Principes généraux de codage entropique d'une source

Cours : Compression d'images

Master II: IASIG

Dr. Mvogo Ngono Joseph

Table des matières

Objectifs

L'objectif de ce module est de rappeler les résultats fondamentaux du codage entropique ou compression reversible d'une source de données (texte, image, ...).

Ces résultats préliminaires seront utilisés dans notre méthodologie de compression des images de télédétection. Plusieurs types de méthodes de codage entropique seront abordées notamment les méthodes d'ordre zéro, non-adaptatives à savoir :

l'algorithme de codage de Shannon-Fano

l'algorithme de codage de Huffman

l'algorithme de codage arithmétique

Introduction

La notion de codage entropique ou compression réversible d'une source correspond à un codage sans perte des symboles de la source avec pour objectif d'atteindre une limite théorique du gain de compression de Shannon caractérisée par l'entropie de la source. Le but du codage entropique est de remplacer les messages émis par une source S utilisant un alphabet N-aire $\{s_1, s_2, \ldots\ldots, s_N\}$ par des messages écrits dans l'alphabet binaire utilisé par les systèmes de stockage d'information sur ordinateur. Les codes étudiés ici sont des codes de longueur variable, on suppose qu'un code associe à chaque symbole $s_i (i = 1, \ldots.. N)$ de la source un mot de code binaire m_i, séquence de l_i bits. Un tel code doit avoir des propriétés permettant de reconstruire sans ambiguïté les messages émis par la source, nous passerons en revue dans la section suivante quelques définitions et résultats fondamentaux de la théorie des codes.

Entropie d'une source

I

Définition : Notion d'entropie d'une source simple ou sans mémoire

Soit une source S définie par son alphabet $\{s_1, s_2, ..., s_N\}$ de symboles et ses caractéristiques d'émission régies par une loi de probabilité P : $\{P(s_1), P(s_2), ... P(s_N)\}$ .Une source sera dite simple (ou sans mémoire) si les symboles émis par la source S sont indépendants et de même loi. Une suite de N symboles émis aux instants $1, 2, ...n$ par S suit donc une loi de probabilité : $P(s_1, s_2, ... s_n) = p(s_1)\, p(s_2) ..... p(s_n)$.

L'entropie d'ordre zéro H(S) d'une source simple S, de loi de probabilité P, est définie par l'expression : $H(S) = -\sum_{i=1}^{N} p(s_i) \log_2(p(s_i))$.

L'entropie dans ce cas a la propriété suivante:

- H(S) est maximale si tous les symboles $\{s_1, s_2, ..., s_N\}$ de la source S sont équiprobables. Dans ce cas , l'entropie est égale à l'information associée à chaque message pris individuellement.

$$\forall\, i \in \{1,2,.....N\},\ p(s_i) = \frac{1}{N} \Leftrightarrow H(S) = \log_2(N).$$

Exemple : Calcul de l'entropie d'une source binaire

Considérons le cas d'une source binaire S dont l'alphabet est {0,1} tel que P(1) = p et donc P(0)=1-p

$$0 < p \leqslant 1$$

$$H(S) = -p log_2(p) - (1-p) \log_2(1-p) = f(p)$$

la fonction f(p) est symétrique par rapport à p=0.5 et est maximale pour p=0.5

Définition : Entropie d'ordre K

On suppose une source à amplitude discrète avec mémoire, Il existe alors des dépendances statistiques entre les échantillons successifs de la source.

Soit $s = s(n) = (s_n, s_{n+1}, ..., s_{n+K-1})^T$ un vecteur constitué de K échantillons successifs de la source. Ce vecteur est caractérisé par sa probabilité conjointe $P_S(s)$

qui est indépendante du temps si on considère une source stationnaire, s est une réalisation du vecteur aléatoire $S = (S_n, S_{n+1}, ...S_{n+K-1})^T$ .On désigne par entropie d'ordre k ou entropie conjointe des vecteurs aléatoires l'expression :

$$H_K(S) = \frac{1}{K} E(-\log_2 P_S(S)) = \frac{-1}{K} \sum_s \sum .... \sum p_S(s) \log_2 p_S(s)$$

II

Définitions et résultats fondamentaux liés au codage de source

Définition : Régularité

Un code est dit régulier si tous les mots de code m_i sont distincts. Tous les codes doivent au moins être réguliers pour permettre un décodage univoque.

Définition : Déchiffrabilité

Un code régulier est déchiffrable si pour toute suite $m_1, m_2, ...m_n$ de mots de code il est possible de distinguer les m_i sans ambiguïté et reconstruire ainsi les symboles s_i correspondants.

Pour les codes de longueur variable, cette propriété est satisfaite pour les codes dits sans préfixes, obtenus en évitant qu'un mot du code ne soit identique au début d'un autre.

Définition : Codes instantanés

On dit d'un code qu'il est à décodage instantané s'il est possible de décoder les mots de code dès lors que tous les symboles qui en font partie ont été reçus.

Définition : Extension d'un code

L'extension d'ordre n d'un code est le code formé par les séquences de n mots du code initial.

Fondamental : Condition nécessaire et suffisante de déchiffrabilité

Pour qu'un code soit déchiffrable il faut et il suffit que toutes ses extensions soient régulières.

Attention

L'inégalité de Kraft constitue un résultat fondamental en théorie des codes. Elle fournit une condition nécessaire et suffisante d'existence de codes déchiffrables et instantanés, exprimée en fonction de la longueur des mots de code.

Fondamental : Inégalité de Kraft

Soient $l_1, l_2, ..., l_N$ des longueurs de mots candidates pour coder une source N-aire dans un alphabet binaire. Alors l'inégalité de Kraft :

$$\sum_{i=1}^{N} \frac{1}{2^{li}} \leq 1$$

est une condition nécessaire et suffisante d'existence de codes déchiffrables et instantanés respectant ces longueurs de mots.

Fondamental : THÉORÈME DE SHANNON

- Limite inférieure de la longueur moyenne d'un code.

Soit une source stationnaire sans mémoire, la longueur moyenne $\bar{n}$ des mots codés m_i est limitée par la valeur de l'entropie de la source S.

$$\bar{n} = \sum_{i=1}^{N} l_i p(s_i) \geq \sum_{i=1}^{N} p(s_i) \log_2(p(s_i)).$$

- Borne supérieure de la limite supérieure (code de Shannon)

  Il est également possible de trouver un code déchiffrable tel que :

  $H(S) \leqslant \bar{n} < H(S) + 1$.

Les algorithmes de codage statistiques

III

Le principe des algorithmes de codage statistique est d'utiliser les probabilités d'occurence de chaque symbole dans une séquence de symboles émanant de la source. Dans le cadre de ce cours nous examinerons dans cette catégorie deux algorithmes.

Méthode : Algorithme de Shannon-Fano

L'algorithme comporte les étapes suivantes :

1) Les probabilités d'apparition de chaque symbole sont placées dans un tableau trié par ordre décroissant de probabilités .

2) Le tableau est coupé en deux groupes de symboles S_0 et S_1 dont la somme des probabilités de chaque groupe avoisine 0.5.

3) Le groupe S_0 est codé par un "0" et S_1 par un "1".

4) Si un groupe S_i n'a qu'un seul élement, c'est une feuille terminale, sinon la procédure reprend récursivement à l'étape 2 sur le groupe S_i.

Exemple : Application de l'algorithme de Shannon-Fano

1. Pour illustrer cet algorithme, nous allons coder la phase suivante :

"Le codage est indispensable"

Pour simplifier nous n'allons pas prendre en compte le symbole espace (blanc).

cette phase est une source de 24 symboles

Tous ces symboles émanent de l'alphabet.

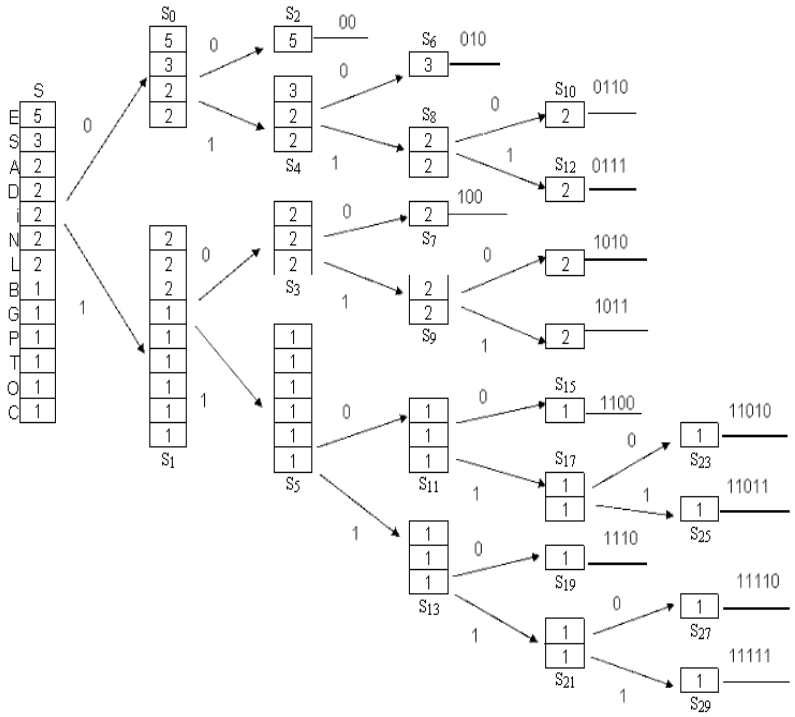
$A=\{E,S,A,D,I,N,L,B,G,P,T,O,C\}$

Cet Alphabet a N=13 symboles

- ETAPE 1

Symbole	Nombre de fois	Probabilités
E	5	5/24
S	3	3/24
A	2	2/24
D	2	2/24
i	2	2/24
N	2	2/24
L	2	2/24
B	1	1/24
G	1	1/24
P	1	1/24
T	1	1/24
O	1	1/24
C	1	1/24

- Application successive des étapes 2,3 et 4



Exemple : Algorithme de Shannon-Fano

Méthode : Algorithme de Huffman

Comme pour le codage de shannon-Fano, les probabilités d'apparition des symboles sont placées dans un tableau trié par ordre décroissant de probabilités. L'algorithme de Huffman est implémenté suivant une structure d'arbre.

Le principe de cet algorithme consiste à regrouper les deux symboles de probabilités la plus faible pour en faire un nouveau symbole dont la probabilité est la somme des probabilités de ces deux symboles. On itère cette opération et à chaque étape le nombre de symboles diminue. On construit de cette manière un arbre dont les feuilles sont les symboles à coder et les embranchements les codages intermédiaires.

Exemple : Algorithme de Huffman

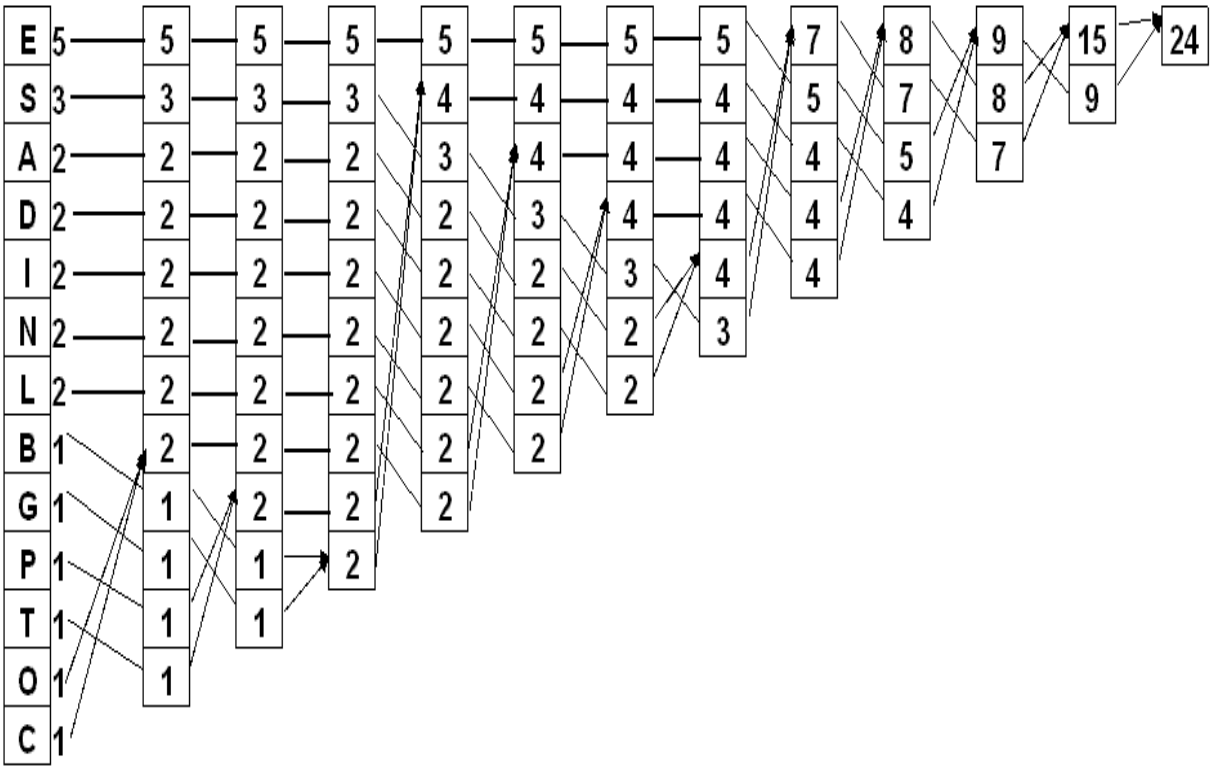
Considérons à nouveau l'exemple précédent :

Il s'agit de coder la phrase : "Le codage est indispensable".

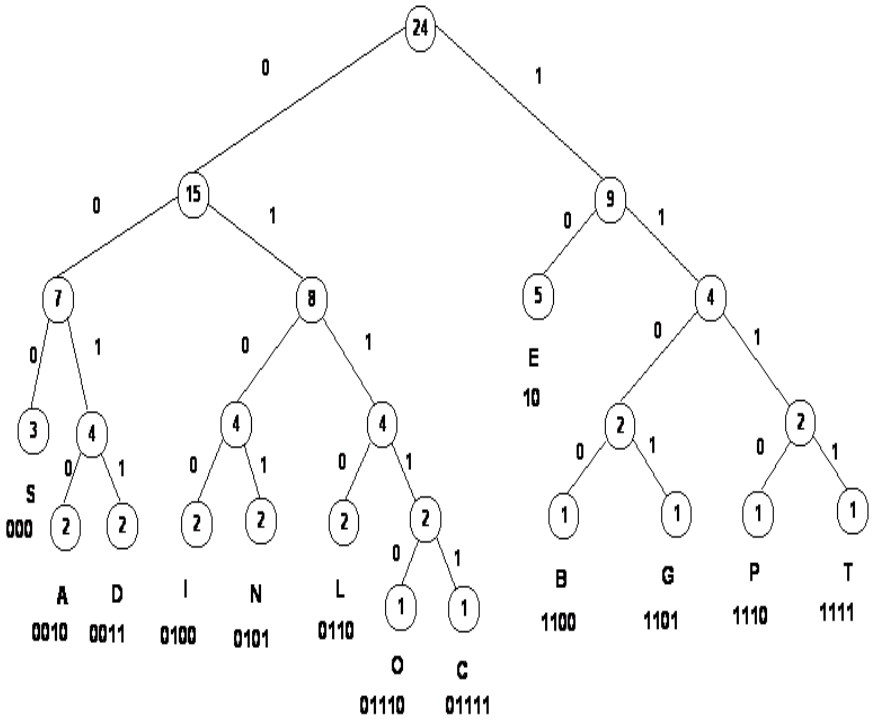
Le tableau des probabilités d'occurrence est le suivant :

Symbole	Nombre de fois	Probabilités
E	5	5/24
S	3	3/24
A	2	2/24
D	2	2/24
i	2	2/24
N	2	2/24
L	2	2/24
B	1	1/24
G	1	1/24
P	1	1/24
T	1	1/24
O	1	1/24
C	1	1/24

On applique successivement les étapes de l'algorithme selon le principe de l'algorithme précédemment indiqué.



L'arbre de Huffman associé est présenté ci-après :



Algorithme de codage arithmétique

IV

Principes de base

Contrairement aux deux algorithmes précédents, le code est associé à la séquence et non chaque symbole pris individuellement.

On associe à chaque symbole à coder un intervalle [a_k , b_k [de [0,1[

On itère ce processus jusqu'à ce que toute la séquence soit traitée

La séquence est alors codée par un réel de l'intervalle [0,1[

Méthode : Algorithme de codage

Cet algorithme comporte 5 étapes successives :

1) On initialise l'intervalle de codage [a_c,b_c[avec les valeurs a_c=0 et b_c=1, cet intervalle a une largeur L=b_c-a_c=1

2) cet intervalle est partitionné en N sous-intervalles (N nombre de symboles de l'alphabet de la source) proportionnellement aux probabilités p(s_k) de chaque symbole s_k. cette partition est constituée de sous-intervalles [a_k,b_k[tels que :

b_k-a_k=p(s_k) avec a_k=a_c+largeur x $\sum_{i=1}^{k-1} p(s_i)$ et b_k=a_c+largeur x $\sum_{i=1}^{k} p(s_i)$

3) On choisit le sous-intervalle correspondant au prochain s_k à coder dans la séquence et on met à jour les valeurs a_c et b_c de la manière suivante : a_c=a_c+largeur x a_k et b_c=a_c+largeur x b_k

4) Avec le nouvel intervalle [a_c,b_c[on recommence le processus de l'étape 2.

5) Les étapes 2,3 et 4 sont répétés jusqu'à épuisement des symboles de la séquence et obtention du dernier intervalle [a_c,b_c[

La représentation binaire de tout réel x_c de l'intervalle [a_c,b_c[est un code de la séquence.

Exemple : Codage arithmétique d'une source

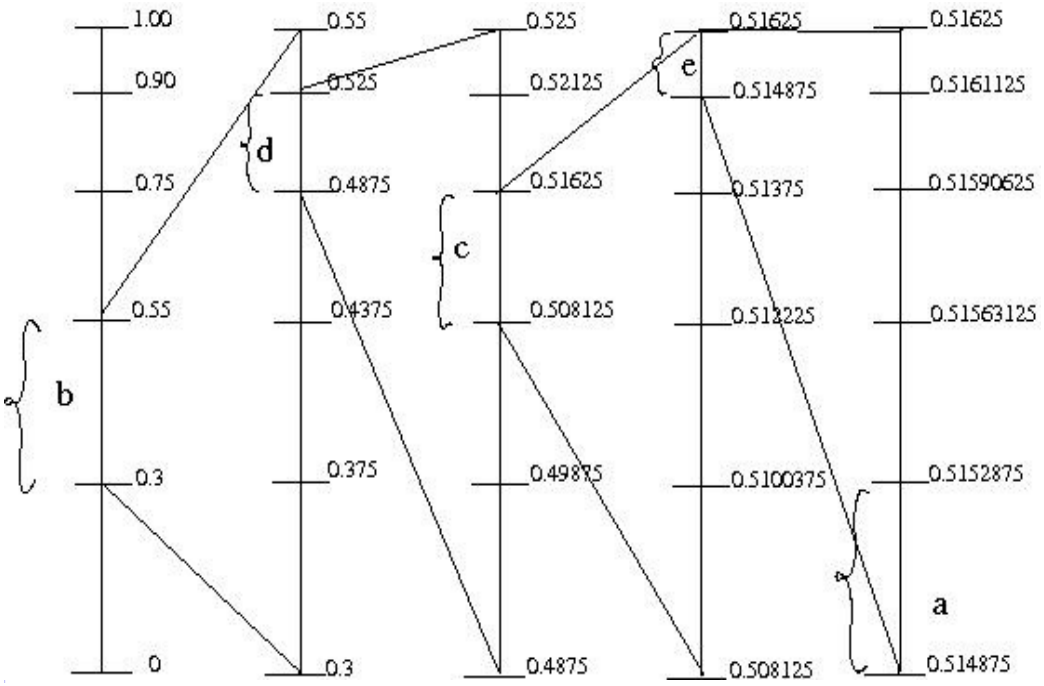
On considère la source S={a,b,c,d,e} avec les probabilités respectives d'occurrence des symboles suivantes :

P(a)=0.3, P(b)=0.25,P(c)=0.20, P(d)=0.15, P(e)=0.1

On souhaite coder la séquence bdcea

Pour coder cette séquence on divise l'intervalle [0,1[en 5 sous-intervalles, puis on se place sur le sous-intervalle correspondant au premier symbole de la séquence à coder, il s'agit du symbole "b". Pour le symbole suivant de la séquence "d" on subdivise le sous intervalle de b, [0.3,0.55[en 5 sous-intervalles correspondant au nombre de symboles de l'alphabet de la source S.

On procède ainsi récursivement pour toute la séquence (voir figure ci-dessous). (Cliquez pour afficher)



Méthode : Algorithme de décodage

Cet algorithme comporte six étapes successives :

1) On initialise $a_c=0$ et $b_c=1$

2) On calcule la largeur du sous-intervalle du code : largeur = b_c-a_c

3) On trouve le sous-intervalle $[a_k, b_k[$ du symbole s_k avec $1 \leqslant k \leqslant N$ tel que : $a_k \leqslant \frac{(x_c - a_c)}{largeur} < b_k$

On rappelle que x_c est le réel codant la séquence.

4) On obtient le symbole s_k

5) On met à jour le sous-intervalle de codage : $a_c=a_c+\text{largeur} \times a_k$ et $b_c=a_c+\text{largeur} \times b_k$

6) On répète les 2,3,4 et 5 jusqu'à obtenir le décodage de tous les symboles de la séquence.

Exemple : Décodage de l'exemple précédent

On applique l'algorithme de décodage à l'exemple précédent :

On considère la valeur $x_c=0.51508125$ codant la séquence

Etape 1 :

On initialise $a_c=0$ et $b_c=1$

Etape 2 :

On calcule la largeur du sous-intervalle du code : largeur = b_c-a_c =1

Etape 3 :

On calcule le nombre $\frac{x_c - a_c}{largeur}$ dont la valeur est 0.51508125 et on cherche k tel que ce nombre soit compris dans la partition initiale.

Etape 4 :

k=2, Il s'agit du sous-intervalle [0.3,0.55[qui correspond au symbole b.

Etape 5 :

On met à jour le sous-intervalle de codage $a_c=a_c+\text{largeur} \times a_k$ et $b_c=a_c+\text{largeur} \times b_k$
$a_c=0 + 1x0.3=0.3$
$b_c=0+1 \times 0.55=0.55$
On répète l'étape 2 : largeur = 0.55-0.3=0.25
Etape 3 : (0.51508125-0.3)/0.25 = 0,860325
Etape 4 : k=4, il s'agit du sous-intervalle [0.75,0.90[qui correspond au symbole d.
On revient à l'étape 5 et ainsi de suite ...

Conclusion

Dans le cadre de ce module, nous avons rappelé quelques résultats liés au codage de source et avons présenté quelques algorithmes de codage entropique d'une source notamment le codage de Shannon-Fano, le codage de Huffman et le codage arithmétique.

Ces algorithmes ont la particularité d'avoir une longueur moyenne de code qui approche la limite de Shannon qui est l'entropie de la source. Les algorithmes étudiés dans ce module implémentent les méthodes d'ordre zéro, non adaptatives. Le lecteur intéressé par la théorie des codes pourrait avantageusement étudier les méthodes d'ordre supérieur, les méthodes adaptatives et les algorithmes à base de dictionnaire tel que l'algorithme de Lempel,Ziv et Welsch.