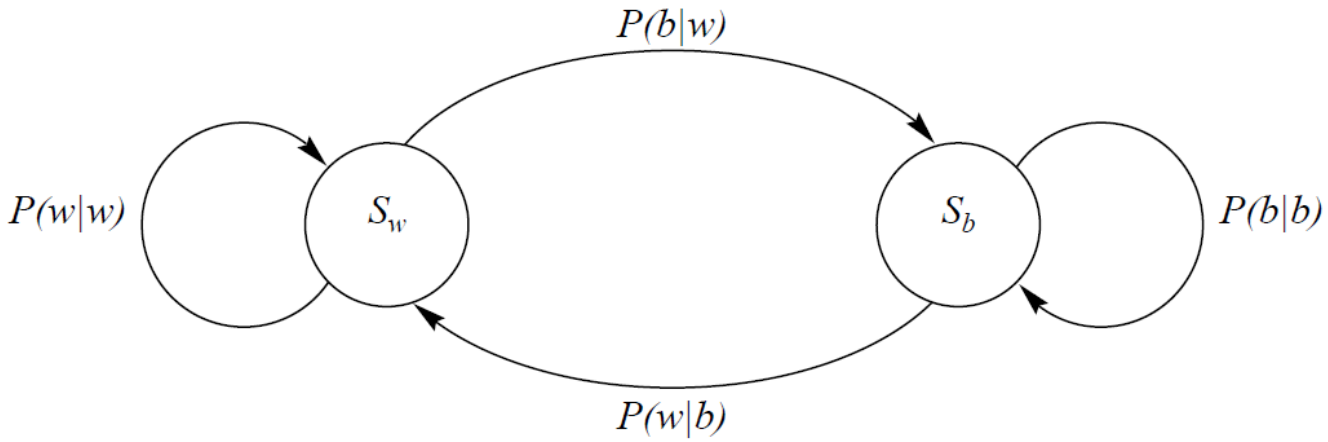
{2,a,b,c}{-8,d}{0,c}{-5,b}{5,a,b,c,d,e,f}

RLE usa ahora 17 bytes.

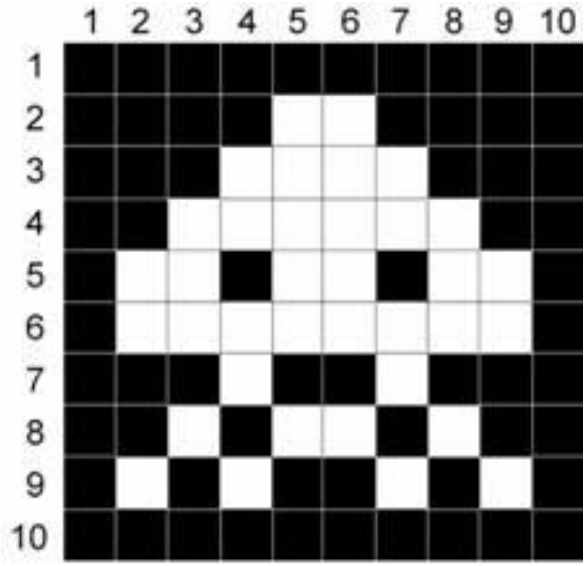
II. Codificación de datos binarios usando RLE

El modelo que da origen a la codificación RLE es el modelo de Capon, un modelo de Markov con dos estados S_w y S_b (S_w corresponde a que el píxel que acabamos de codificar es blanco y S_b a que es negro).

Las probabilidades $P(S_b)$, $P(S_w)$ y $P(w|w)$ y $P(w|b)$ determinan el sistema completamente



Cabe esperar que P(b|w) y P(w|w) correspondan a una distribución muy descompensada y probablemente P(w|b) y P(b|b) también lo sea.



Básicamente decimos que una vez que un píxel es blanco (o negro) es altamente probable que siga en ese estado.

Podemos codificar simplemente la longitud de las rachas sabiendo codificador y decodificador si la primera racha codificada corresponde a negros o blancos.

Si comenzamos por un negro la codificación sería

14, 2, 7, …

Si comenzamos con blanco

0, 14,2,7,….

Esta estrategia se usa en el estándar de compresión de fax ITU-T T.4 (CCITT Group 3)

¿Cómo codificamos la longitud de las ráfagas?

1. FLC (código de longitud fija)
2. VLC (código de longitud variable, en particular, Huffman)

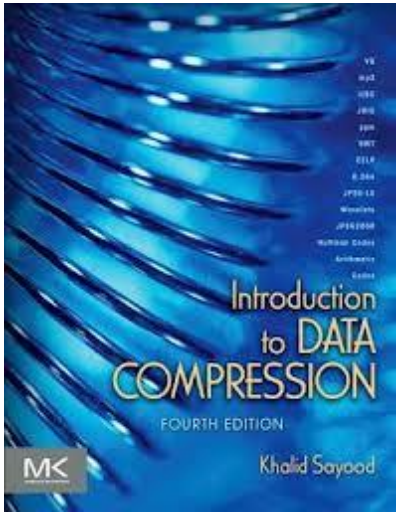
Dentro del código de Huffman, podemos usar el código de Huffman truncado que describiremos a continuación.

- La idea del código de Huffman truncado es reducir la tabla de códigos y la longitud máxima del código de Huffman utilizando sólo los símbolos más probables.
    - Combinamos los J símbolos menos probables de un alfabeto de tamaño M en un símbolo auxiliar ESC
    - Usamos Huffman con los M-J símbolos más probables y ESC
    - Cuando codifiquemos ESC le añadimos, de los J símbolos asociados a ESC, el que ha aparecido.

El código de Huffman truncado aumenta la longitud media del código, pero resulta en una mejor relación complejidad/eficiencia con una elección correcta de J.

VIII Bibliografía

K. Sayood, "Introduction to Data Compression", Morgan and Kaufmann, 2012.



Apuntes de Prof. Luis Torres, Codificación de Contenidos Audiovisuales, Escuela Técnica Superior de Ingeniería de Telecomunicación de Barcelona, Universidad Politécnica de Catalunya

Apuntes de Prof. J. Mateos, Laboratorio Multimedia, Escuela Técnica Superior de Ingenierías de Informática y Telecomunicación, Universidad de Granada