Compression de fichiers par l'algorithme de Huffman

Il existe plusieurs algorithmes permettant de compresser des documents. Les deux principaux sont :
-Huffman avec son prétraitement
-Lempel-Ziv-Welsh
Nous nous intéresserons à l'algorithme de Huffman, avec ses prétraitements :
-Permutation des caractères (Burrows-Wheeler)
-renommage (Move to Front)
Il faut tout d'abord calculer le code, coder le texte, puis le décoder.

1- Codage des caractères

Les caractères composant un texte sont en général codés sur un ou plusieurs octets. Par exemple, dans le jeu de caractères ASCII – le plus simple – chaque caractère est représenté en mémoire sur exactement un octet. En fait, dans ce code, seuls 128 caractères sont représentés et codés par les entiers de 0 à 127. Par exemple, le caractère '0' est représenté par l'entier 48, dont l'écriture en base 2, sur un octet, est 00110000, car 48 = 32+16 = 25 +24 . Ainsi, la suite de bits
0100101001000010001100000011000000110111
se décompose en 5 octets :
01001010 01000010 00110000 00110000 00110111
qui représentent la suite de 5 caractères 'JB007'.

Le principe de l'algorithme de Huffman est de représenter les caractères sur un nombre variable de bits, et non, par exemple, systématiquement sur 8 bits comme c'est le cas pour le codage ASCII. L'idée est simple : si on utilise un nombre variable de bits pour représenter les caractères, on peut utiliser des codes courts pour les caractères fréquents, et des codes plus longs pour les caractères rares.
Cet algorithme se décompose en plusieurs phases. Pour l'illustrer, considérons le texte

C'est un texte

Il y a 14 caractères codés sur 14x8 bits de mémoire.

1- Phase 1 : Fréquence de chaque caractère

La première phase consiste à effectuer lire le texte et à calculer la fréquence d'apparition de chaque caractère : (c,1) (',1) (e,3) (s,1) (t,3) (,2) (u,1) (n,1) (x,1)

2- Phase 2 : Tri

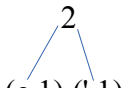
On trie ensuite le tableau obtenu. Sur l'exemple, on peut obtenir :
(c,1) (',1) (s,1) (u,1) (n,1) (x,1)
(,2)
(e,3) (t,3)

3- Création d'un arbre binaire

Un arbre binaire est un arbre dans lequel chaque nœud a soit deux soit aucun fils. En informatique, la racine d'un arbre est en haut. Les feuilles sont en bas de l'arbre et n'ont pas de fils. La racine est le seul nœud qui n'a pas de père. La racine est une feuille pour un arbre élémentaire à un élément. La probabilité de trouver un c ou un ' est de 2/14

(s,1) (u,1) (n,1) (x,1)

Noeud

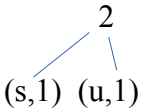


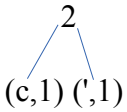
(,2)

Fils gauche Fils droit

(e,3) (t,3)

Puis la probabilité de trouver un s ou un u est aussi de 2/14
(n,1) (x,1)

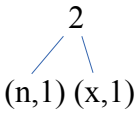


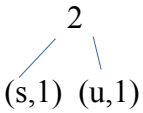


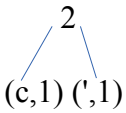
(,2)

(e,3) (t,3)

De même



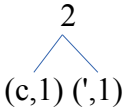




(,2)

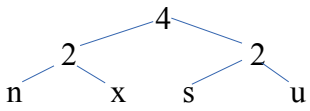
(e,3) (t,3)

Bâtissons maintenant un arbre de fréquence 4

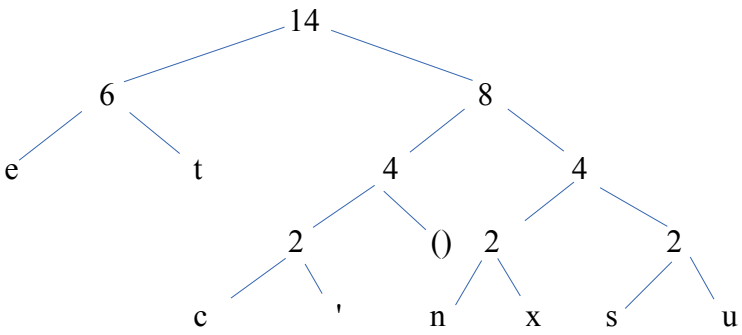


(,2)

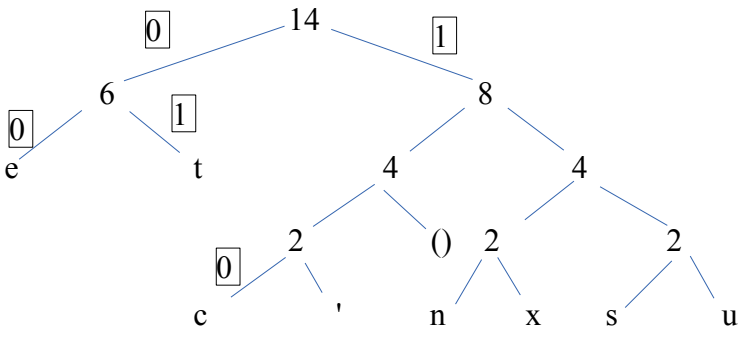
(e,3) (t,3)



et ainsi de suite pour obtenir l'arbre final :



Les caractères les plus fréquents sont le plus rapidement accessibles depuis la racine. Les caractères peu fréquents sont profonds dans l'arbre.L'arbre construit permet d'attribuer à chaque caractère un code. Pour cela, on parcourt l'unique chemin de la racine jusqu'au noeud étiqueté par le caractère dont on veut calculer le code. Le long du chemin,
— à chaque fois que l'on suit un fils gauche, on émet un 0,
— à chaque fois que l'on suit un fils droit, on émet un 1.
La plus longue branche fait 4, donc au maximum un caractère sera codé sur 4 bits.



codage de chaque lettre par cet arbre :
e : 00
t : 01
c:1000
':1001
(espace) : 101
n:1100
x:1101
s:1110
u:1111

Ecrire le code de la phrase « c'est un texte » avec cet arbre :

Pour décoder, il faudra lire les codes sur l'arbre. Il faut donc stocker l'arbre et le code.

2- Décodage

Contrairement au codage ASCII, il n'est pas aussi évident de redécouper le texte obtenu, puisque les codes sont maintenant de longueur variable. Comment savoir où se termine le premier caractère de notre message initial une fois codé ? Aucun mot de cet ensemble n'est préfixe d'un autre, et cela résulte immédiatement de ce que les mots de code sont associés aux feuilles d'un arbre. Donc, pour décoder, il suffit de lire la suite de 0 et de 1, en suivant les arêtes correspondantes dans l'arbre, jusqu'à ce qu'on arrive à une feuille.
Décoder le code précédent avec l'arbre.
Recommencer la procédure avec la phrase 'mon ananas '.

3- Complément

Le codage de Huffman est souvent utilisé avec 2 phases préliminaires, qui permettent d'obtenir des répétitions de caractères :
— La transformée de **Burrows-Wheeler**, qui permutte les caractères, de façon que l'on puisse décoder, en induisant des blocs de caractères identiques.
— La transformation **Move-to-front**, qui recode les caractères de façon également décodable, en transformant les blocs de caractères identiques en blocs de 0.L'ensemble de ces 2 transformation crée un texte de même taille que le texte de départ (en nombre de caractères), mais possédant de grands blocs de 0, ce qui augmente le taux de compression de l'algorithme de Huffman.

Burrows-Wheeler

On écrit toutes les permutations circulaires possibles du texte, puis on les tri par ordre alphabétique:

tete a tete	atetetete	1	
ete a tetet	eatetetet	2	
te a tetete	eteatetet	3	
e a tetetet	eteteatet	4	
a tetetete	eteteteat	5	
tetetetea	teatetete	6	
eteteteat	teteatete	7	texte original
teteteate	teteteate	8	
eteteatet	tetetetea	9	

puis
teteatete
Le texte sera codé avec la dernière colonne : (7, ettttteeea)

Move to Front

On code la chaîne précédente en utilisant un dictionnaire :
abcde.....t.....z
01234.....19...25

7, ettttteeea
e est codé par 4, puis on permute e au début du dictionnaire qui devient :
eabcd.....t.....z
01234.....19...25 e a le code 0
t est codé par 19 puis il est permuté au début du dictionnaire qui devient :
teabcd.....su.....z
012345.....1920...25 t a le code 0
Les t suivants auront donc le code 0 : 7, ettttteeea codé 7 4 19 000...
Le e suivant est codé 1, puis permuté et les e suivant sont codés 0. le dernier a est codé 2 donc 7, ettttteeea codé 7 4 19 0001002

Conclusion :

Donc pour compresser un texte, il faut :
- Burrows Wheelers
- Move to Front
- Huffman

puis pour décoder le texte :
- Huffman
- Move to Front
- Burrows Wheelers