Représentation et codage de l’information

8-Fichiers archives, Théorie de l’information, Compression avec pertes

L1 Informatique, Université d’Orléans

Florent Foucaud, 2019

Fichiers archives

Fichiers archives

Idée : encoder plusieurs fichiers/dossier dans un seul

Optionnellement :

- Compression
- Cryptage

Quelques formats d'archives

- AR, "the archiver" (Bell Labs, 1971)
- TAR, "tape archive" (Bell Labs, 1979)
- SHAR, "shell archive" (James Gosling, ~1980)

  fichier exécutable en commande shell

J. Gosling

- ZIP (PKZIP, Philip Katz, 1989)

  DEFLATE : LZ77 + Huffmann

P. Katz

- GZIP "GNU zip" (Mark Adler+Jean-Loup Gailly, 1992)

  DEFLATE : LZ77 + Huffmann

gzip

Mark Adler

J.-L. Gailly

- RAR "Roshal archive" (Yevgenii Roshal, 1993)

  algorithme de type Lempel-Ziv

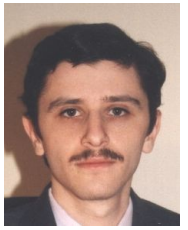
J. Seward

- BZIP2 (Julian Seward, 1996) bzip2

  utilise Burrows-Wheeler

- 7-zip (Igor Pavov, 1999) 7ZIP

  variante LZMA de Lempel-Ziv + Burrow-Wheeler



Y. Roshal

Un peu de théorie de l’information

Entropie de Shannon (1948)

En thermodynamique, l'entropie est une grandeur physique qui mesure le "désordre" d'un système physique.

(2ème loi de la thermodynamique : l'entropie d'un système fermé ne diminue jamais)

Informellement :

- le désordre contient plus d'information que l'ordre

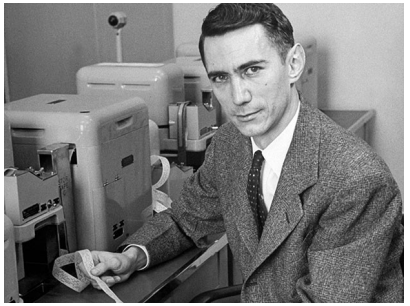
- un événement peu probable nous donne beaucoup d'information

Entropie de Shannon d'une chaîne S :

nombre minimum de bits nécessaires pour encoder S

$$H(S) = \sum_{s \in S} p(s) * \log_2 \frac{1}{p(s)}$$

p(s) : probabilité d'appartition du caractère s



Claude Shannon

H(S) = 0 si p(s)=1 pour tout s

H(S) élevée si beaucoup de valeurs peu probables

Remarque : codes de Huffman optimaux par rapport à l'entropie !

Peut-on tout compresser ?

Prenons l'ensemble des messages à n bits. Il y en a : 2^n

Combien de codes compressés différents existent ?

$2^0 + 2^1 + 2^2 + \ldots + 2^{n-1} = 2^n - 1 < 2^n$

Par le "principe des tiroirs" (pigeonhole principle) :

Au moins un des messages ne peut pas être compressé !

De plus, la moitié des messages n'ont gagné qu'un seul bit...

Complexité de Kolmogorov (~1960)

Définition : Soit une chaîne s.
K(s) est la longueur d'un plus petit programme (par ex. en python) qui affiche s.

Andrey Kolmogorov

Exemples :

- s = abababababababababababababababababababab

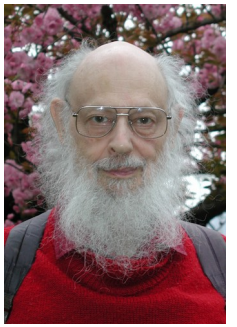
```
for i in range(20)          K(s) <= 29
    print("ab")
```

Gregory Chaitin

- s = qbgthtnsbkjntndbjnjnr

```
print("qbgthtnsbkjntndbjnjnr")      K(s) <= |s| + 8
```

- Calcul de la suite des décimales de Pi :
  algorithme de Gauss-Legendre



Ray Solomonoff

Complexité de Kolmogorov (~1960)

Définition : Soit une chaîne s.
K(s) est la longueur d'un plus petit programme qui affiche s.

Théorème : il n'existe aucun programme pour calculer K(s)

Preuve par l'absurde : Soit A un tel programme de longueur |A|. Soit le programme suivant, qui renvoie la première chaîne s telle que K(s) >= |A| + 1000

```
Pour tout i de 1 à l'infini :
    Pour toute chaîne s de longueur i :
        si K(s) >= |A|+1000
            renvoyer s
```

Mais on obtient un programme de taille inférieure à |A|+1000 qui calcule s !
Donc, K(s) < K(s) – une contradiction ! – ***CQFD***

- *Paradoxe de Berry (Russell, 1906) :*
  "Le plus petit entier qu'on ne peut pas définir avec moins de 80 symboles"
- *Paradoxe du barbier :*
  "Le barbier rase tous les habitants qui ne se rasent pas eux-mêmes"

Compression avec pertes

Principe général

Fichiers image/vidéo/audio

Idée : altérer la qualité de façon peu perceptible pour nos sens :
on réduit le niveau de détail

- Images : réduire le nombre de couleurs
  réduire le nombre de détails

- Sons : enlever des fréquences peu audibles
  réduire le nombre de fréquences différentes

Exemple : compression JPEG

JPEG : **Joint Photographic Experts Group** (1992)

Permet différents taux de compression :

39 kilo octets

16 kilo octets

9 kilo octets

Lena Söderberg, 1972

Compression par transformée

Méthode utlisée pour compresser :
images JPEG, vidéos MPEG, audios MP3...

1. Diviser les données en “blocs” de taille égale, N

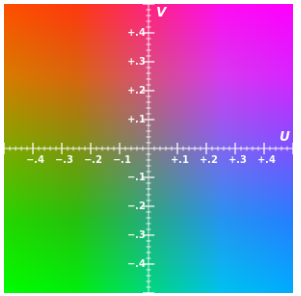
2. Transformer chaque bloc via une *fonction de transformée* (transformée de cosinus, transformée de fourier, etc)

   Cela revient à exprimer la donnée par une somme de fonctions sinusoïsales (cosinus, sinus)

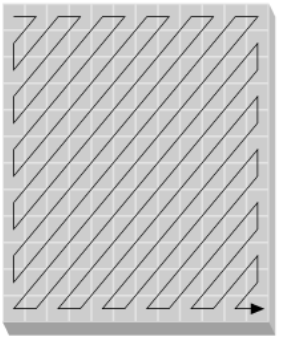
3. “Quantifier” le résultat, c’est-à-dire, l’approximer
   C’est ici que s’effectue la compression avec pertes !

4. Compresser encore davantage le résultat via une méthode de compression sans pertes (RLE, Huffman, etc)

JPEG : quelques détails



- Encodage en espace de couleurs YUV
- Découpage de l’image en blocs 8x8
- **Transformée en cosinus discrète**
- **Quantification**
- Encodage en zigzag
- RLE + Huffman

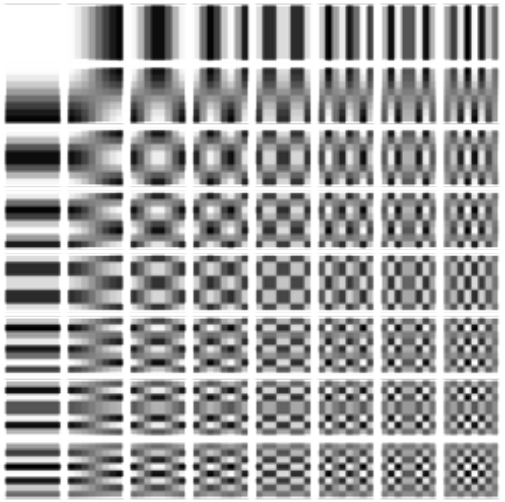


JPEG : quelques détails

Pour chaque bloc 8x8 :

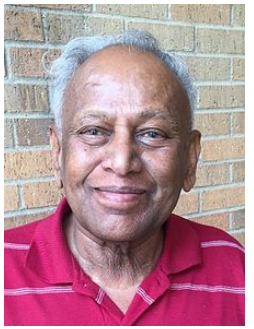
- On considère les 64 valeurs comme une fonction
- **On applique à cette fonction la transformée en cosinus discrète** (DCT, 1974) :

> *Toute fonction discrète f(x) sur n valeurs de x peut être approximée par une somme pondérée de n fonctions cosinus de type cos(a.x)*



Nasir Ahmed

T. Natarajan



K. R. Rao

JPEG : quelques détails

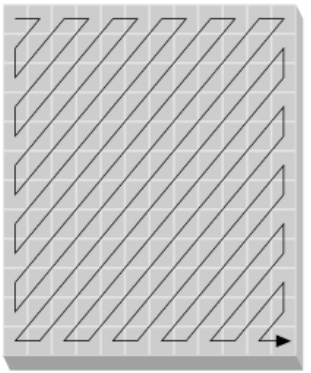
Pour chaque bloc 8x8 :

- On considère les 64 valeurs comme une fonction
- On applique à cette fonction la transformée en cosinus discrète (DCT, 1974)
- On divise chacun des 64 coefficients obtenus par une valeur définie par une matrice de quantification, et on arrondit. (plus les diviseurs sont grands, plus la compression est forte)

Exemple de matrice de quantification :

16	11	10	16	24	40	51	51
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

- La partie bas/droite a tendance à être remplie de coefficients 0. On prend donc un ordre “zig-zag” et on encode le tout avec RLE.
- On termine par un encodage de Huffman



MP3 (1993)

- Essentiellement la même méthode que pour JPEG.

- On utilise des “modèles psychoacoustiques” permettant de filtrer les sons en fonction de notre ouïe

MPEG

Principe de compression similaire.

On a régulièrement des images de référence (codées en JPEG): les i-frames (i=“intracoded”)

Pour les autres images (p-frames, p=“predicted”) :

- On code des valeurs de différence par rapport à l’image précédente
- On code des vecteurs de translation (pour des objets qui se déplacent)