Département Informatique

Compression en codage de Huffman

Sujet (inspiré d'un sujet proposé à l'Université de Bordeaux) :

Une idée, apparue très tôt en informatique, pour compresser les données repose sur la remarque suivante : les caractères d'un texte sont habituellement codés sur un octet, donc le même nombre de bits pour tous. Il pourrait être plus économique en terme d'espace disque, pour un fichier donné, de coder ses caractères sur un nombre variable de bits, en utilisant peu de bits pour les caractères fréquents et plus de bits pour les caractères rares. Ce codage dépend donc du fichier à compresser. Les propriétés d'un tel codage doivent être les suivantes :

a) Chaque caractère est codé sur un nombre différent de bits (pas nécessairement un multiple de 8) ;
b) Les codes des caractères fréquents dans le fichier sont courts, ceux des caractères rares sont plus longs ;
c) Bien que les codes soient de longueur variable, on peut décoder le fichier compressé de façon unique.

Cette dernière propriété est automatiquement assurée si le code est tel que :

d) Si c1 et c2 codent deux caractères différents, c1 ne commence pas par c2 et c2 ne commence pas par c1.

En effet, si cette dernière propriété est vérifiée, lorsqu'on décode le fichier compressé en le lisant linéairement, dès que l'on reconnaît le code d'un caractère, on sait que l'on ne pourra pas le compléter en un autre code.

L'algorithme de Huffman, qui garantit ces propriétés, fonctionne de la façon suivante :
- On calcule d'abord les fréquences d'apparition de chaque caractère dans le fichier à compresser ;
- On calcule ensuite pour chaque caractère un code satisfaisant les propriétés a), b) et d) ;
- On écrit ce code au début du fichier compressé (pour que le décompresseur y ait accès), suivi des données compressées elles-mêmes.

Pour calculer le code de chaque caractère, l'algorithme construit un arbre binaire par itérations :
i) Les feuilles de l'arbre sont les caractères apparaissant dans le fichier ;
ii) Deux nœuds n1 et n2 de fréquences minimales sont choisis. On construit un nouveau nœud n qui devient :
iii) père de n1 et n2, et dont la fréquence est la somme de celles de n1 et n2 ;
iv) On répète l'étape précédente jusqu'à atteindre une unique racine.

Vous trouverez de nombreuses descriptions de l'algorithme de Hoffman sur le web. Je vous suggère de consulter le document suivant :

http://www-igm.univ-mlv.fr/~lecroq/cours/huff.pdf

Il illustre de façon claire le principe de construction de cet arbre de codage. Vous y trouverez également des pseudocodes utiles pour la réalisation de ce projet.

L'arbre de codage permet le calcul du code de chaque caractère. Il permet également de décoder le fichier compressé. Il doit donc être connu du programme d'extraction. Pour cela le programme de compression doit écrire l'arbre de codage au début du fichier compressé, sous un format à définir. Le fichier compressé sera donc constitué de deux parties :

- une première partie contenant l'arbre de codage pour permettre au décompresseur de retrouver le caractère associé à chaque code ;
- une seconde partie contenant la suite des codes des caractères du fichier à compresser.

Attention, le décompresseur doit toujours pouvoir trouver la séparation entre ces deux parties.

Un dernier problème technique est que le nombre de bits occupés par le codage du fichier n'est plus un multiple de 8. Il est donc impossible de détecter la fin des données. Une solution consiste à calculer l'arbre de codage comme s'il existait un caractère supplémentaire (appelé pseudo-caractère) marquant la fin du fichier original. Ce pseudo-caractère aura donc comme fréquence 1. Lorsque la fin du fichier original est atteinte, le compresseur écrit le code de ce pseudo-caractère, et complète par des bits 0 pour avoir une taille totale du fichier multiple de 8 bits. Lorsque le décompresseur rencontre le code de ce caractère, il sait que la fin du fichier compressé est atteinte.

Travail demandé

Ce projet doit être réalisé en groupes de 2 à 4 élèves.

Gestion des fichiers :

Si les fichiers avant compression et après décompression sont des fichiers simples contenant du texte, les fichiers compressés sont des fichiers binaires auxquels on doit pouvoir accéder, en lecture comme en écriture, bit par bit. Vous développerez donc un module permettant la gestion de ce type de fichiers. Votre module comportera :

- Une structure permettant de gérer les échanges avec le disque, mémorisant notamment l'état des échanges en cours ;
- Une fonction permettant d'ouvrir un fichier binaire en lecture ou écriture ;
- Une fonction permettant d'écrire un bit dans un fichier binaire ;
- Une fonction permettant de lire un fichier binaire ;
- Une fonction permettant de fermer un fichier.

Arbre de codage :

Après avoir écrit une fonction déterminant la fréquence d'apparition des caractères dans un fichier, vous implémenterez l'algorithme de construction de l'arbre de codage. Vous utiliserez le module `arbres_binaires` que vous avez développé dans le cours d'Algorithmique.

Compression et extraction :

Vous écrirez ensuite la fonction de compression d'un fichier texte dans un fichier binaire puis la fonction d'extraction d'un fichier texte depuis un fichier compressé.

Evaluation de votre travail

Vous disposez de 5 séances de 2 heures, réparties sur 3 semaines avant les vacances, pour réaliser ce projet. La présence lors de ces séances est obligatoire et sera prise en compte dans la note attribuée à chaque élève : un élève qui n'aura jamais été présent aura donc 0, même si son groupe réalise un projet parfait.

La notation prendra en compte les méthodes de travail du groupe (démarche, documentation, apprentissage, gestion des difficultés rencontrées, tests, etc.) que j'observerai au cours des 5 séances.

Il existe de nombreux programmes mettant en œuvre l'algorithme d'Huffman. Vous pouvez évidemment vous en inspirer, mais en aucun cas les copier. Votre code sera organisé comme demandé dans la section précédente. Tout élève devra être en mesure de m'expliquer l'architecture développée par son groupe et le principe de fonctionnement de chaque fonction.

Vous prendrez le temps de documenter votre code au fur et à mesure de son développement, et ce dès le début du projet.

Lors de la dernière séance chaque groupe devra être en mesure de compresser un fichier que je fournirai et de décompresser le fichier ainsi obtenu. La convivialité d'utilisation du programme sera prise en compte dans l'élaboration de la note.