Travaux pratiques

En vous aidant de la page Wikipedia, vous allez implémenter le codage de Huffman qui est un algorithme de compression de données sans perte. Il sert à encoder un texte en binaire, en utilisant pour chaque lettre un nombre de bits dépendant du nombre de fois où la lettre est présente : plus la lettre apparaît, plus le nombre de bits est petit. Le codage ASCII standard utilisant huit bits pour coder chaque lettre, Huffman réduit donc le nombre de bits utilisés pour stocker du texte. Le principe est de construire un arbre binaire dont les feuilles correspondent aux caractères à encoder. Le chemin emprunté pour atteindre une feuille depuis la racine définit le code de la lettre sur cette feuille : à gauche 0, à droite 1. L'algorithme de Huffman commence par compter dans le texte le nombre d'apparitions de chaque caractère, puis il initialise un arbre par caractère, pondéré par le nombre d'apparition de ce caractère. À chaque itération, l'algorithme sélectionne les deux arbres ayant les poids les plus faibles, et les assemble pour former les deux enfants d'un arbre binaire dont le poids est la somme des deux arbres.

début
    créer une file
    pour chaque *caractère c de l'alphabet* **faire**
        créer un arbre a
        $a.caractere \leftarrow c$
        $a.poids \leftarrow$ frequence de c
        insérer a dans la file
    fin
    tant que *la file contient plus d'un arbre* **faire**
        créer un arbre a
        $a.gauche \leftarrow$ l'arbre de poids le plus faible de la file.
        Retirer cet arbre de la file.
        $a.droite \leftarrow$ l'arbre de poids le plus faible de la file.
        Retirer cet arbre de la file.
        $a.poids \leftarrow a.gauche.poids + a.droite.poids$
        insérer a dans la file.
    fin
    retourner le seul arbre contenu dans la file.
fin

Algorithme 1 : Construction de l'arbre de Huffman

Pour représenter un arbre de Huffman, nous aurons besoin d'une classe Noeud qui comportera :

- son enfant gauche ;
- son enfant droit ;
- un poids ;
- un caractère ;

Une fois l'arbre de Huffman généré, nous pourrons le parcourir afin de générer une table qui nous permettra d'associer à chaque caractère du dictionnaire son code de Huffman. Ce code pourra être stocké dans un vecteur de booléens.
Il ne reste alors plus qu'à écrire les méthodes qui serviront à coder et décoder des chaînes de caractères.