ACCQ202 Sept. 13th 2018

Cours 1

Enseignant: A. Tchamkerten *Crédit: D. Fernandes, H. Hmida*

Refs: Cover and Thomas Chapters 5.1-5.6, 2, 2.1, 2.3

1 Source d'information et codes

Expérience répétée plusieurs fois avec résultats $\in \{a_1, a_2, a_3, a_4\}$

On enregistre les résultats sur un disque dur.

Méthode 1:	Méthode 2:
$a_1 \to 00$	$a_1 \to 0$
$a_2 \to 01$	$a_2 \to 10$
$a_3 \to 10$	$a_3 \to 110$
$a_4 \to 11$	$a_4 \to 111$

Comment choisir la meilleure méthode, celle qui minimise le nombre de bits stocké? Cela dépend des résultats d'expérience; si "a_1" domine, alors la Méthode 2 sera plus efficace, alors que si "a_3" ou "a_4" (ou les deux) domine, alors ce sera la Méthode 1. Ceci motive à définir une source d'information de façon probabiliste:

Definition 1 (Source d'information) *Une source d'information est une variable aléatoire $X \in \mathcal{X}$ définie sur un alphabet $\mathcal{X}$ de taille finie. On fait l'hypothèse que X suit une loi P_X connue. On note également $p_i := Pr(X = i)$.*

Definition 2 (Code) *Un code binaire est une fonction* $C : \mathcal{X} \to \{0,1\}^*$
$x \mapsto C(x)$ *mot code*

Definition 3 (Longueur moyenne et taux de compression) *La longueur moyenne d'un code C associé à une source $X \sim P_X$ est :*

$$L(C) = \sum p_i l_i \text{ bits/symbol source}$$

où l_i désigne la longueur du i-ème mot code. Le taux de compression est défini comme $1/L(C)$.

Exemple 1 $X \in \{1,2,3,4\}$ *avec*
$p_1 = 1/2$, $p_2 = 1/4$, $p_3 = p_4 = 1/8$ *et*
$C(1) = 1$, $C(2) = 10$, $C(3) = 110$, $C(4) = 111$ *donne*

$$L(C) = 7/4 \text{ bits/symbol source}$$

Remarque 1 *L'alphabet $\mathcal{X}$ représente l'ensemble des charactères apparaissant dans le fichier à comprimer. Cela à l'air bien défini, mais en fait il y a beaucoup de possibilités. Considérons un fichier contenant n séquences "aaaa" et n séquences "bbbb" concaténées. Si l'alphabet est $\mathcal{X} = \{a, b\}$ alors le fichier pourra être stocké sur $8n$ bits, alors que si l'alphabet est $\mathcal{X}' = \{aaaa, bbbb\}$ le fichier pourra être stocké sur $2n$ bits. Dans le premier cas on a $L_{\mathcal{X}}(C) = 1$ bit/symbol source et dans le deuxième on a $L_{\mathcal{X}'}(C) = 1/4$ bits/symbol source.*

Implicitement, on veut qu'un code puisse être décodé de manière fiable/unique.

Definition 4 *Un code est dit non-singulier si $C(x_i) \neq C(x_j)\ \forall\ i \neq j$*

Exemple 2 $\mathcal{X} = \{a, b, c\}$ *avec* $C(a) = 0$, $C(b) = 01$, $C(c) = 10$.

Cependant *accc* peut être confondu avec *bbba*. Donc un code non singulier n'est pas forcément "à décodage unique". Pour définir ce dernier on définit l'extension d'un code:

Definition 5 *L'extension de C est l'application $\tilde{C} : \mathcal{X}^* \to \{0,1\}^*$ définie par $\tilde{C}(x_1,.....,x_n) = C(x_1)*.....*C(x_n)\ \forall\ (x_1,.....,x_n), \forall\ n \geq 1$, où * représente l'opération de concaténation.*

Definition 6 *Un code est à décodage unique si son extension est non singulière.*

Definition 7 *Un code est instantané (ou sans préfixe) si aucun mot code n'est le préfixe d'un autre.*

Clairement, un code instantané est à décodage unique.

Exemple 3

$\mathcal{X}$	*singulier*	*unique non inst.*	*inst.*
1	*0*	*10*	*0*
2	*0*	*01*	*10*
3	*0*	*11*	*01*
4	*0*	*110*	*111*

2 Inégalité de Kraft

Theorem 8 *(Inégalité de Kraft): Soit C un code instantané alors:*

$$\sum_i 2^{-l_i} \leq 1.$$

Inversement, si $\{l_i\}$ *satisfait cette inégalité alors* $\exists$ *C instantané avec ces longueurs de mots code.*

Theorem 9 *(McMillan) Le théorème précédent est valable verbatim en remplaçant "instantané" par "à décodage unique".*

(Preuves des théorèmes 8 et 9 voir cours)

Implication importante:

$$\min_{C \text{ a dec. unique}} L(C) = \min_{C \text{ inst.}} L(C)$$

En effet, en posant $L_1 = min_{Cadec.unique}$ et $L_2 = min_{Cinst.}$
$\{$codes inst. $\} \subset \{codesadec.unique\} \Rightarrow L_1 \leq L_2$
et $L_1 \geq L_2$ car: si C^* est un code à décodage unique alors $\{l_i^*\}$ vérifie

$$\sum 2^{-l_i^*} \leq 1$$

(Th.9) et par Th.8 il existe un un code instantané avec les $\{l_i^*\}$ pour longueurs.

3 Borne entropique

But trouver

$$\min_{C\text{: à décodage instantané}} L(C) = \min_{C:\sum_i 2^{-l_i} \leq 1} \sum_i p_i l_i.$$

Le problème d'optimisation de droite est linéaire en sa fonction objectif mais la contrainte ne l'est pas. De plus, les l_i sont entiers.

Afin de simplifier le problème, ignorons les contraintes d'intégralité et supposons que le minimum sature la contrainte, i.e., les l_i optimaux sont tels que $\sum_i 2^{-l_i} = 1$. Dans ce cas on peut trouver un points stationaire du Lagrangien associé au problème d'optimisation:

$$J(\lambda, \{l_i\}) = \sum_i p_i l_i - \lambda(\sum_i 2^{-l_i} - 1)$$

$$\frac{\partial J}{\partial \lambda} = 0 \Leftrightarrow \sum_i 2^{-l_i} = 1 \tag{1}$$

$$\frac{\partial J}{\partial l_i} = 0 \Leftrightarrow p_i - \lambda 2^{-l_i} \log_e 2 = 0 \tag{2}$$

De 1 et 2:
$p_i = \lambda 2^{-l_i} \log_e 2 \Rightarrow \sum_i p_i = \lambda \sum_i 2^{-l_i} \log_e 2 \Rightarrow \lambda^* = \frac{1}{\ln 2}$

$\Rightarrow l_i^* = \log_2 \frac{1}{p_i}$(point stationnaire) Ce point stationnaire donne une fonction objectif égal à

$$\sum_i p_i l_i^* = -\sum_i p_i \log_2 p_i$$

Vu les hypothèses faite, $-\sum_i p_i \log_2 p_i$ représente la fonction objectif évalué à un points stationaire du Lagrangien d'un problème d'optimisation autre que celui d'origine. Par magie, on verra plus bas que $-\sum_i p_i \log_2 p_i$ est la borne ultime de compression.

Definition 10 *L'entropie d'une source d'information $X \sim P$ est:*

$$H(X) = -\sum_i p_i \log_2 p_i. \tag{3}$$

Theorem 11 *(Inégalité de Jensen)*
Soit f une fonction convexe, alors

$$Ef(X) \geq fE(X).$$

Si f est strictement convexe, alors l'égalité implique que X est constant (presque partout).

Proof Soit $g(x)$ la tangente à $f(x)$ au point $x = m = EX$. On a donc

$$g(x) = f(m) + \alpha(x - m)$$

où α est la pente de g en m. Puisque f est convexe $f(x) \geq g(x)$ pour tout x. Donc

$$Ef(X) \geq Eg(X) = E(f(m) + \alpha(x - m)) = f(m) = f(EX).$$

Si f est strictement convexe, $f(x) > g(x)$ sauf si $x = m$. Donc si X n'est pas constante, $Ef(X) > Eg(X)$. ■

Definition 12 *Soit P, Q deux distributions sur un même alphabet. La divergence entre P et Q est:*

$$D(P||Q) := \sum_i p_i \log \frac{p_i}{q_i}.$$

Lemma 13

$$D(P||Q) \geq 0 \forall P, Q$$

avec égalité si et seulement si $p_i = q_i \forall i$.

Proof

$$\begin{aligned} -D(P||Q) &= \sum_i p_i \log \frac{q_i}{p_i} \\ &= \sum_x P(x) \log \frac{P(x)}{Q(x)} \\ &= E\left(\log \frac{Q(X)}{P(X)}\right) \\ &\leq \log E\left(\frac{Q(X)}{P(X)}\right) \\ &= 0 \end{aligned}$$

par l'inégalité de Jensen, avec égalité si et seulement si

$$q_i = cp_i$$

pour une constante ci. En sommant sur les i de part et d'autre de cette derni´ere égalité on déduis que $c = 1$. ■

Theorem 14 *(Théorème fondamental de codage de source) $L(C) \geq H(X)$ quelque soit C à décodage unique avec égalité ssi la source est diadique, i.e., $p_i = 2^{-l_i}$ avec l_i entier.*

Proof Soit $c = \sum_i 2^{-l_i}$ et soit $q_i = 2^{-l_i}/c$ (les q_i sont une probabilité)

$$\begin{aligned} L(C) - H(X) &= \sum_i p_i l_i + \sum_i p_i \log_2 p_i \\ &= \sum_i p_i \log_2 \frac{1}{2^{-l_i}} + \sum_i p_i \log_2 p_i \\ &= \sum_i p_i \log_2 \frac{p_i \cdot c}{2^{-l_i} \cdot c} \\ &= D(P||Q) - \log c \\ &\geq 0 \end{aligned}$$

puisque $D(P||Q) \geq 0$ et $c \leq 1$ par le théorème de McMillan, car C est à décodage unique. L'inégalité est une égalité si et seulement si $p_i = q_i$ pour tout i et $c = 1$, i.e., si et seulement si $p_i = q_i$ car la condition $c = 1$ est redondante. ■

4 Code de Shannon

Idée : $l_i = \left\lceil \log \frac{1}{p_i} \right\rceil$

$\sum_i 2^{-l_i} = \sum_i 2^{-\left\lceil \log \frac{1}{p_i} \right\rceil} \leq \sum_i 2^{-\log \frac{1}{p_i}} = 1$ donc l'inégalité de Kraft est satisfaite et donc on peut trouver un code à décodage unique avec ces longueurs.

En utilisant que

$$\log \frac{1}{p_i} \leq l_i \leq \log \frac{1}{p_i} + 1$$

on a immédiatement que pour un code de Shannon

$$H(X) \leq L(C) \leq H(X) + 1.$$

Et en conséquence, les meilleurs codes sont au plus à 1 bit de la borne ultime de compression.

5 Codage en bloc

$X^n = (X_1, \cdots, X_n) \to c(X_1, \cdots, X_n)$ et on suppose que les X_i sont i.i.d. avec $X_i \sim X$ pour tout i.

$L_n = \frac{1}{n} El(X^n) = \frac{1}{n} \sum_{X^n} p(x^n) l(x^n)$
$H(X^n) \leq El(X^n) \leq H(X^n) + 1$

$$\begin{aligned} H(X^n) &= -\sum_{x^n} p(x^n) \log p(x^n) \\ &= -\sum_{x^n} p(x^n) (\sum_i \log p(x_i)) \\ &= -\sum_i \sum_{x^n} p(x^n) \log p(x_i) \\ &= \sum_i H(X) \end{aligned}$$

$\Rightarrow H(X^n) = n \cdot H(X)$
$\Rightarrow n \cdot H(X) \leq El(X^n) \leq n \cdot H(X) + 1$

et donc $H(X) \leq \frac{1}{n} El(X^n) \leq H(X) + \frac{1}{n}$. En d'autre terme, en codant par bloc et non pas symbol par symbol la longueur moyenne par unite de symbol source est au plus a $1/n$ bit de l'optimal (au lieu de au plus 1 bit de l'optimal).

6 Codage de Huffman

Exemple
Soit $P = (0,25; 0,25; 0,2; 0,15; 0,15)$ une distribution de probabilité. Construisons un code de Huffman correspondant à cette distribution.

0,25	0,25	0,2	0,15	0,15
{0,15 ; 0,15}	0,25	0,25	0,2	
{0,25 ; 0,2 }	{0,15 ; 0,15}	0,25		
{{0,15 ; 0,15} ; 0,25 }	{0,25 ; 0,2 }			
{{{0,15 ; 0,15} ; 0,25 } ; {0,25 ; 0,2 }}				

On a ainsi le codage suivant :

0,25	0,25	0,2	0,15	0,15
01	10	11	000	001

Exemples
Pour $(p_1 = 0,9999; p_2 = 0,0001)$, un code de Huffman peut donner $l_1 = l_2 =$

1, alors qu'un code de Shannon donnera $l'_1 = \left\lceil \log_2 \frac{1}{0{,}9999} \right\rceil = 1$ et $l'_2 = 14$.

Par contre, pour ($p_1 = \frac{1}{3}, p_2 = \frac{1}{3}, p_3 = \frac{1}{4}, p_4 = \frac{1}{12}$, un code de Huffman peut donner $l_1 = 1$, $l_2 = 2$ et $l_3 = l_4 = 3$, tandis qu'un code de Shannon donnera $l'_1 = l'_2 = l'_3 = 2$ et $l'_4 = 4$. On voit ainsi qu'un code de Huffman n'a pas forcément les longueurs optimales pour chacun des mots-code ($l'_3 < l_3$).

Theorem 15 *Un code de Huffman minimise $L(C)$ parmi les codes à décodage unique.*

Lemma 16 *Pour toute source d'information suivant la probabilité P, il existe un code optimal tel que :*

1. $\forall j, k \quad p_j > p_k \Rightarrow l_k \geq l_j$;
2. *chaque branche de la représentation arborescente du code à une sœur ;*
3. *les mots-code associés aux deux probabilités les plus petites sont représentés par des branches soeurs.*

Preuve du lemme

1. Soit C un code optimal (i.e., minimisant $L(C)$) et soit C' obtenu à partir de C en interchangeant l_j et l_k. Alors

$$\begin{aligned} 0 &\leq L(C') - L(C) \\ &= p_j l_k + p_k l_j - p_j l_j - p_k l_k \\ &= (p_j - p_k)(l_k - l_j). \end{aligned}$$

2. Supposons qu'une branche n'ait pas de sœur. Alors, considérons l'arbre obtenu en supprimant cette branche et en remontant ses éventuels filles d'un niveau. On constate immédiatement que cet arbre correspond à un code de longueur moyenne inférieure à celle du premier code. Contradiction.

3. Soit j et k les indices correspondants à ces deux probabilités, avec $p_j \geq p_k$. On sait par 1. que $l_k = \min_i l_i$. Supposons par l'absurde que $l_j \neq l_k$. On a alors que

$$\forall i \neq k, \quad l_i > l_k.$$

Ainsi la feuille correspondant à k est la seule son niveau de profondeur dans l'arbre représentant le code. Contradiction avec 2. On a donc $l_j = l_k$. Par ailleurs, il est facile de constater que l'on peut échanger les étiquettes des feuilles de même niveau sans changer les propriétés précédemment montrées et donc obtenir que les deux feuilles qui nous intéressent soient sœurs.

Preuve du théorème

On raisonne par récurrence sur $|\mathcal{X}| = n$, en ayant comme hypothèse de récurrence $L(T_n) = L(H_n)$, ce qui équivaut à :

$$\forall T'_n, \qquad L(H_n) \leq L(T'_n)$$

avec T_n un arbre optimal, H_n un code de Huffman et T'_n un arbre représentant un code quelconque adapté à $\mathcal{X}$.
Le résultat est évident pour $n = 2$. Supposons le résultat vrai pour $n \geq 3$ et soit $p_1 \geq p_2 \geq \cdots \geq p_{n+1}$ les probabilités qui nous intéressent.
D'après le lemme, les branches de T_{n+1} associées aux probabilités p_n et p_{n+1} sont sœurs. Définissons alors l'arbre T'_n comme l'arbre obtenu à partir de T_{n+1} en fusionnant les deux branches sœurs correspondant à p_n et p_{n+1} en une branche correspondant à $p_n + p_{n+1}$. On a alors

$$\begin{aligned} L(T_{n+1}) &= L(T'_n) - (l_n - 1)(p_n + p_{n+1}) + l_n p_n + l_n p_{n+1} \\ &= L(T'_n) + p_n + p_{n+1}. \end{aligned} \tag{4}$$

Soit H_n un code de Huffman pour $(p_1, \ldots, p_{n-1}, p_n + p_{n+1})$. Par construction des codes de Huffman, si on remplace la feuille d'étiquette $p_n + p_{n+1}$ par une branche ayant deux filles d'étiquettes p_n et p_{n+1}, on a toujours un arbre de Huffman. On nomme cet arbre H_{n+1}. On a également

$$L(H_{n+1}) = L(H_n) + p_n + p_{n+1}. \tag{5}$$

Comme $L(H_n) \leq L(T'_n)$, par (4) et (5) on déduit

$$L(H_{n+1}) \leq L(T_{n+1}).$$

Or, par définition, $L(T_{n+1}) \leq L(H_{n+1})$, d'où $L(H_{n+1}) = L(T_{n+1})$.