Codage des sources discrètes sans mémoire

Plan du cours

1. Historique : codage de Morse ;
2. Définitions : codage ; efficacité d'un codage ;
3. codes de longueur fixe, codage des chiffres ;
4. condition du préfixe ;
5. théorèmes de Kraft et de Mac Millan
6. Premier théorème de Shannon pour une source discrète sans mémoire.
7. Génération d'une loi de probabilités avec des tirages pile ou face.

1. Historique

L'idée générale : coder par des mots de code courts les lettres les plus fréquentes.

C'est le cas du codage de Morse

A	.-	N	-.	0	-----
B	-...	O	---	1	.----
C	-.-.	P	.--.	2	..---
D	-..	Q	--.-	3	...--
E	.	R	.-.	4	....-
F	..-.	S	...	5	.....
G	--.	T	-	6	-....
H	....	U	..-	7	--...
I	..	V	...-	8	---..
J	.---	W	.--	9	----.
K	-.-	X	-..-	.	.-.-.-
L	.-..	Y	-.--	,	--..--
M	--	Z	--..	?	..--..

En fait codage ternaire (symbole supplémentaire pour séparer les lettres).

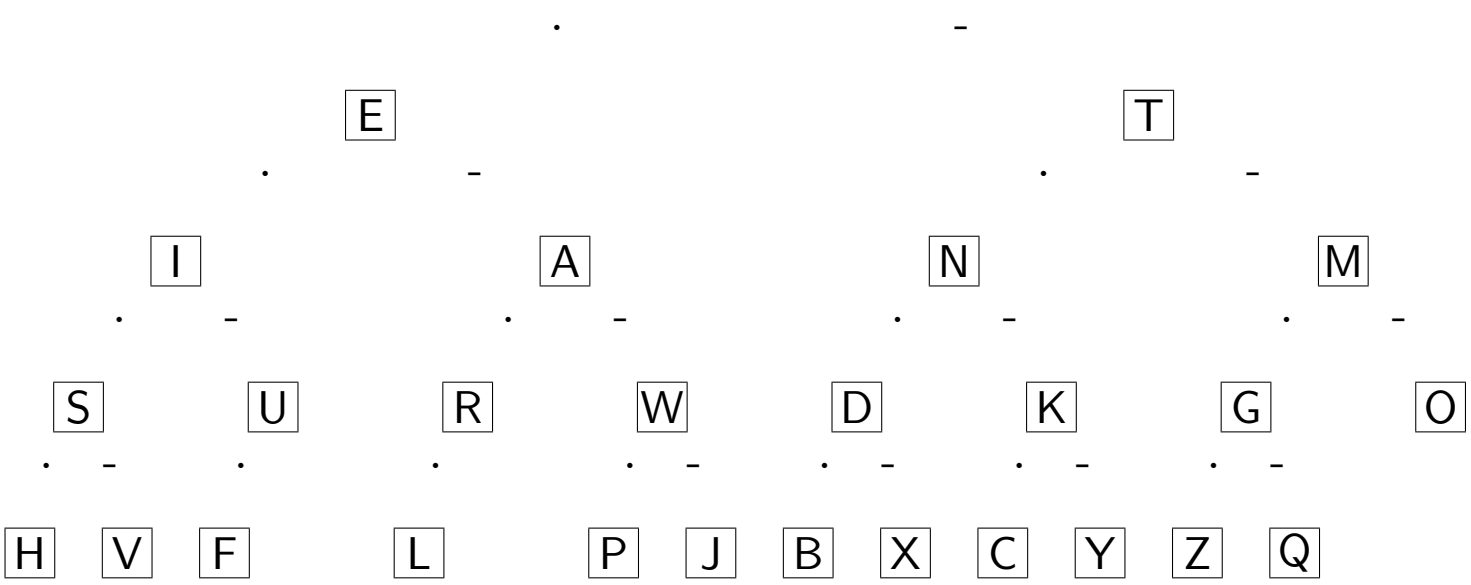
Sinon : impossible de distinguer

- "BAM" $\rightarrow$ "-....---"
- "NIJ" $\rightarrow$ "-....---"

Adapté à un opérateur humain, mais pas aux moyens de communication modernes.

Code Morse

On peut représenter le codage de Morse à l'aide d'un arbre binaire. Chaque nœud, à l'exception de la racine, est le codage d'une lettre.



2. Définitions

Code et Codage

Soit un alphabet (fini) $\mathcal{X}$.

Définition : Un *code* de $\mathcal{X}$ est une application $\varphi : \mathcal{X} \to \{0,1\}^*$ (l'ensemble des mots binaires de longueur arbitraire).

Définition Un *mot de code* est un élément de $\varphi(\mathcal{X})$.

Définition Un *codage* de $\mathcal{X}$ est une application $\psi : \mathcal{X}^* \to \{0,1\}^*$, qui à toute séquence finie de lettres de $\mathcal{X}$ associe une séquence binaire.

À tout code φ de X on peut associer le codage

$$(x_1, x_2, \ldots, x_L) \to (\varphi(x_1) \parallel \varphi(x_2) \parallel \ldots \parallel \varphi(x_L))$$

Codages non ambigu

Définition Un code (resp. codage) est dit *non ambigu* si deux lettres (resp. séquences de lettres) distinctes sont codées par des mots distincts.

Un codage ambigu implique une perte d'information. Codage de Morse : introduction d'un séparateur : trois blancs.

Un code est *à décodage unique* si son codage associé est non ambigu.

	$\phi_1(a_k)$
a_1	0
a_2	010
a_3	01
a_4	10

Comment décoder 010 : a_2, a_1a_4, ou a_3a_1 ?

Source sans mémoire – Efficacité

Une source discrète est une suite $(X_i)_i$ de variables aléatoires prenant leurs valeurs dans un même alphabet fini $\mathcal{X}$.

Définition : Une source $X = (\mathcal{X}, p)$ est *sans mémoire* si les X_i sont i.i.d. et leur loi est donnée par p. Son *entropie* est par définition égale à $H(X) \stackrel{\text{def}}{=} -\sum_{x\in\mathcal{X}} p(x)\log_2 p(x)$.

Définition : La *longueur moyenne* d'un code φ d'une source discrète sans mémoire $X = (\mathcal{X}, p)$ est définie par

$$|\varphi| \stackrel{\text{def}}{=} \sum_{x\in\mathcal{X}} p(x)|\varphi(x)|$$

Définition : L'*efficacité* d'un code φ est définie par $E(\varphi) \stackrel{\text{def}}{=} \frac{H(X)}{|\varphi|}$.

Efficacité d'un codage

Soit $(x_1, \ldots, x_n)$ une séquence finie de lettres de $\mathcal{X}$, nous noterons $p(x_1, \ldots, x_n)$ sa probabilité.

La *longueur moyenne par lettre* du codage ψ sera définie par la limite suivante (si elle existe)

$$\mathcal{L}(\psi) \stackrel{\text{def}}{=} \lim_{n\to\infty} \frac{1}{n} \sum_{x_1,\ldots,x_n} p(x_1, \ldots, x_n)|\psi(x_1, \ldots, x_n)|$$

Définition : Soit X une source discrète sans mémoire et ψ un codage de X dont la longueur moyenne par lettre est définie. L'*efficacité* de ψ est égale à

$$E(\psi) \stackrel{\text{def}}{=} \frac{H(X)}{\mathcal{L}(\psi)}.$$

3. Codes de longueur fixe

Codes de longueur fixe : ce sont des codes dont tous les mots ont la même longueur n.

Proposition 1. *Pour tout code non ambigu de longueur n d'une source X de cardinal K, nous avons*

$$\log_2 K \leq n$$

Preuve

Si $2^n < K$, on essaye de coder sur un ensemble plus petit $\Longrightarrow$ perte d'injectivité. Donc $n \geq \log K$.

L'efficacité d'un tel code est donc limitée par $H(X)/\log_2 K$ (qui vaut 1 si la loi de X est uniforme).

Borne supérieure

Proposition 2. *Pour toute source X de cardinal K, il existe un code non ambigu de longueur n tel que*

$$\log_2 K \leq n < 1 + \log_2 K$$

Preuve

Soit n le plus petit entier tel que $2^n \geq K$, alors à la lettre a_k on associe l'écriture en base 2 de k, sur n bits. On a $2^{n-1} < K$, donc

$$n < 1 + \log_2 K$$

Efficacité des codes de longueur fixe

Corollaire 1. *Il existe un codage non ambigu de X dont l'efficacité est arbitrairement proche de $H(X)/\log_2 K$.*

Preuve On code la source par paquets de taille L (ce qui revient à considérer maintenant des lettres dans l'alphabet $\mathcal{X}^L$), alors il existe un code de longueur n' tel que

$$L \log_2 K \leq n' < L \log_2 K + 1$$

Donc la longueur par lettre est

$$\frac{n'}{L} \to \log_2(K)$$

Et donc

$$E = H/\mathcal{L} \to H/\log_2 K$$

Codage des chiffres

Soit la source $\mathcal{X} = \{0, 1, \ldots, 9\}$ munie de la loi de probabilité uniforme. Le code de longueur fixe d'une telle source a une longueur au moins 4. Par exemple

lettre	mot de code
0	0000
1	0001
2	0010
3	0011
4	0100

lettre	mot de code
5	0101
6	0110
7	0111
8	1000
9	1001

L'efficacité de ce code est égale à $H(X)/4 = (\log_2 10)/4 = 0,83$

En les regroupant 3 par 3, on a un cardinal de 1000, qu'on code sur 10 bits ($2^{10} = 1024$). L'efficacité est de

$$\frac{\log_2(1000)}{10} = 0,9965$$

4. Codes de longueur variable

Rappel Un code est dit *à décodage unique* si son codage associé est injectif.

Autrement dit, une séquence binaire finie donnée correspond au plus à une séquence de lettres de la source.

Condition du préfixe
Aucun mot de code n'est le début d'un autre

Définition Un code est dit *préfixe, ou instantané* s'il vérifie la condition du préfixe. Nous parlerons aussi de code *instantané*.

Proposition 3. *Tout code préfixe est à décodage unique.*

Il existe des codes à décodage unique qui ne sont pas préfixes.

Exemple et contre exemple

$\mathcal{X}$	$\phi_0(a_k)$	$\phi_1(a_k)$	$\phi_2(a_k)$	$\phi_3(a_k)$	$\phi_4(a_k)$
a_1	0	0	1	0	10
a_2	11	010	10	10	00
a_3	11	01	100	110	11
a_4	10	10	1000	111	110

ϕ_0 : ambigu.

ϕ_1 : non ambigu, pas à décodage unique. 010 : a_2 ou a_1a_4 ou a_3a_1.

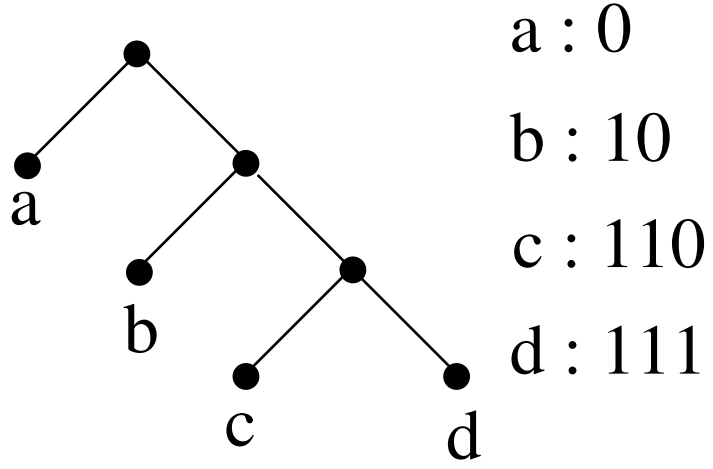
ϕ_2 est à décodage unique (facile) mais n'est pas préfixe.

ϕ_3 satisfait la condition du préfixe.

ϕ_4 non préfixe, à décodage unique.

Arbre associé à un code préfixe

Pour tout code préfixe, il existe un arbre dont les mots de code sont les feuilles (condition nécessaire et suffisante).

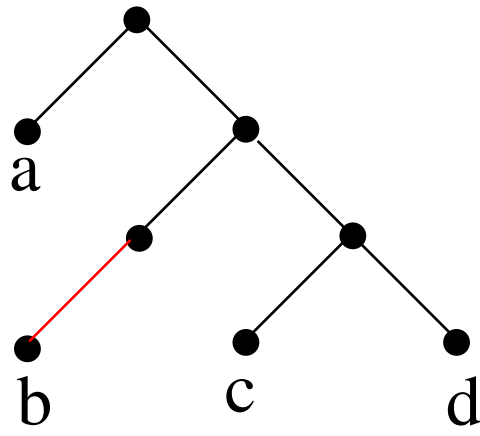


a : 0

b : 10

c : 110

d : 111



a : 0

b : 100

c : 110

d : 111

On dira que le code est irréductible si les nœuds de son arbre ont soit 0 soit 2 fils.

On préférera les codes irréductibles, car ils sont plus courts.

5. Inegalité de Kraft – Théorème de Mac Millan

Théorème 1. [(Kraft)] *Il existe un code préfixe dont les K mots ont pour longueur $n_1, n_2, \ldots, n_K$ si et seulement si*

$$\sum_{k=1}^{K} \frac{1}{2^{n_k}} \leq 1.$$

Théorème 2. [(Mac Millan)] *Il existe un code à décodage unique dont les K mots ont pour longueur $n_1, n_2, \ldots, n_K$ si et seulement si*

$$\sum_{k=1}^{K} \frac{1}{2^{n_k}} \leq 1.$$

Preuve du théorème de Kraft

$\Longrightarrow$ (S'il existe un code préfixe, alors $\sum_{k=1}^{K} \frac{1}{2^{n_k}} \leq 1$)

Soit $N = \max n_l$.

On considère l'arbre complet binaire de profondeur N, en partant de la racine, quand on tombe sur un mot de code e_k (qui n'a donc pas de descendant), on enlève le sous-arbre fils.

Si n_k est la profondeur de e_k, on enlève donc 2^{N-n_k} feuilles de l'arbre complet. Le nombre total de feuilles est 2^N.

On a donc

$$\sum_k 2^{N-n_k} \leq 2^N$$

qui donne bien la relation désirée : $\sum 2^{-n_k} \leq 1$.

Preuve du théorème de Kraft

$\Longleftarrow$ (si $\sum_{k=1}^{K} \frac{1}{2^{n_k}} \leq 1$ alors il existe un code préfixe).

Trions les n_k par ordre croissant, soit $N = \max n_k$. On considère encore l'arbre binaire complet de profondeur N.

Algorithme : *à l'étape k, on prend le premier (dans l'ordre lexicographique) nœud survivant à profondeur n_k, et on enlève tout le sous-arbre en dessous.*

A l'étape k on a supprimé au plus $\sum_{1 \leq i < k} 2^{n_k - n_i}$ noeuds à la profondeur n_k. Or

$$\sum_{1 \leq i < k} 2^{n_k - n_i} = 2^{n_k} \sum_{1 \leq i < k} 2^{-n_i} < 2^{n_k}.$$

Par conséquent, on peut toujours trouver un noeud survivant à la profondeur n_k pour la k-ème étape.

Théorème de Mac Millan

Théorème 3. *(Mac Millan) S'il existe un code à décodage unique alors* $\sum_{k=1}^{K} \frac{1}{2^{n_k}} \leq 1$.

Soit ϕ un code à décodage unique, et Φ son codage associé. On a $\Phi(x_1, \ldots, x_L) = \phi(x_1)||\ldots||\phi(x_L)$. Soit $m = \max_{x \in \mathcal{X}} l(x)$.

$$
\begin{aligned}
(\sum_{x} 2^{-|\phi(x)|})^L &= \sum_{x_1,\ldots,x_L} 2^{-|\phi(x_1)|} \ldots 2^{-|\phi(x_L)|} \\
&= \sum_{x_1,\ldots,x_L} 2^{-|\Phi(x_1,\ldots.x_L)|} \\
&= \sum_{i=1}^{mL} a_L(i) 2^{-i}
\end{aligned}
$$

avec $a_L(i) = |\{(x_1, \ldots, x_l); |\Phi(x_1, \ldots, x_L)| = i\}|$.

$$\left(\sum_x 2^{-|\phi(x)|}\right)^L = \sum_{i=1}^{mL} a_L(i) 2^{-i}$$

On a $a_L(i) \leq 2^i$, sinon deux mots auraient le même codage, ce qui est impossible car le code est à décodage unique.

On a donc

$$\left(\sum_x 2^{-|\phi(x)|}\right)^L \leq mL$$

Soit

$$\left(\sum_x 2^{-|\phi(x)|}\right) \leq (mL)^{1/L} \to 1, \text{ quand } L \to \infty.$$

$\Rightarrow$ Les codes à décodage unique généraux ne sont pas meilleurs que les codes préfixes.

6. Premier théorème de Shannon (pour une source discrète sans mémoire)

Théorème 4. *1. Pour toute source d'entropie H codée au moyen d'un code à décodage unique de longueur moyenne $\bar{n}$, on a $\bar{n} \geq H$.*

2. Pour toute source d'entropie H, il existe un code préfixe de longueur moyenne $\bar{n}$ tel que $H \leq \bar{n} < H + 1$.

Preuve du 1.

Soit $X = (\mathcal{X}, p)$ une source discrète sans mémoire d'entropie H. Soit φ un code à décodage unique pour X. Montrons que $|\varphi| - H \geq 0$. Soit $S \overset{\text{def}}{=} \sum_{x \in \mathcal{X}} 2^{-|\varphi(x)|}$ et q la distribution de probabilité sur $\mathcal{X}$ donnée par $q(x) \overset{\text{def}}{=} \frac{2^{-|\varphi(x)|}}{S}$.

$$
\begin{aligned}
|\varphi| - H &= \sum_{x \in \mathcal{X}} p(x)|\varphi(x)| + \sum_{x \in \mathcal{X}} p(x) \log_2 p(x) \\
&= \sum_{x \in \mathcal{X}} p(x) \left(\log_2 p(x) + \log_2 2^{|\varphi(x)|} \right) \\
&= \sum_{x \in \mathcal{X}} p(x) \log_2 \frac{p(x)}{2^{-|\varphi(x)|}} \\
&= D(p||q) + \log_2 \frac{1}{S} \\
&\geq 0 \quad \text{(positivité de } D \text{ et } S \leq 1 \text{)}
\end{aligned}
$$

Preuve du 2.

Notons $\ell(x)$ l'unique entier vérifiant $2^{-\ell(x)} \leq p(x) < 2^{-\ell(x)+1}$. On a

$$\sum_{x\in\mathcal{X}} 2^{-\ell(x)} \leq \sum_{x\in\mathcal{X}} p(x) = 1,$$

Kraft $\Longrightarrow$ il existe un code préfixe φ avec $|\varphi(x)| = \ell(x), x \in \mathcal{X}$.

Le logarithme de l'inégalité de droite donne $\log_2 p(x) < -\ell(x) + 1$, soit encore

$$|\varphi(x)| = \ell(x) < \log_2 \frac{1}{p(x)} + 1.$$

En passant à la moyenne, nous obtenons

$$|\varphi| = \sum_x p(x)|\varphi(x)| < \sum_x p(x) \left(\log_2 \frac{1}{p(x)} + 1 \right) = H + 1.$$

Cas dyadique

Distribution dyadique : les probas vérifient : $p_i = 2^{-l_i}$.

Proposition 4. *Il existe un code tel que $|\phi| = H$ si et seulement si la distribution est dyadique.*

Il suffit pour s'en rendre compte de reprendre la démonstration du point 1. précédent.

Premier théorème de Shannon

Théorème *(Shannon) Pour toute source discrète X sans mémoire, il existe un codage non ambigu dont l'efficacité est arbitrairement proche de 1.*

Preuve *On considère ϕ_i le code de la source X^i obtenu en regroupant les symboles de X par paquet de taille i. On a $H(X^i) = iH$ et donc :*

$$iH \leq |\phi_i| < iH + 1.$$

On fixe une longueur de paquet l, et on code le mot $x_1, \ldots, x_m, x_r$, de longueur $L = ml + r$, comme suit

$$\phi_l(x_1)|| \ldots ||\phi_l(x_m)||\phi_r(x_r)$$

Preuve (suite)

$L = ml + r$

$$\phi_l(x_1)||\ldots||\phi_l(x_m)||\phi_r(x_r)$$

La longueur moyenne est

$$\begin{aligned} m|\phi_l| + |\phi_r| &\leq m(Hl+1) + rH + 1 \\ &= (ml+r)H + m + 1 = LH + m + 1 \end{aligned}$$

La longueur moyenne par lettre est donc

$$\mathcal{L}(\Phi) = \frac{m|\phi_l| + |\phi_r|}{L} \leq H + \frac{m+1}{ml+r} \leq H + \frac{1}{l} + \frac{1}{ml}$$

Par conséquent

$$\overline{\lim_{L\to\infty}}\, \mathcal{L}(\Phi) \leq H + \frac{1}{l}$$

La borne sup tend vers H, quand $l \to \infty$.

7. Génération d'une distribution ou combien d'aléa pour engendrer une distribution ?

On a

a_1	*1/2*
a_2	*1/4*
a_3	*1/4*

et une pièce $p(Z=0) = p(Z=1) = 1/2$.

Deux manières de procéder

1. *on lance deux fois la pièce :*
   - *si on obtient* 00 *ou* 11 *on donne* a_1,
   - *pour* 01 *on donne* a_2, *et pour* 10 *on donne* a_3

   nombre moyen de lancers : 2
2. *on lance une fois la pièce :*
   - *si obtient zéro on donne* a_1,
   - *sinon, on relance, et si on a* 01 *on donne* a_2, *sinon* a_3.

   nombre moyen de lancers : $1/2 + 1/4 \cdot 2 + 1/4 \cdot 2 = 1,5$.

Cette dernière procédure est plus efficace. on retrouve l'entropie $H = 1/2\log(2) + 1/4\log(4) + 1/4\log(4) = 3/2$.

Le problème

On a $X = (\mathcal{X}, p(x))$, et une séquence de v.a. Z_i, $p(Z_i = 0) = p(Z_i = 1) = 1/2$.

On veut obtenir les $x \in \mathcal{X}$ avec la probabilité $p(x)$ en utilisant les Z_i.

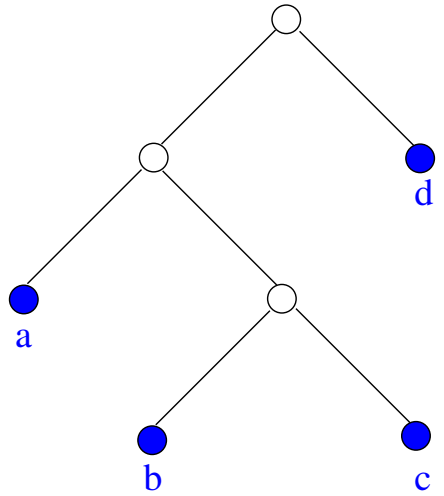
Un tirage $z_1, \ldots, z_m, \ldots$ va correspondre à un parcours dans un arbre binaire.

Chaque feuille a aussi un label $x \in \mathcal{X}$.

Quand on tombe sur une feuille, on émet la lettre correspondante.

Le problème est de construire l'arbre optimal, tel que le parcours moyen soit le plus court, et tel que les probabilités obtenues soient justes.

Un exemple



Cet arbre engendre la source $\mathcal{X} = \{a, b, c, d\}$ *avec les probabilités :*

x	*p(x)*
a	*1/4*
b	*1/8*
c	*1/8*
d	*1/2*

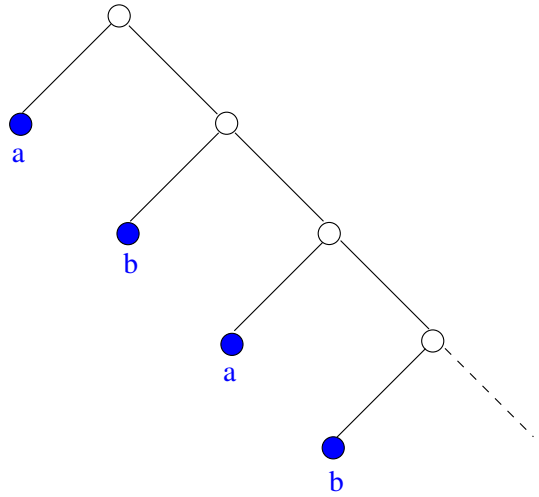
Par algorithme de génération de la source on entend un tel arbre binaire.

Le nombre moyen de lancer de pièces est

$$1/2 \cdot 1 + 1/4 \cdot 2 + 1/8 \cdot 3 + 1/8 \cdot 3 = 1.75 = H(X)$$

Arbre infini

Pour la source $\mathcal{X} = \{a, b\}$, *avec* $p(a) = 2/3$ *et* $p(b) = 1/3$, *on a un arbre infini :*



On vérifie

$$\begin{aligned} p(a) &= \frac{1}{2} + \frac{1}{8} + \cdots + \frac{1}{2^{2i+1}} + \cdots = \frac{2}{3} \\ p(b) &= \frac{1}{4} + \frac{1}{16} + \cdots + \frac{1}{2^{2i}} + \cdots = \frac{1}{3} \end{aligned}$$

Nombre moyen de lancers : $\sum \frac{i}{2^i} = 2$. *Or* $H(X) = h_2(1/3) = 0.918$.

Hauteur moyenne

***Lemme* 1.** *Soit un arbre binaire, tel qu'il existe une loi sur les feuilles vérifiant :*

chaque feuille y à profondeur $k(y)$ a pour probabilité $2^{-k(y)}$; et soit $Y = (\mathcal{Y}, p)$ l'espace des feuilles munies de cette probabilité.

Alors le nombre de moyens de tirages de pièces pour simuler cette distribution $\bar{T}$ est égale à $H(Y)$.

Preuve

$$\bar{T} = \sum_{y \in \mathcal{Y}} k(y) 2^{-k(y)}$$

D'autre part

$$H(Y) = -\sum_{y \in \mathcal{Y}} \frac{1}{2^{k(y)}} \log_2 \frac{1}{2^{k(y)}} = \sum_{y \in \mathcal{Y}} k(y) 2^{-k(y)}$$

Première inégalité

***Proposition* 5.** *Pour tout algorithme engendrant X, le nombre moyen $\bar{T}$ de lancer de pièces est supérieur ou égal à $H(X)$.*

Preuve

Un algorithme engendrant $\mathcal{X}$ est représenté par un arbre binaire : à chaque feuille y on associe un symbole x.

Soit Y la variable aléatoire des feuilles de cet arbre, avec $p(y) = 2^{-k(y)}$.

On a $\bar{T} = H(Y)$

D'autre part X est une fonction de Y : chaque feuille détermine une lettre de $\mathcal{X}$. On a donc (l'application de fonction réduit l'entropie)

$$H(X) \leq H(Y) = \bar{T}.$$

Cas dyadique

Distribution dyadique : les probas vérifient : $p_i = 2^{-l_i}$.

Dans ce cas, il existe un codage prefixe ϕ *tel que* $l(\phi(x)) = -\log_2(p(x))$, *et* $|\phi| = H$. *(prochain cours).*

***Théorème* 5.** *Soit une v.a. X de distribution dyadique. Alors il existe un algorithme de lancer de pièces tel que* $\bar{T} = H(X)$.

Rappel *Il existe un codage préfixe tel que* $|\varphi(x)| = l(x) = -\log_2 p(x)$.

La profondeur de la feuille x *est exactement* $-\log_2(p(x))$ *(pas d'arrondi), donc la profondeur moyenne est*

$$\sum_x p(x)\,(-\log_2(p(x))) = H(X)$$

Cas général

Dans le cas non dyadique :

Théorème 6. *Le nombre moyen $\bar{T}$ de lancer de pièces d'un algorithme optimal de génération d'une variable aléatoire X vérifie*

$$H(X) \leq \bar{T} < H(X) + 2.$$