MATLAB: CODE DE HUFFMAN

Voici une description détaillée de la fonction définie dans **huffman.m** (**huffman2.m** ne contient que la version binaire du même algorithme). Il s'agit d'un algorithme de calcul de l'arbre de Huffman pour une cardinalité D donnée. Pour pouvoir produire les mots de code associés à chacune des probabilités en entrée, la fonction se sert d'un indice unique 2^N où $N \in [0, Np-1]$ où Np est le nombre total de probabilités du vecteur p donné (Ce nombre est calculé après ajustement par l'ajout de probabilités fantômes ($p = 0$) dans le but de rendre l'arbre optimal).

Un vecteur appelé q contient alors le vecteur des probabilités et celui des indices dans une matrice telle que :

$$q = \begin{bmatrix} p_1 & 2^0 \\ \vdots & \vdots \\ p_{Np} & 2^{Np-1} \end{bmatrix} \tag{1}$$

À chaque itération, le vecteur q est trié selon les probabilités qu'il contient (de ce fait les indices suivent toujours leur probabilité respective). Il fait ensuite la somme des D plus petites probabilités de même que la somme des indices correspondants. Le nouveau vecteur indice obtenu (qui est trié avant) est alors ajouté à la matrice ind qui donne au bout de k itérations une matrice complète des indices.

Exemple no.1

On veut calculer l'arbre de Huffman binaire du vecteur de probabilités p :

$$p = \left[\tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{8}, \tfrac{1}{16}, \tfrac{1}{32}, \tfrac{1}{64}, \tfrac{1}{64}\right] \tag{2}$$

Le vecteur q initial sera donc :

$$q = \begin{bmatrix} \frac{1}{64} & 1 \\ \frac{1}{64} & 2 \\ \frac{1}{32} & 4 \\ \frac{1}{16} & 8 \\ \frac{1}{8} & 16 \\ \frac{1}{4} & 32 \\ \frac{1}{2} & 64 \end{bmatrix} \tag{3}$$

Après une itération complète, nous aurons que le vecteur q sera :

$$q = \begin{bmatrix} \frac{1}{32} & 3 \\ \frac{1}{32} & 4 \\ \frac{1}{16} & 8 \\ \frac{1}{8} & 16 \\ \frac{1}{4} & 32 \\ \frac{1}{2} & 64 \\ 1 & 0 \end{bmatrix} \tag{4}$$

Après toutes les itérations nous aurons que la matrice *ind* sera :

$$ind = \begin{bmatrix} 1 & 3 & 7 & 15 & 31 & 63 \\ 2 & 4 & 8 & 16 & 32 & 64 \\ 4 & 8 & 16 & 32 & 64 & 0 \\ 8 & 16 & 32 & 64 & 0 & 0 \\ 16 & 32 & 64 & 0 & 0 & 0 \\ 32 & 64 & 0 & 0 & 0 & 0 \\ 64 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5}$$

Exemple no.2

On veut calculer l'arbre de Huffman ternaire du vecteur de probabilités p :

$$p = \big[0.15, 0.20, 0.30, 0.35\big] \tag{6}$$

Le vecteur q initial sera donc :

$$q = \begin{bmatrix} 0 & 1 \\ 0.15 & 2 \\ 0.20 & 4 \\ 0.30 & 8 \\ 0.35 & 16 \end{bmatrix} \tag{7}$$

Après toutes les itérations nous aurons que la matrice *ind* sera :

$$ind = \begin{bmatrix} 1 & 8 \\ 2 & 7 \\ 4 & 16 \\ 8 & 0 \\ 16 & 0 \end{bmatrix} \tag{8}$$

Par la suite, il suffit de parcourir la matrice *ind* en partant du premier élément de la dernière colonne et d'attribuer les symboles à chaque probabilité faisant partie de l'indice non-nul rencontré. Lorsque l'on rencontre un indice puissance de 2 dans la matrice, cela signifie que le mot de code est complété et lors de rencontres subséquentes, on n'ajoute plus de symboles à cet indice. Par exemple, dans l'exemple ternaire précédent, lorsque l'on rencontre le nombre 7, la fonction sait que les probabilités d'indice {1,2,4} en font partie et ajoute alors un symbole à chacune des probabilités constituantes.

L'algorithme retourne donc les mots de code de chacune des probabilités (vecteur h) (incluant les probabilités fantômes) ainsi que la longueur moyenne du code via la variable l.

Exemple no.1

Les mots de code de l'arbre du premier exemple sont donc :

$$h = \begin{bmatrix} 000000 \\ 000001 \\ 00001 \\ 0001 \\ 001 \\ 01 \\ 1 \end{bmatrix} \tag{9}$$

Exemple no.2

Les mots de code de l'arbre du deuxième exemple sont donc :

$$h = \begin{bmatrix} 10 \\ 11 \\ 12 \\ 0 \\ 2 \end{bmatrix} \tag{10}$$