Algoritmo de Huffman

El **algoritmo de Huffman** es un algoritmo para la construcción de códigos de Huffmann, desarrollado por David A. Huffmann en 1952 y descrito en "A Method for the Construction of Minimum-Redundancy Codes".

Este algoritmo toma un alfabeto de *n* símbolos, junto con sus frecuencias de aparición asociadas, y produce un código de Huffman para ese alfabeto y esas frecuencias.

Contenidos:

1. Descripción

El algoritmo consiste en la creación de un árbol binario que tiene cada uno de los símbolos por hoja, y construido de tal forma que siguiéndolo desde la raíz a cada una de sus hojas se obtiene el código Huffman asociado.

1. Se crean varios árboles, uno por cada uno de los símbolos del alfabeto, consistiendo cada uno de los árboles en un nodo sin hijos, y etiquetado cada uno con su símbolo asociado y su frecuencia de aparición.
2. Se toman los dos árboles de menor frecuencia, y se unen creando un nuevo árbol. La etiqueta de la raíz será la suma de las frecuencias de las raíces de los dos árboles que se unen, y cada uno de estos árboles será un hijo del nuevo árbol. También se etiquetan las dos ramas del nuevo árbol: con un 0 la de la izquierda, y con un 1 la de la derecha.
3. Se repite el paso 2 hasta que sólo quede un árbol.

Con este árbol se puede conocer el código asociado a un símbolo, así como obtener el símbolo asociado a un determinado código.

Para obtener el código asociado a un símbolo se debe proceder del siguiente modo:

1. Comenzar con un código vacío
2. Iniciar el recorrido del árbol en la hoja asociada al símbolo
3. Comenzar un recorrido del árbol hacia arriba
4. Cada vez que se suba un nivel, añadir al código la etiqueta de la rama que se ha recorrido
5. Tras llegar a la raíz, invertir el código
6. El resultado es el código Huffman deseado

Para obtener un símbolo a partir de un código se debe hacer así:

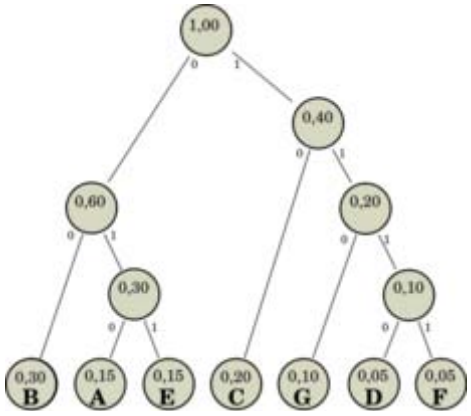
1. Comenzar el recorrido del árbol en la raíz de éste
2. Extraer el primer símbolo del código a descodificar
3. Descender por la rama etiquetada con ese símbolo

4. Volver al paso 2 hasta que se llegue a una hoja, que será el símbolo asociado al código

En la práctica, casi siempre se utiliza el árbol para obtener todos los códigos de una sola vez; luego se guardan en tablas y se descarta el árbol.

1. 1. Ejemplo de uso

La tabla describe el alfabeto a codificar, junto con las frecuencias de sus símbolos. En el gráfico se muestra el árbol construido a partir de este alfabeto siguiendo el algoritmo descrito.



Árbol para construir el código Huffman del ejemplo
SímboloFrecuencia A0,15 B0,30 C0,20 D0,05 E0,15 F0,05 G0,10

Se puede ver con facilidad cuál es el código del símbolo **E**: subiendo por el árbol se recorren ramas etiquetadas con **1**, **1** y **0**; por lo tanto, el código es **011**. Para obtener el código de **D** se recorren las ramas **0**, **1**, **1** y **1**, por lo que el código es **1110**.

La operación inversa también es fácil de realizar: dado el código **10** se recorren desde la raíz las ramas **1** y **0**, obteniéndose el símbolo **C**. Para descodificar **010** se recorren las ramas **0**, **1** y **0**, obteniéndose el símbolo **A**.

2. Limitaciones

Para poder utilizar el algoritmo de Huffman es necesario conocer de antemano las frecuencias de aparición de cada símbolo, y su eficiencia depende de lo próximas a las frecuencias reales que sean las estimadas. Algunas implementaciones del algoritmo de Huffman son adaptativas, actualizando las frecuencias de cada símbolo conforme recorre el texto.

La eficiencia de la codificación de Huffman también depende del balance que exista entre los hijos de cada nodo del árbol, siendo más eficiente conforme menor sea la diferencia de frecuencias entre los dos hijos de cada nodo.

Ejemplos:

- La codificación binaria es un caso particular de la codificación de Huffman que ocurre cuando todos los símbolos del alfabeto tienen la misma frecuencia. Se tiene pues que la codificación binaria es la más eficiente para cualquier número de símbolos equiprobables.
- El algoritmo de Huffman aplicado sobre un alfabeto de dos símbolos asignará siempre un 1 al primero y un 0 al segundo, independientemente de la frecuencia de aparición de dichos símbolos. En este caso nunca se realiza compresión de los datos, mientras que otros algoritmos sí podrían conseguirlo.

Una manera de resolver este problema consiste en agrupar los símbolos en palabras antes de ejecutar el algoritmo. Por ejemplo, si se tiene la cadena de longitud 64

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAB

El algoritmo de Huffman aplicado únicamente a los símbolos devuelve el código:

11111111111111111111111111111111111111111111111111111111111110

También de longitud 64. Sin embargo, si antes de utilizar el algoritmo, se agrupan los símbolos en las palabras **"AA"**, **"AB"** y **"B"** (que se codifican como 1, 01 y 00), el algoritmo devuelve la siguiente cadena:

11111111111111111111111111111101

que tiene longitud 32, la mitad que si no se hubiera agrupado. Si observa el árbol de Huffman, se puede comprobar que la diferencia de frecuencias entre las ramas del árbol es menor que en el caso anterior.

3. Variaciones del algoritmo

3. 1. Códigos Huffman *n* -arios

Es posible crear códigos de Huffman ternarios, cuaternarios, y, en general, *n*-arios. Para ello sólo es necesario realizar dos modificaciones al algoritmo:

1. Los árboles a crear tendrán tantos hijos como símbolos posibles puedan aparecer en los códigos Huffman. Por ejemplo, si es ternario se crearán árboles con tres hijos; si es cuaternario, con cuatro.
2. Si se expresa como *s* el número de símbolos en el alfabeto a codificar, y *n* el número de símbolos que aparecen en el código Huffman, entonces *s*-1 debe ser múltiplo de *n*-1. Es decir, para un código ternario, *s* debe valer 3, 5, 7, etc. Si esta condición no se cumple, entonces se deben añadir símbolos "nulos" con frecuencia 0, que servirán sólo como relleno a la hora de construir el árbol.