Algoritmo de compresión de Huffman

Generalidades

Se trata de un algoritmo que puede ser usado para compresión o encriptación de datos.

Este algoritmo se basa en asignar códigos de distinta longitud de bits a cada uno de los caracteres de un MENSAJE. Si se asignan códigos más cortos a los caracteres que aparecen más a menudo se consigue una compresión del MENSAJE.

Esta compresión es mayor cuando la variedad de caracteres diferentes que aparecen es menor. Por ejemplo: si el texto se compone únicamente de números o mayúsculas, se conseguirá una compresión mayor.

Para recuperar el MENSAJE original es necesario conocer el código asignado a cada carácter, así como su longitud en bits, si ésta información se omite, y el receptor del MENSAJE la conoce, podrá recuperar la información original. De este modo es posible utilizar el algoritmo para encriptar MENSAJES.

Mecanismo del algoritmo

- Contar cuantas veces aparece cada carácter en el MENSAJE a comprimir o a encriptar. Y crear una lista enlazada con la información de caracteres y frecuencias.
- Ordenar la lista de menor a mayor en función de la frecuencia.
- Convertir cada elemento de la lista en un árbol.
- Fusionar todos estos árboles en uno único, para hacerlo se sigue el siguiente proceso, mientras la lista de árboles contenga más de un elemento:
    - Con los dos primeros árboles formar un nuevo árbol, cada uno de los árboles originales en una rama.
    - Sumar las frecuencias de cada rama en el nuevo elemento árbol.
    - Insertar el nuevo árbol en el lugar adecuado de la lista según la suma de frecuencias obtenida.
- Para asignar el nuevo código binario de cada carácter sólo hay que seguir el camino adecuado a través del árbol. Si se toma una rama cero (izquierda), se añade un cero al código, si se toma una rama uno (derecha), se añade un uno.
- Se recodifica el MENSAJE según los nuevos códigos.

Veamos un ejemplo

Tomemos un texto corto, por ejemplo:

"ata la jaca a la estaca"

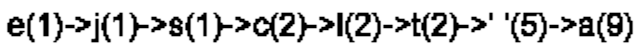
1) Contamos las veces que aparece cada carácter y hacemos una lista enlazada:

' '(5), a(9), c(2), e(1), j(1), l(2), s(1), t(2)

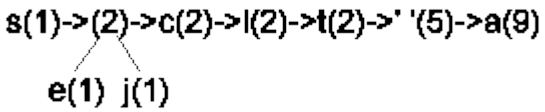
2) Ordenamos por frecuencia de menor a mayor

e(1), j(1), s(1), c(2), l(2), t(2), ' '(5), a(9)

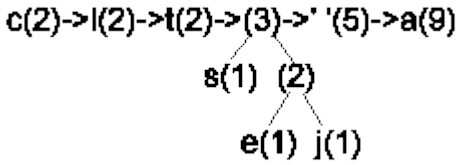
3) Consideremos ahora que cada elemento es el nodo raíz de un árbol.



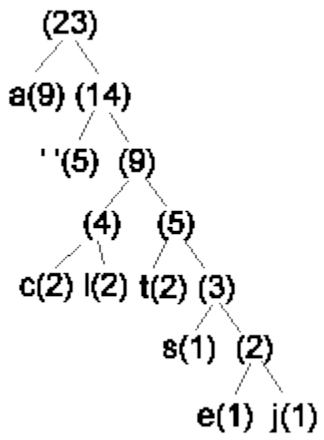
4) Unimos los dos primeros nodos (árboles) en un nuevo árbol, sumamos sus frecuencias y lo colocamos en el lugar correspondiente:



Y así sucesivamente:



El resultado final es:



5) Asignamos los códigos, las ramas a la izquierda son ceros, y a la derecha unos (por ejemplo), es una regla arbitraria.

a	' '	c	l	t	s	e	j
0	10	1100	1101	1110	11110	111110	111111

6) Y traducimos el texto:

a	t	a	' '	l	a	' '	j	a	c	a	' '	a	' '	l	a	' '	e	s	t	a	c	a
0	1110	0	10	1101	0	10	111111	0	1100	0	10	0	10	1101	0	10	111110	11110	1110	0	1100	0

Y sólo queda empaquetar los bits en grupos de ocho, es decir en bytes (código BCD):

01110010	11010101	11111011	00010010	11010101	11110111	10111001	10000000
0x72	0xD5	0xFB	0x12	0xD5	0xF7	0xb9	0x80

Ese código de bytes (hexadecimal), se envía al receptor el cual deberá tener el árbol (llave) para poder comprender (descomprimir) el MENSAJE.