

Chapitre 8

Compression

05/12/03

Compression

1

Entropie

05/12/03

Compression

5

Référence 1

- Auteur : Xavier Marsault
- Titre : Compression et cryptage des données multimédias
- Edition : Hermes, 2^{ème} édition, 243 pages, 1995
- Prix : 200 F

05/12/03

Compression

2

Principe

- Détection et correction d'erreurs
 - augmenter la redondance
- Compression
 - diminuer la redondance

05/12/03

Compression

6

Référence 2

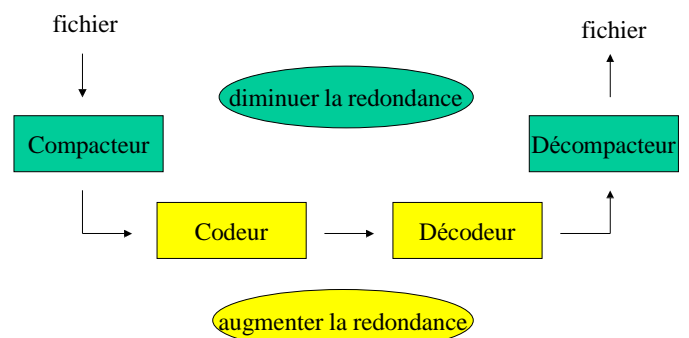
- Auteur : David Salomon
- Titre : Data compression
- Edition : Springer, 428 pages, 1998
- Prix : 375 F

05/12/03

Compression

3

Transmission de l'information



05/12/03

Compression

7

Plan

- Aspect théorique
 - Entropie
- Méthodes de compression
 - Huffmann
 - LZW

05/12/03

Compression

4

Théorie de l'information

- L'information présente dans un message est-elle quantifiable ?
- Idée : le niveau de surprise qu'apporte le message
 - Il y a cours de réseaux dans 15 jours
 - Il n'y a pas cours de réseaux dans 15 jours
 - Il n'y aura plus de cours de réseaux

05/12/03

Compression

8

Aspect quantitatif de l'information

Une information désigne un événement parmi un ensemble d'événements possibles.

Une information I est d'autant plus intéressante que l'événement (de probabilité p) qu'elle code est peu probable.

$$Q(I) = \log_2(1/p) \text{ bits (ou logon)}$$

05/12/03

Compression

9

05/12/03

Entropie

Soit un alphabet A constitué de n symboles,

Soit un message M construit à partir de A,

Soit p_i la fréquence du $i^{\text{ème}}$ symbole de A dans M

$$E(M) = p_1 \log_2(1/p_1) + \dots + p_n \log_2(1/p_n)$$

L'entropie représente la quantité d'information moyenne apportée par un symbole dans M.

Compression

13

Exemple 1

Le lancer d'une pièce de monnaie

- 1 question élémentaire est nécessaire pour cerner le résultat
- Quantité d'information présente dans le message M décrivant le résultat du lancer :
 - $Q(M) = 1$

05/12/03

Compression

10

05/12/03

Compression

14

Redondance entropique

Si tous les symboles de A sont codés sur $\log_2 n$ bits alors :

$$R(M) = \log_2 n - E(M)$$

La redondance entropique représente la quantité d'information moyenne redondante par symbole.

Exemple 1-bis

Le lancer d'une pièce de monnaie truquée

- pile : 1 chance sur 4
- face : 3 chances sur 4

- Quantité d'information présente dans le message M décrivant pile comme résultat du lancer :
 - $Q(M) = 2$

05/12/03

Compression

11

05/12/03

Exemple 1/2

Soit l'alphabet $A = \{a_1, a_2, a_3, a_4\}$

Soit le message M tel que :

$$p_1 = p_2 = p_3 = p_4 = 1/4$$

On a :

$$E(M) = 2$$

$$R(M) = 0$$

Compression

15

Exemple 2

Tirer au hasard une valeur entre 1 et 1024

- 10 questions élémentaires sont nécessaires pour cerner le résultat
- Quantité d'information présente dans le message M décrivant le résultat du tirage :
 - $Q(M) = 10$

05/12/03

Compression

12

05/12/03

Exemple 2/2

Soit l'alphabet $A = \{a_1, a_2, a_3, a_4\}$

Soit le message M tel que

- $p_1 = 0.49, p_2 = 0.01$
- $p_3 = p_4 = 1/4$

On a :

$$E(M) = 1.57$$

$$R(M) = 0.43$$

Compression

16

Méthodes de compression

- Méthodes de type statistique
 - Algorithme de Huffman
 - ...
- Méthodes de type dictionnaire
 - Algorithme LZW
 - ...

05/12/03

Compression

17

Exemple

- Soit le message suivant à coder :
les poissons sont rouges
- Les fréquences des différents symboles sont:
 $f(l) = f(p) = f(i) = f(t) = f(r) = f(u) = f(g) = 1/24$
 $f(e) = f(n) = 1/12$
 $f() = 1/8$
 $f(o) = 1/6$
 $f(s) = 1/4$

05/12/03

Compression

21

Méthodes de compression

Code de Huffman

- Sur l'arbre de Huffman, numéroter chaque couple d'arcs de même destination avec 0 et 1 respectivement.
- Effectuer sur l'arbre de Huffman une lecture du code de chaque symbole en considérant le chemin menant de la racine au nœud adéquat.

05/12/03

Compression

18

05/12/03

Compression

22

Algorithme de Huffman

Principe

Attribuer aux différents symboles constituant un fichier des codes binaires de taille variable selon leurs fréquences.

- Symbole fréquent \Rightarrow code court
- Symbole rare \Rightarrow code long

Exemple

On peut obtenir sur notre exemple :

| | |
|-----------|-----------|
| s : 00 | o : 01 |
| : 100 | n : 1100 |
| e : 1010 | g : 1011 |
| u : 11010 | r : 11011 |
| t : 11100 | i : 11101 |
| p : 11110 | l : 11111 |

05/12/03

Compression

19

05/12/03

Compression

23

Arbre de Huffman

- Initialement
 - Créer un nœud par symbole
 - Chaque nœud est étiqueté par la fréquence d'apparition du symbole associé.
- Itérativement
 - Sélectionner deux nœuds x et y (sans arc sortant) étiquetés par les plus faibles valeurs.
 - Construire un nouveau nœud z étiqueté par la somme des étiquettes de x et y, ainsi que deux arcs sortants reliant respectivement x et y à z.

Exercice

- Si on utilise un codage de taille fixe, quelle est la redondance entropique du message ?
- Si on utilise le codage de Huffman (codage de taille variable), quelle est la redondance entropique du message ?

05/12/03

Compression

20

05/12/03

Compression

24

Algorithme LZW

- LZW = Lempel, Ziv [1978] et Welch [1984]
- Méthode de type dictionnaire
 - n'utilise pas de modèle statistique
 - n'utilise pas de codage de taille variable
 - utilise un dictionnaire dynamique

05/12/03

Compression

25

Exemple

- Soit le message suivant à coder :
un pour tous pour un
- Les nouvelles entrées du dictionnaire sont :
'un' 'n ' ' p'
'po' 'ou' 'ur'
'r ' ' t' 'to'
'ous' 's ' ' to' ...

05/12/03

Compression

29

Dictionnaire

- Contient des séquences d'octets répétitives du fichier traité
- Est construit pendant la compression sans lecture préalable (compactage au vol)
- N'a pas à être sauvé puis transmis
- Est reconstruit automatiquement à la décompression

05/12/03

Compression

26

Décompression

```
a ← f.lireOctet()
precedentMot ← mot(a,D)
Emettre precedentMot
Tant que (non f.eof()) répéter
  a ← f.lireOctet()
  mot ← mot(a,D)
  Emettre mot
  Si precedentMot + mot(1) ∈ D alors
    ajouter precedentMot + mot(1) dans D
  Fin si
  precedentMot ← mot
Fin tant que
```

05/12/03

Compression

30

Notations

- f : le fichier à compresser
- D : le dictionnaire qui contient initialement 256 entrées correspondant aux 256 valeurs possibles d'un octet
- adresse(m,D) : l'adresse du mot m dans D
- mot(a,D) : le mot dans D d'adresse a
- m(j) : la j^{ème} lettre (octet) du mot m

05/12/03

Compression

27

Compression

```
c ← f.lireOctet()    mot ← c
Tant que (non f.eof()) répéter
  c ← f.lireOctet()
  Si mot+c ∈ D alors
    mot ← mot + c
  Sinon
    Émettre adresse(mot,D)
    Ajouter mot+c dans D
    mot ← c
  Fin si
Fin tant que
Emettre adresse(mot,D)
```

05/12/03

Compression

28