Codage statistique

Codage d'Huffman

Codage statistique optimal

- ***Définitions***:
  - S: source discrète et simple de messages m_i de loi de probabilité $p = (p_1, \ldots\ldots\ldots\ldots, p_N)$ (source homogène)
  - Codage sur un alphabet $A = \{ a_1, a_2, \ldots\ldots, a_q \}$
  - Entropie de la source H(S) et longueur moyenne des mots-codes E(n)
- ***Théorème de Mac Millan***:
  - il existe au moins un code irréductible déchiffrable vérifiant :

$$H / \log2\ q \leq E(n) < (H / \log2\ q) + 1$$

$\Rightarrow$ égalité si p_i de la forme : $p_i = q^{-ni}$ (si $q = 2$ => $n_i = - \log2\ p_i$)

- ***Théorème de SHANNON*** (1^{er} théorème sur le codage sans bruit)

$$H / \log2\ q \leq E(n) < (H / \log2\ q) + \varepsilon$$

Nous avons vu dans la ressource précédente : « Définition du codage et propriétés» que les objectifs du codage sont principalement de transcrire des informations et de réduire la quantité de symboles nécessaires à la représentation de l'information. En « optimisant le codage », on souhaite donc réduire au maximum cette quantité de symboles :

On considère une source simple d'information homogène $S = \{m_1, m_2, \ldots, m_N\}$ munie d'une loi de probabilité $p = \{ p_1, p_2, \ldots, p_N \}$ où $p_i = Pr\{m = m_i\}$.
Si M_i est le mot-code correspondant au message m_i, on appelle $n_i = n(M_i)$ le nombre de caractères appartenant à l'alphabet A (Card(A) = q) nécessaires au codage de m_i, n_i est donc la longueur du mot-code M_i.

La longueur moyenne des mots-codes est donc : $E(n) = \sum_{i=1}^{N} p_i n_i$.

L'incertitude moyenne d'une source est l'entropie H. L'incertitude moyenne par caractère de l'alphabet A est donc égale à $\frac{H}{E(n)}$, elle vérifie : $\frac{H}{E(n)} \leq \log_2 q$ car l'alphabet A comporte « q » caractères. De cette inégalité, on déduit que $E(n) \geq \frac{H}{\log_2 q}$.

Pour optimiser le codage, on souhaite donc réduire E(n), la longueur moyenne des mots-codes. Cette longueur moyenne ne peut pas être rendue inférieure à $\frac{H}{\log_2 q}$. Néanmoins, deux théorèmes démontrent qu'il est possible d'obtenir l'égalité $E(n) = \frac{H}{\log_2 q}$ et d'obtenir un code optimal :

- ***Théorème de Mac Millan :***

Une source d'information S d'entropie H codée de façon déchiffrable par un alphabet à q caractères est telle que : $E(n) \geq \frac{H}{\log_2 q}$ et il existe au moins un code irréductible d'une loi donnée qui vérifie : $E(n) \leq \frac{H}{\log_2 q} + 1$. L'égalité est atteinte si $p_i = q^{-ni}$ (i.e. $n_i = - \log_q p_i$) et on a alors un codage optimal.

Remarque :

Dans le cas particulier d'un l'alphabet binaire {0, 1}, on a q = 2. Si la relation $n_i = - \log_2 p_i$ est vérifiée, on a alors E(n) = H : l'entropie est la borne inférieure de l'ensemble des longueurs moyennes des mots-codes et cette borne inférieure est atteinte.

- ***Théorème de SHANNON sur le codage sans bruit :***

Toute source d'information homogène est telle qu'il existe un codage irréductible pour lequel la longueur moyenne des mots-codes est aussi voisine que l'on veut de sa borne inférieure $\frac{H}{\log_2 q}$. La démonstration de ce théorème utilise le codage irréductible en blocs (le codage en bloc affecte à chaque bloc de « k » messages de S, consécutifs ou non, un mot-code).

Codage statistique optimal

- ***Codes de Fano - Shannon***

- ***Codes arithmétiques*** *(codage en bloc, de type codage d'intervalles)*
  possibilités d'adaptation en ligne

- ***Codes de Huffman***
  3 principes de base:
  - si $p_i < p_j \Rightarrow n_i \geq n_j$
  - les 2 codes les moins probables sont de même longueur
  - les 2 codes les moins probables (de longueur max) ont le même préfixe de longueur n_{max}-1

La présentation ci-dessus met en avant trois types de codes qui s'approchent du code optimal :

♦ ***Le codage de Fano- Shannon :***

Il essaie d'approcher au mieux le code irréductible le plus compact, mais la probabilité p_i ne s'écrit généralement pas 2^{-ni} et donc le codage ne peut qu'approcher le codage optimal.
Les probabilités p_i associées aux messages m_i sont ordonnées par ordre décroissant puis on fixe n_i tel que : $n_i \geq \log_2 \frac{1}{p_i} \geq n_i - 1$. Enfin, on choisit chaque mot code M_i de longueur n_i de sorte qu'aucun des mots-codes choisis avant n'en soit un préfixe (évite les ambiguïtés au décodage cf. : « *Définition du codage et propriétés* »).

♦ ***Le codage arithmétique :***

Le code est associé à une séquence de messages émis par la source et non pas à chaque message. Contrairement au code de Huffman qui doit avoir une longueur entière de bits par message, et qui ne permet donc pas toujours de réaliser une compression optimale, le codage arithmétique permet de coder un message sur un nombre non entier de bits : c'est une méthode plus performante, mais aussi plus lente. Les différents aspects de ce codage sont amplement développés dans la ressource : « Codage statistique : codage arithmétique ».

- ***Le codage de Huffman :***

Le code de Huffman est le code irréductible optimal. Il s'appuie sur trois principes :

- si $p_j > p_i$ alors $n_i \leq n_j$,
- les deux mots les moins probables ont des longueurs égales,
- ces derniers sont écrits avec les mêmes n_{max}-1 premiers caractères.

En utilisant itérativement cette procédure, on construit les mots-codes M_i des messages m_i.

- Exemple :

On considère la source $S = \{m_1, m_2, \ldots, m_8\}$ munie de la loi de probabilité : $p_1 = 0,4$; $p_2 = 0,18$; $p_3 = p_4 = 0,1$; $p_5 = 0,07$; $p_6 = 0,06$; $p_7 = 0,05$; $p_8 = 0,04$.

On place ces probabilités par ordre décroissant dans la colonne $p_i^{(0)}$ du tableau ci-dessous. On remarque que dans la colonne $p_i^{(0)}$ les probabilités des messages m_7 et m_8 sont les plus faibles, on les somme alors puis on réordonne, toujours par ordre décroissant, les probabilités afin de créer la colonne $p_i^{(1)}$:

Messages	$p_i^{(0)}$	$p_i^{(1)}$
m_1	0,4	0,4
m_2	0,18	0,18
m_3	0,1	0,1
m_4	0,1	0,1
m_5	0,07	0,09
m_6	0,06	0,07
m_7	0,05	0,06
m_8	0,04	

De manière générale, on somme les deux probabilités les plus faibles de la colonne $p_i^{(k)}$, puis on réordonne les probabilités par ordre décroissant afin d'obtenir la colonne $p_i^{(k+1)}$. Finalement on a donc le tableau suivant :

Messages	$p_i^{(0)}$	$p_i^{(1)}$	$p_i^{(2)}$	$p_i^{(3)}$	$p_i^{(4)}$	$p_i^{(5)}$	$p_i^{(6)}$
m_1	0,4	0,4	0,4	0,4	0,4	0,4	0,6
m_2	0,18	0,18	0,18	0,19	0,23	0,37	0,4
m_3	0,1	0,1	0,13	0,18	0,19	0,23	
m_4	0,1	0,1	0,1	0,13	0,18		
m_5	0,07	0,09	0,1	0,1			
m_6	0,06	0,07	0,09				
m_7	0,05	0,06					
m_8	0,04						

On attribue les bits ‘0’ et ‘1’ aux deux derniers éléments de chaque colonne :

Messages	$p_i^{(0)}$	$p_i^{(1)}$	$p_i^{(2)}$	$p_i^{(3)}$	$p_i^{(4)}$	$p_i^{(5)}$	$p_i^{(6)}$
m_1	0,4	0,4	0,4	0,4	0,4	0,4	0,6 '0'
m_2	0,18	0,18	0,18	0,19	0,23	0,37 '0'	0,4 '1'
m_3	0,1	0,1	0,13	0,18	0,19 '0'	0,23 '1'	
m_4	0,1	0,1	0,1	0,13 '0'	0,18 '1'		
m_5	0,07	0,09	0,1 '0'	0,1 '1'			
m_6	0,06	0,07 '0'	0,09 '1'				
m_7	0,05 '0'	0,06 '1'					
m_8	0,04 '1'						

Pour chaque message m_i, on parcourt le tableau de gauche à droite et on détecte dans chaque colonne la probabilité associée $p_i^{(k)}$ (chemin en bleu sur la figure ci-dessous).
Le mot-code M_i est alors obtenu en partant de la dernière colonne (à droite) et en remontant vers la première colonne (à gauche), tout en sélectionnant les bits associés aux probabilités $p_i^{(k)}$ du message m_i (rectangles verts sur la figure ci-dessous).

On propose par exemple de déterminer le mot-code M_6 du message m_6. On détecte donc toutes les probabilités $p_6^{(k)}$:

Messages	$p_i^{(0)}$	$p_i^{(1)}$	$p_i^{(2)}$	$p_i^{(3)}$	$p_i^{(4)}$	$p_i^{(5)}$	$p_i^{(6)}$
m_1	0,4	0,4	0,4	0,4	0,4	0,4	0,6 '0'
m_2	0,18	0,18	0,18	0,19	0,23	0,37 '0'	0,4 '1'
m_3	0,1	0,1	0,13	0,18	0,19 '0'	0,23 '1'	
m_4	0,1	0,1	0,1	0,13 '0'	0,18 '1'		
m_5	0,07	0,09	0,1 '0'	0,1 '1'			
m_6	0,06	0,07 '0'	0,09 '1'				
m_7	0,05 '0'	0,06 '1'					
m_8	0,04 '1'						

Le mot-code M_6 est alors obtenu par simple lecture de droite à gauche des bits contenus dans les rectangles verts : '0' – '1' – '0' – '1'. En procédant de même pour chacun des messages, on obtient :

Messages	code de Huffman
m_1	1
m_2	0 0 1
m_3	0 1 1
m_4	0 0 0 0
m_5	0 1 0 0
m_6	0 1 0 1
m_7	0 0 0 1 0
m_8	0 0 0 1 1

La longueur moyenne des mots-codes vaut donc :

$$E(n) = \sum_{i=1}^{8} p_i \, n_i = 0{,}4\times1 + 0{,}18\times3 + 0{,}1\times3 + 0{,}1\times4 + 0{,}07\times4 + 0{,}06\times4 + 0{,}05\times5 + 0{,}04\times5$$

$$E(n) = 2{,}61$$

On peut comparer cette grandeur avec l'entropie H de la source :

$$H = -\sum_{i=1}^{8} p_i \, . \log_2 p_i = 2{,}552$$

L'efficacité η du code de Huffman pour cet exemple est donc de $\frac{2{,}552}{2{,}61} = 97{,}8\ \%$. À titre de comparaison, il faut 3 bits pour coder en binaire naturel 8 messages différents ($2^3 = 8$). Pour cet exemple, l'efficacité du code binaire naturel est donc de seulement $\frac{2{,}552}{3} = 85\ \%$.