Structure de données et algorithmes

Projet 2 : Structures de donnees

LEWIN Sacha

MAKEDONSKY Aliocha

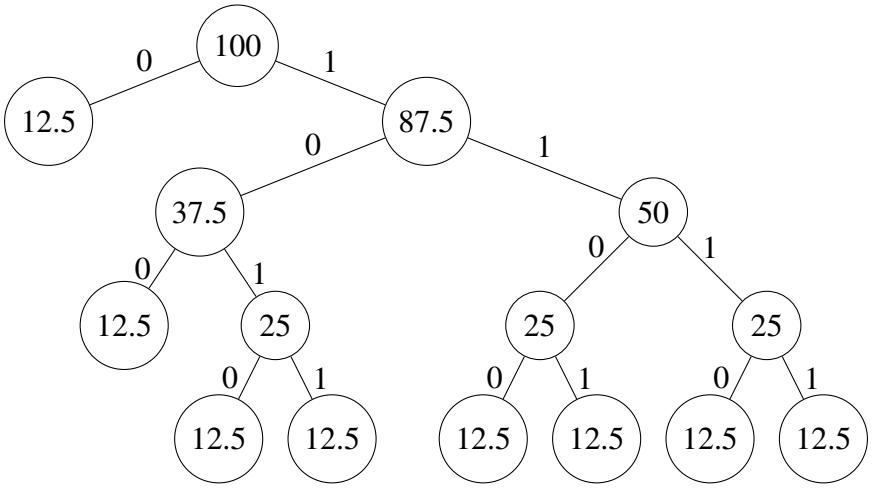
Ingénieur civil 2^{e} bachelier
Année académique 2019-2020

1 Structure des arbres sur base des distributions de fréquences

Dans ces arbres construits avec l'algorithme de *Huffman*, il y a toujours un nombre de feuilles équivalent au nombre d'éléments qu'on veut placer dans l'arbre (ce qui est logique puisque chaque feuille représente un élément et sa fréquence, ici donnée en %). Chaque parent contient une fréquence égale à la somme des fréquences contenues dans leur 2 fils. Par conséquent la racine de l'arbre contient la fréquence 100%. Parcourir l'arbre de la racine à une feuille nous donne le code binaire de l'élément contenu dans cette feuille (en regardant les chiffres inscrits sur les branches), le code s'écrivant de gauche à droite en descendant dans l'arbre.

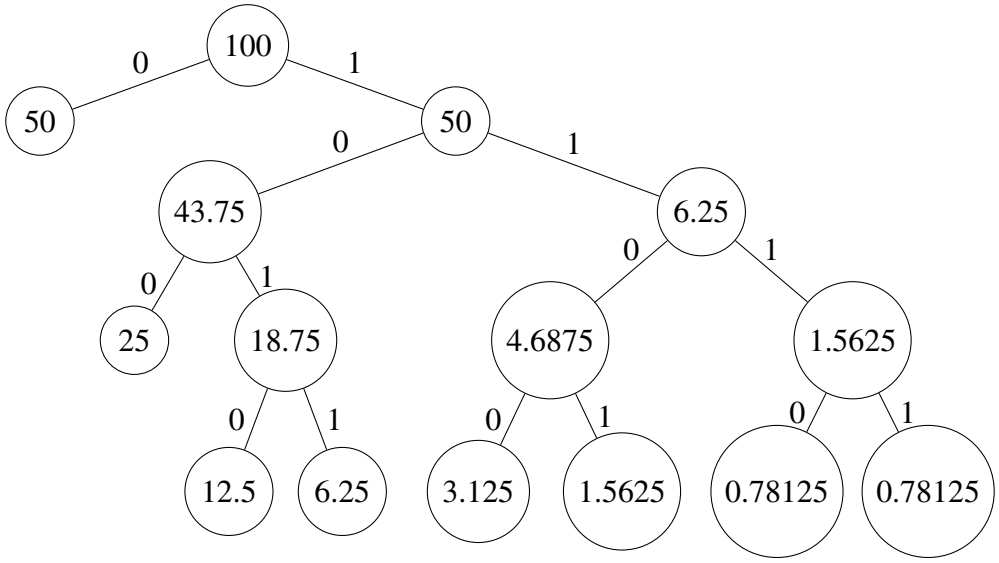
1.a Distribution équiprobable

Soit i éléments. La fréquence de chaque élément est donnée par $f_i = \frac{1}{k} \forall i \in \{1, ..., k\}$. Par conséquent tous les éléments ont exactement la même fréquence, par conséquent l'ordre dans lequel on les placerait dans une file n'a aucun importance puisqu'ils ont tous autant de chance d'apparaître, il n'y a donc pas de bon tri à effectuer sur ceux-ci. Les codes binaires attribués aux différents éléments sont donc "interchangeables" puisque ils ont tous la même fréquence donc attribué un code court à un ou à l'autre n'influencera en rien la mémoire totale prise par le codage d'un texte.



1.b Distribution "puissance de deux"

La relation de cette distribution est : $f_i = \frac{1}{2^i} \forall i \in \{1, ..., k-1\}$ et $f_k = f_{k-1}$. La dernière partie de cette relation est là pour pouvoir obtenir une fréquence finale de 100%, sinon on tendrait vers 100 mais on ne l'atteindrait jamais. L'écart entre chaque élément double à chaque fois, par conséquent celui-ci augmente très rapidement. Il doit donc y avoir un grand écart dans les distributions de chaque élément, certains doivent apparaître bien plus souvent que d'autres. Avec seulement 8 éléments il y a déjà un facteur 64 (2^{k-2}) entre la fréquence la plus grande et la plus petite.

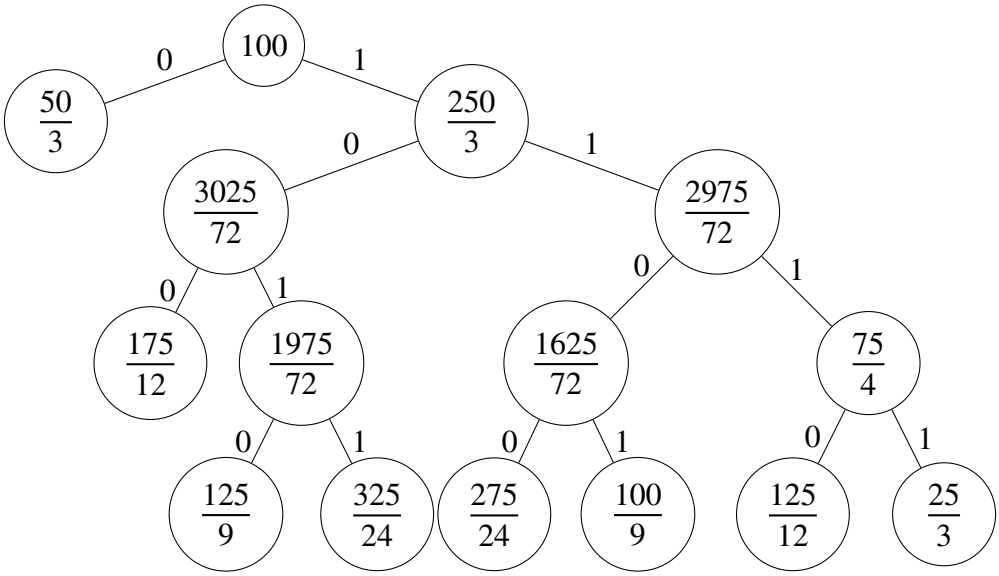


1.c Distribution "epsilon"

Ici les fréquences des éléments sont plus proches les unes des autres que dans le cas de la distribution en puissance de deux, mais quand même pas au point de la distribution équiprobable. Le rapport entre la fréquence de l'élément le plus présent et celle du moins présent vaut

$$\frac{\frac{1}{k}+\frac{1}{3k}}{\frac{1}{k}-\frac{1}{3k}} = \frac{\frac{4}{3k}}{\frac{2}{3k}} = 2$$

. Ce qui est intéressant ici est que ce rapport est, comme nous pouvons le voir, constant. Par conséquent, nos fréquences f_i



2 Complexité de l'algorithme de Huffman

2.a Cas de la file à priorité avec liste

2.b Cas de la file à priorité avec tas

3 Algorithme de décodage

3.a Pseudo-Code de DECODE

```
1:  function DECODE(B, T)
2:      currentBit = 1
3:      while currentBit < B.nbBits do
4:          k = 0
5:          for j = p to r do
6:              if A[i] > A[j] then
7:                  k = k + 1
8:              end if
9:          end for
10:         if k = i − p or (i ≠ p and A[i] = A[i − 1]) then
11:             Break
12:         end if
13:     end while
14: end function
```

3.b Complexité de DECODE avec encodage de Huffman

3.c Complexité de DECODE avec encodage à largeur fixe

3.d Différence de taille des fichiers

Nous pouvons remarquer une diminution d'environ 40% entre la taille du fichier encodé à largeur fixe, et celui encodé avec l'encodage de Huffman.

En effet, pour différents fichiers testés, voici les données qu'on en tire et la diminution entre l'encodage à largeur fixe et l'encodage de Huffman pour chacun des livres testés :

Livre	Taille (largeur fixe, en kB)	Taille (Huffman, en kB)	Diminution (%)
Frankenstein	443	247	44.24
Pride and Prejudice	785	423	46.11s
Alice in Wonderland	164	96	41.46
The Importance of Being Earnest	139	82	41