Compression des données

Exercices

Titre
Objectif(s) :
A la fin des exercices, l'élève doit être capable de :

- Comprendre ce qu'est la compression
  - Compression par dictionnaire
- Comprendre la compression RLE
- Comprendre le codage Huffman
- Comprendre la compression par dictionnaire LZ78
- Calculer le taux de compression
- Compression avec un outils comme 7Zip

Durée prévue : Selon UD

1 Compression par dictionnaire (exemple inventé)

Nous avons le texte ci-dessous à compresser qui à 59 caractères au total (1 caractères = 1 octet pour l'exemple) :

« un petit problème ou un tout petit problème est un problème »

Pour réaliser cet exemple de compression, nous allons créer un dictionnaire contenant chaque mot différent contenu dans cette phrase. Nous attribuerons un chiffre à chacun de ces mots, y compris l'espace. De cette manière nous arrivons à réduire le nombre d'octets nécessaire pour enregistrer le texte.

1) Établir un tableau :

- 1ère colonne, le numéro de correspondance au mot
- 2e colonne, chaque mot différent du texte
- 3e colonne comptabiliser le nombre de symbole contenus dans les deux premières colonnes

Additionner la 3ème colonne et mettre le total dans l'encadré

DICTIONNAIRE		
Numéro	***Mot unique***	***Total caractères***
1	*un*	*3 (1un)*
2	␣	*2*
3	petit	6
4	problème	9
5	ou	3
6	tout	5
7	est	4
	Nombre total de symboles	32

2) Réécrire la phrase mais avec les numéros correspondants de la première colonne du tableau

La phrase devient donc (1 chiffre par case) :

1	2	3	2	4	2	5	2	1	2	6	2	3	2	4	2	7	2	1	2	4			

Quel est le nombre de caractères de la phrase codée ? 21 caractères

3) Additionner le nombre de caractères figurant dans la phrase codée avec le total des caractères du dictionnaire :

Nb total de caractères du dictionnaire	32
Nb caractères figurant dans la phrase codée	21
Total	53

4) Quel taux de compression obtenez-vous ?

T = 1 - (53 / 59) = 10.16%

2 Compression RLE (non destructive)

1) Décrire le principe RLE :

C'est un codage simple basé sur la répétition d'éléments consécutifs permettant

une compression de données.

2) Sachant qu'une lettre est codée sur 1 octet, combien d'octets sont nécessaires pour coder sans compression WAAAAAOUUUUUUUH ?

15 octets

3) Coder avec cet algorithme la même chaîne de caractères (WAAAAAOUUUUUUUH)

1W5A1O7U1H

4) Sachant que, dans le code compressé, une lettre ou un chiffre seront aussi codés avec 1 octet, combien d'octets faut-il ?

10 octets

5) Quel est le taux de compression sur cet exemple ?

T = 1 – (10 / 15) = 33.33%

6) Recommencer avec la chaîne de caractères suivante : SUPER

1S1U1P1E1R

7) Taux de compression :

SUPER = 5 octets et après compression = 10 octets.

T = 1 – (10 / 5) = -100%. Aucune compression ! On a même doublé l

8) Conclusion :
Dans quel cas la compression par RLE est-elle intéressante ?

Dans le cas où nous avons plusieurs fois les mêmes lettres. Ce sera la même chose pour des pixels ou autres données avec forte redondance.

3 Codage Huffman (non-destructive)

L'algorithme de compression de Huffman est un système de codage statistique. Pour Huffman, il faut compter la fréquence de chaque caractère dans un texte ou une chaîne.

Voici le message à compresser "**ta tata et ton tonton**".

1) Pourquoi est-il faux de dire que ce message contient 17 caractères ?

Car il faut compter les espaces.

2) Combien en compte-t-il réellement ?

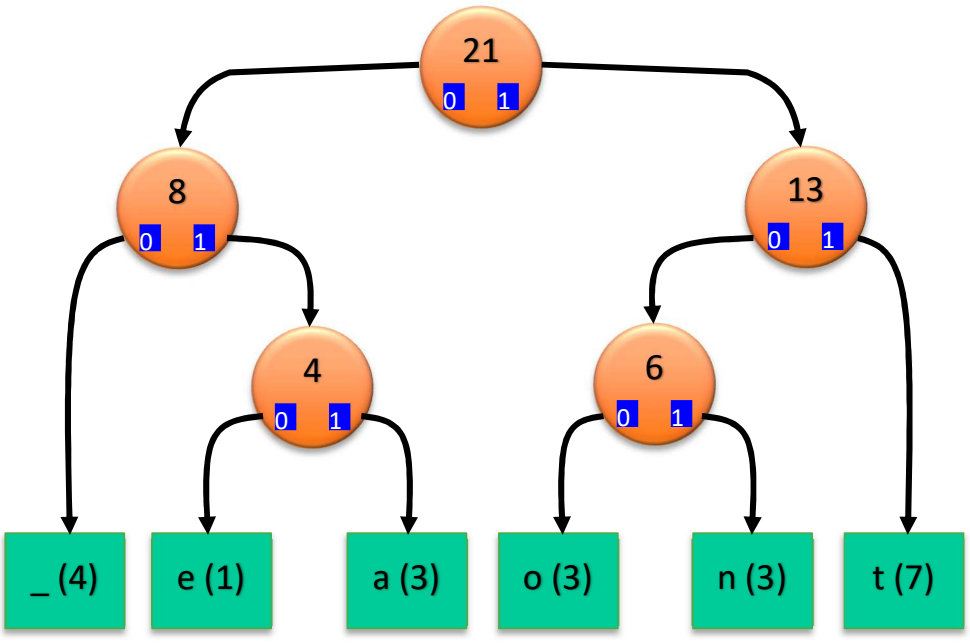
21

3) La méthode de Huffman se décompose en 3 étapes :

1ère étape : Compter les occurrences des différents caractères (_ pour espace) :

Lettres	t	_	a	o	n	e
Occurrences	7	4	3	3	3	1

2e étape : Construire l'arbre de codage.



On cherche à avoir des branches équilibrées.

Le code des caractères devient :

Lettres	t	-	a	o	n	e
Codes	11	00	011	100	101	010

3e étape :

Le message « **ta tata et ton tonton** » devient avec le code précédent (espacer les codes binaires) :

11 011 00 11 011 11 011 00 010 11 00 11 100 101 00 11 100 101 11 100 101

4) Calcul du taux de compression pour cet exemple :

Nombre de bits du message initial (8 bits par caractères) :

(21 * 8) = 168

Nombre de bits du message final :

52

Taux de compression :

T = 1 – (52 / 168) = 69%

4 Compression par dictionnaire LZ78

Décoder le message suivant :

(0, v) (0, e) (0, r) (0, i) (0, d) (4, q) (0, u) (2, _) (7, n) (0, _) (0, p) (6, u) (8, n) (12, e)

Reconstruire le dictionnaire au fur et à mesure :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
v	e	r	i	d	iq	u	e_	un	_	p	iqu	e_n	ique				

Message décodé :

veridique un pique nique……

Le dico contient …14… éléments mais on compte jusqu'à …12…max. pour indiquer la position dans le dictionnaire, ce nombre se code sur …4… bits.

La phrase initiale faisait …24… caractères soit …24 x 8= 192… bits.

La phrase codée fait …14x (4 + 8) = 168… bits.

Soit un taux de compression de …1 - (168 / 192) = 12.5%.....

5 Compression des fichiers archives

5.1 Le format .docx

Avec Word, créez un nouveau document et le nommer « **document.docx** ». Ecrire « **Hello** », puis le fermer. Changer l'extension du document en « **zip** », et le dézipper avec 7-zip. Rechercher dans les fichiers le texte « **Hello** » puis le renommer avec le texte « **Salut** ». Récrire l'archive avec 7-zip, et changer l'extension en « **docx** ». L'ouvrir avec Word.

Que constatez-vous ?

On constate que le format Word n'est rien d'autre qu'une archive contenant plusieurs fichiers et dossiers. En modifiant le fichier document.xml, on modifie le document Word.

5.2 Les dossiers compressés

Récupérez le fichier « sauvegarde.exe » et exécutez-le afin d'extraire dans un dossier de travail les fichiers nécessaires. Le dossier « sauvegarde » sera créé automatiquement.

Créer une archive « **zip** » depuis l'explorateur de fichier (il gère les zip) du dossier « **sauvegarde** ». Si vous ne savez pas comment faire, un clic droit sur le dossier devrait vous aider.

Pourquoi est-ce important de faire cela si l'on veut pérenniser les données de ce dossier ?

Simplement cela permet de gagner de la place sur notre support de sauvegarde et ainsi permettre de sauvegarder plus de données.

5.3 7-Zip

Récupérez « paysage.exe » procédez à son auto extraction dans votre dossier de travail.

Cette fois, **avec l'application 7-Zip** faites les manipulations ci-dessous.

Faites une archive de chaque fichier séparément (bmp.7z et png.7z) selon leur extension avec les paramètres suivants :

- Format de l'archive : 7z
- Niveau de compression : 9 - Ultra

Pourquoi « png.7z » a quasi le même poids que « paysage.png » à quelques octets près ?

Car le PNG est déjà compressé donc inutile de recompresser. Le gain serait faible.

Que s'est-il passé pour le fichier « paysage.bmp » ?

Il a été compressé avec un taux d'environ 40%.

Suite de l'exercice sur la page suivante…

⚠ Supprimez les deux archives 7z avant de procéder à la suite !

Maintenant faite une archive du dossier « paysage » avec les 2 images uniquement en utilisant les paramètres suivants :

- Format de l'archive : 7z
- Niveau de compression : 5 - Normale
- Chiffrement, mot de passe : toto123
- Diviser en volumes, octets : 1M

Il y a maintenant 4 fichiers qui composent cette archive, pourquoi et à quoi cela peut-il servir ?

On vient de créer une archive multi volumes de 1 Mo max. chacun. Cela peut être utile si on doit stocker une archive sur un support qui limite la taille des fichiers.

Pourquoi le mot de passe ?

Les données sont chiffrées. On ne peut pas les extraire de l'archive sans le mot de passe.

6 Calcul du poids d'un audio

Démontrez par calcul pourquoi le fichier « musique.wav » fait 3.36 Mo :

- 16 bits
- 44 100 Hz
- Stéréo (2 voies)
- Durée 20"

20 secondes x 44100 Hz x 2 canaux x 2 octets (16 bits) = 3 528 000 octets = 3.36 Mo

Un entête de fichier WAV ne prend que 44 octets de place.