Théorie de l'information

Pascal SZACHERSKI

23 juin 2008

TABLE DES MATIÈRES

Information ? S'il n'y a pas beaucoup d'information, alors cela va s'écrire rapidement (numériquement : avec peu de bits). Plus l'événement est incertain, plus il y a d'information.

CHAPITRE 1

RAPPELS DE PROBABILITÉ

Définition [1.1] *Un **espace de probabilité** (ou espace probabilisé) est un couple (Ω, P) tel que*
- *Ω est un ensemble fini,*
- *$P : \mathscr{P}(\Omega) \to [0,1]$, $A \mapsto P(A)$.*

Propriétés [1.2]
- $A \cap B = \emptyset \Rightarrow P(A \cup B) = P(A) + P(B)$.
- $P(\Omega) = 1$.
- $P(A|B) = \frac{P(A \cap B)}{P(B)}$.
- $P(A, B) = P(B) \cdot P(A|B)$.

Définition [1.3] *Une **variable aléatoire** X est telle que $X : \Omega \to \mathbb{R}, \mathbb{N}, \mathbb{Z}, \mathbb{N} \times \mathbb{N}$.*
Si X et Y sont deux variables aléatoires alors (X, Y) est une variable aléatoire.

Souvent, on n'a pas accès à la fonction X mais à sa valeur.

Définition [1.4] *La **loi** de X (ou distribution de probabilié) est une donnée de $P(X = x) = P(X^{-1}(x))$ pour toutes les valeurs prises par X.*
*L'**espérance** de Xest donnée par*

$$\begin{aligned} E[X] &= \sum_{\omega \in \Omega} P(\omega) X(\omega) \\ &= \sum_{x} x P(X = x). \end{aligned}$$

Propriétés [1.5] $E[X + Y] = E[X] + E[Y]$.

Définition [1.6] *On a une notion d'**indépendance** :*
- *indépendance d'événements : $A, B \subseteq \Omega$, $P(A, B) = P(A)P(B)$. (Cela inclut : $P(A|B) = P(A)$.)*
- *indépendance de variables aléatoires : X et Y indépendats. $\iff \forall x, y\ P(X = x, Y = y) = P(X = x) \cdot P(Y = y)$.*

*Une grandeur associée aux variables aléatoires est la **variance** :*

$$\begin{aligned} \mathrm{Var}(X) &= E[X^2] - E[X]^2 \\ &= E[(X - E[X])^2] \end{aligned}$$

permet de faire des estimations sur les probabilités.

Théorème [1.7] (de Tchebichev)

$$P(|X - E[X]| > a) \ < \ \frac{\mathrm{Var}(X)}{a^2} = \left(\frac{\sigma(X)}{a}\right)^2.$$

Exemple [1.8] (Pile ou face) $X_1 = \begin{cases} 0 \mapsto 1-p \\ 1 \mapsto p \end{cases}$ *. C'est une variable aléatoire de Bernoulli. On fait plusieurs lancés.* $X_1, X_2, \ldots, X_n$ *indépendants et de même loi.*

Soit $X = \sum_{i=1}^{n} X_i$.

Définition [1.9] *L'**écart type** est* $\sigma(X) = \sqrt{\mathrm{Var}(X)}$. *(C'est l'écart entre* X *et sa moyenne.)*

Théorème [1.10] *Si* X *et* Y *sont indépendants, alors*

$$\mathrm{Var}(X+Y) = \mathrm{Var}(X) + \mathrm{Var}(Y).$$

On a $\mathrm{Var}(X) = n \cdot \mathrm{Var}(X_1)$.

$$\begin{aligned} E[X_1] &= 1 \times \textit{probabilité d'avoir 1} + 0 \times \textit{probabilité d'avoir 0} \\ &= 1 \cdot p \\ &= p. \end{aligned}$$

D'où $\mathrm{Var}(X_1) = p - p^2$. *On a alors* $\mathrm{Var}(x) = n(p-p^2)$, *puis* $\sigma(X) = \sqrt{np(1-p)}$.

La loi des grands nombres *nous dit que, pour* $X = \sum X_i$, *on a*

$$\frac{X - E[X]}{n} \longrightarrow 0.$$

Exemple [1.11] *Pile ou face avec* $p = \frac{1}{4}$. *(Une chance sur quatre d'avoir face.)*

On considère la variable aléatoire

$$X = \textit{gain} \cdot \begin{cases} 2 & \frac{1}{4} \\ -1 & \frac{3}{4} \end{cases}.$$

On a $2 \in$ *si face,* $-1 \in$ *si pile. D'où*

$$\begin{aligned} E[X] &= 2 \cdot \frac{1}{4} - 1 \cdot \frac{3}{4} \\ &= -\frac{1}{4} \\ &< 0. \end{aligned}$$

$\Rightarrow$ *On ne joue pas !!*

Exemple [1.12] (Paradoxe de Saint-Petersbourg) *On regarde quand est-ce que l'on tombe sur face :*

$$\begin{array}{lcl} F & \Rightarrow & 1 \in \\ PF & \Rightarrow & 2 \in \\ PPF & \Rightarrow & 4 \in \\ & \vdots & \\ \underbrace{P \ldots PF}_{i \text{ fois}} & \Rightarrow & 2^i \in \end{array}$$

$$\begin{aligned} E[X] &= 1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{4} + \ldots + 2^i \cdot \frac{1}{2^{i+1}} + \ldots \\ &= \frac{1}{2} + \frac{1}{2} + \ldots + \frac{1}{2} + \ldots \\ &= +\infty \end{aligned}$$

Remarque *La loi des grands nombres ne s'applique que pour des espérances finies.*

***Solution du paradoxe** : Si on joue* n *fois, il faut jouer* $n \log n$.

CHAPITRE 2

INCERTITUDE, INFORMATION

Soit $X_i = \begin{cases} 0 & 1-p \\ 1 & p \end{cases}$, $i = 1, \dots, n$.

On veut mesurer la connaissance des ces X_i. Faut-il beaucoup de bits pour écrire ce vecteur ?

Supposons que p soit très petit, on va avoir des plages de 0 et peu de 1. On peut alors donner les emplacements (coordonnées) des 1 ; Par exemple, i emplacement du premier 1.

Si $p = \frac{1}{4}$, il y a $\frac{n}{4}$ 1.

I Ecriture de $(X_1, \dots, X_n)$

(1) écrire $k = \#\{i \mid X_i = 1\}$.
Par exemple, pour écrire 1000, on utilise 10 bits ; pour un million 20 bits, pour un milliard 30 bits...
$\Rightarrow$ Pour n, on utilise un coût de $\sim \log_2 n$.

(2) définir un ordre parmi tous les n-uples de poids k.
Ecrire le numéro du n-uple de poids k dans la liste.
Le nombre de n-uples de poids $k = \binom{n}{k}$.
Il faut estimer le nombre de bits à écrire ! :
Pour écrire $\binom{n}{k}$, on utilise $\log_2 \binom{n}{k}$. On a

$$\binom{n}{k} \sim \binom{n}{pn} = \frac{n!}{(pn)!(n-pn)!}.$$

Formule de Sterling :

$$n! \sim \sqrt{2\pi n} \left(\frac{n}{\mathbf{e}}\right)^n$$

d'où

$$\begin{aligned} \binom{n}{no} &= \frac{n!}{(pn)!(n-pn)!} \\ &\sim \frac{\cancel{\frac{n^n}{e^n}}}{\cancel{\frac{(pn)^{pn}}{e^{pn}}} \cancel{\frac{(n(1-p))^{n(1-p)}}{e^{n(1-p)}}}} \\ &\sim \left(\frac{1}{p^p(1-p)^{-p}}\right)^n \\ &= 2^{nh(p)} \end{aligned}$$

où $h(p) = p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}$.

Donc, approximativement : $nh(p)$.

$h(p)$ = nombre de bits pour écrire X_1 = entropie de X_1 **ou** information moyenne associée à X_1.

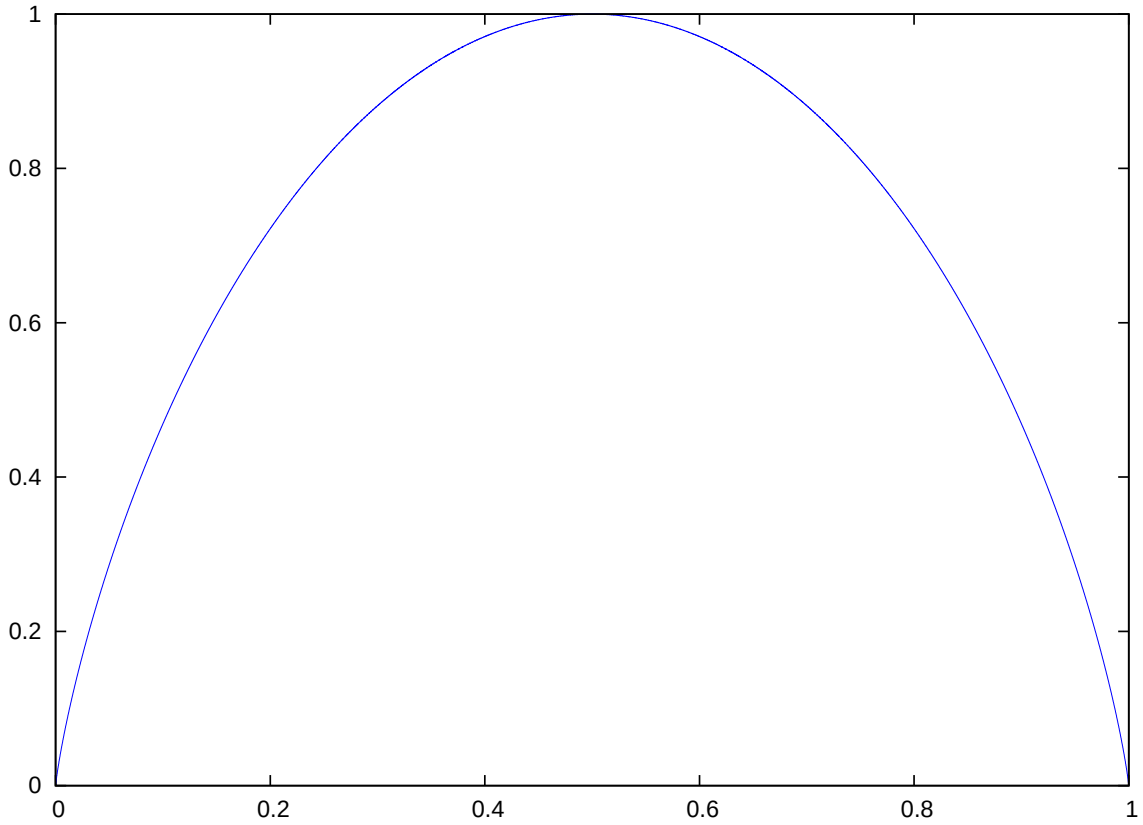


FIG. 2.1 – Représentation graphique de $h(p)$. On voit bien que $h(\frac{1}{2}) = 1 = \max_{x \in [0,1]} h(x)$ et que h est symétrique.

Supposons maintenant que X prend trois valeurs avec ses probabilités correspondantes :

$$\begin{array}{ccc} 1 & 2 & 3 \\ p_1 & p_2 & p_3 \\ \frac{8}{10} & \frac{1}{10} & \frac{1}{10} \\ \frac{9}{20} & \frac{9}{20} & \frac{2}{20} \\ \frac{2}{6} & \frac{3}{6} & \frac{1}{6} \end{array}$$

On se pose la question suivante : Quelle est la loi la plus aléatoire ? Quand on n'a que deux valeurs, celle qui approche le plus la loi uniforme est plus aléatoire, mais ici ce n'est pas aussi facile.

Imaginons n tirages indépendants de X.

#1 $\longrightarrow p_1 n$

#2 $\longrightarrow p_2 n$

#3 $\longrightarrow p_3 n$

Pour décrire X, on introduit une variable aléatoire auxiliaire Y :

$$Y = \begin{cases} 0 & \text{si } X = 1 \qquad p_1 \\ 1 & \text{si } X = 2,3 \qquad 1 - p_1 = p_2 + p_3 \end{cases}$$

Pour décrire X, on commence par décrire Y, puis X sachant Y.

(1) #1 : coût de symboles : $\approx \log n$.

$\boxed{nh(1-p_1)}$ nombre de symboles pour la réalisation de Y pour dire que $X = 1$ est sorti.

(2) #2 et #3 :

$\boxed{(1-p_1)nh\left(\frac{p_2}{1-p_1}\right)}$ nombre de symboles $X=2$ ou $X=3$ sachant $Y=1$.

Alors, on a

$$nh(1-p_1)+(1-p_1)nh\left(\tfrac{p_2}{1-p_1}\right) = n\left(p_1\log\tfrac{1}{p_1}+(1-p_1)\log\tfrac{1}{1-p_1}\right)+n\left(p_2\log\tfrac{1-p_1}{p_2}+p_3\log\tfrac{1-p_1}{p_3}\right)$$
$$= n\left(p_1\log\tfrac{1}{p_1}+p_2\log\tfrac{1}{p_2}+p_3\log\tfrac{1}{p_3}\right),$$

ce qui nous donne le nombre de symboles binaires nécessaires pour écrire n réalisations de X.

Résumé [2.1] *Soit X une variable aléatoire qui suit une loi et $p_1,\dots,p_k$ tels que $\sum_i p_i = 1$. Nous définissons*

$$I := \sum_{i=1}^{k} p_i \log\frac{1}{p_i}.$$

L'ensemble des réalisations "typiques" a environ 2^{nI} éléments. (Ceci dit que la probabilité de tomber sur un événement rare est minime.)

II Définition de l'entropie et des quantités associées

Définition [2.2] *L'**information** associée à un événement A est une fonction de la probabilité que A ait lieu :*

$$i(A) = f(P(A)).$$

Si A et B sont deux événements indépendants, on a envie de dire que

$$i(A\cap B) = i(A)+i(B) \Longleftrightarrow f(p_1p_2) = f(P(A)P(B)) = f(P(A))+f(P(B)) = f(p_1)+f(p_2).$$

Or, la seule fonction (à un facteur près) qui vérifie cela est la fonction logarithmique :

$$f(p) = -\log p = \log\frac{1}{p}.$$

*L'**information moyenne** associée à une variable aléatoire X avec une loi $p_1,\dots,p_k$ est*

$$H(X) := \sum_{i=1}^{k} p_i\log\frac{1}{p_i} = -\sum_{i=1}^{k} p_i\log p_i$$

*et s'appelle **entropie** de X.*

Le choix de la base du logarithme dépend de l'alphabet choisi : $\log_2$ pour un alphabet binaire, $\log_{10}$ pour un alphabet décimal etc.

*L'entropie est mesurée en **shannon** (ou bits d'informations) si le logarithme est de base 2.*

Proposition [2.3] *$H(X)\geq 0\ \forall X$ et $H(X) =: H(p_1,\dots,p_k)$ est maximal quand $p_i = \frac{1}{k}$, $\forall i = 1,\dots,k$ (donc quand X suit la loi uniforme).*

Preuve On utilise l'inégalité suivante :

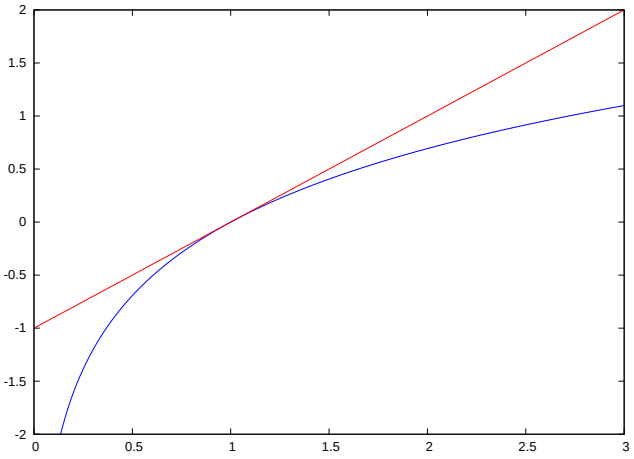
$$\log x \leq x-1 \quad \forall x>0$$

Soient p, q deux probabilités :

$$\log\frac{q}{p} \leq \frac{q}{p}-1$$
$$p\log\frac{q}{p} \leq q-p$$

On écrit $p=(p_1,\dots,p_k), q=(q_1,\dots,q_k)$. Alors :

$$p_i\log\frac{q_i}{p_i} \leq q_i-p_i$$
$$\sum p_i\log\frac{q_i}{p_i} \leq \underbrace{\sum q_i}_{=1} - \underbrace{\sum p_i}_{=1}$$
$$\sum p_i\log\frac{q_i}{p_i} \leq 0 \quad \forall p_i, q_i$$



En particulier, $q_i = \frac{1}{k}\ \forall i$. On obtient :

$$\sum_i p_i \log \tfrac{1/k}{p_i} \leq 0$$
$$\sum_i \left(p_i \log \tfrac{1}{p_i} + p_i \log \tfrac{1}{k} \right) \leq 0$$
$$\left(\sum_i p_i \log \tfrac{1}{p_i} \right) + \left(\log \tfrac{1}{k} \right) \leq 0$$
$$H(p_1, \ldots, p_k) \leq \log k = H(\tfrac{1}{k}, \ldots, \tfrac{1}{k}) = \sum \tfrac{1}{k} \log k \quad \forall p_i,\ i = 1, \ldots, k.$$

Exemple [2.4] *Sur une table se trouvent dix cartes numérotées, les faces cachées, et comparables deux à deux. On joue à classer dans l'ordre ces cartes. Pour cela, on a droit à des questions, demandant si la valeur d'une carte est plus grande ou plus petite que celle d'une autre. Chaque question coûte* 1 €*! Quand on a classé les cartes dans l'ordre, on gagne* 15 €*.*

Est-ce qu'on y joue ? !

Il s'agit en effet d'un problème d'information. On cherche à découvrir quelle est la permutation des cartes. Il y en a $10! = 3\,628\,800$*. Donc, la quantité d'information vaut* $\texttt{ceil}(\log_2 10!) = \texttt{ceil}(21,8) = 22$*. Une réponse rapportant au mieux* 1 *shannon (*1 *bit d'information) et supposons une loi uniforme* $P(oui) = P(non) = \frac{1}{2}$ *(probabilités totales), il nous faudrait donc en moyenne 22 questions bien posées et rapportant toujours une pièce du puzzle pour pouvoir classer !*

Définition [2.5] ("Distance" de Kullbach) *La* ***distance de Kullbach*** *entre lois* $p = (p_1, \ldots, p_k)$, $q = (q_1, \ldots, q_k)$ *se calcule*

$$D(p\|q) := \sum_i p_i \log_2 \frac{p_i}{q_i}$$

(attention, ceci n'est pas une distance topologique !).

Lemme [2.6] $\forall p, q,\ D(p\|q) \geq 0$.

Preuve Vu dans la preuve de la proposition précédente.

Définition [2.7] (Entropie conditionnelle) *Soient* X, Y *des variables aléatoires.* $H(X) = \sum_x P(X = x) \log \frac{1}{P(X=x)}$ *et idem pour* Y. $H(X,Y) := \sum_{x,y} P(X = x, Y = y) \log \frac{1}{P(X=x,Y=y)}$.

Alors :

$$\begin{aligned} H(X|Y) &= \sum_y P(Y = y) \sum_x P(X = x|Y = y) \log \tfrac{1}{P(X=x|Y=y)} \\ &= \sum_{x,y} P(X = x, Y = y) \log \tfrac{1}{P(X=x|Y=y)} \\ &= -\sum_{x,y} P(X = x, Y = y) \log P(X = x|Y = y) \end{aligned}$$

Proposition [2.8] $H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$.

Preuve Tenant compte de la symétrie de $H(X,Y)$, on a :

$$\begin{aligned} H(X,Y) &= -\sum_{x,y} P(X = x, Y = y) \log P(X = x, Y = y) \\ &= -\sum_{x,y} P(X = x, Y = y) \log \left(P(X = x) P(Y = y|X = x) \right) \\ &= -\sum_{x,y} P(X = x, Y = y) \left(\log P(X = x) + \log P(Y = y|X = x) \right) \\ &= -\sum_x \log P(X = x) \underbrace{\sum_y P(X = x, Y = y)}_{=P(X=x)} + H(Y|X) \\ &= H(X) + H(Y|X) \end{aligned}$$

ce qui était à démontrer.

Définition [2.9] *L'****information mutuelle*** *est définie par*

$$\begin{aligned} I(X,Y) &:= D\big(P(X,Y)\|p(X)p(Y)\big) \\ &= \sum_{x,y} P(X = x, Y = y) \log \tfrac{P(X=x,Y=y)}{P(X=x)P(Y=y)} \end{aligned}$$

Remarque [2.10] *(a) Si X et Y sont indépendants, alors $I(X,Y) = 0$.*
(b) $I(X,Y) \geq 0$.

Proposition [2.11] *Avec les notations précédentes, on a*

$$\begin{aligned} I(X,Y) &= H(X) + H(Y) - H(X,Y) \\ &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X). \end{aligned}$$

De plus, on a toujours

$$H(Y|X) \leq H(Y).$$

Applications à la cryptographie

Soit M un message clair, K la clé qui permet de lire M à partir d'un message codé C. En formules : $C = f_K(M)$, $M = f_K^{-1}(C)$.
L'incertitude sur M connaissant le message codé s'écrit $H(M|C)$.

Définition [2.12] *Un système (M,K,C) est appelé **parfait**. $\iff H(M|C) = H(M)$.*

Un système où on ne gagne pas d'information sur le message clair en connaissant le message codé est donc appelé parfait.
Supposons qu'on a un système (M,K,C) parfait. Alors

$$\begin{aligned} H(M) &= H(M|C) \\ &= H(M,K|C) \\ &= H(K,C) + \underbrace{H(M|K,C)}_{=0} \\ &\leq H(K). \end{aligned}$$

Théorème [2.13] (Théorème de Shannon) *Si le système (M,K,C) est parfait alors*

$$H(M) \leq H(K).$$

Distance d'unicité

Soit (M,K,C) un système de chiffrement, par exemple de substitution. $\Rightarrow \mathcal{M} = \mathcal{C} = \{A, B, \ldots, Z\}$. On dit que M_i, un élément du message chiffré, est un mot.

$$M_1, \ldots, M_n \xrightarrow{K} C_1, \ldots, C_n$$

Définition [2.14] *La distance d'unicité d est le plus petit entier tel que*

$$H(K|C_1, \ldots, C_d) = 0,$$

donc tel que l'incertidude sur la clé de déchiffrement connaissant d mots chiffrés soit nulle.

Estimation de d :

$$\begin{aligned} 0 = H(K|C_1, \ldots, C_d) &= H(K, C_1, \ldots, C_d) - H(C_1, \ldots, C_d) \\ &= H(M_1, \ldots, M_d, K, C_1, \ldots, C_d) - H(C_1, \ldots, C_d) \\ &= H(M_1, \ldots, M_d, K) - H(C_1, \ldots, C_d) \\ &= H(M_1, \ldots, M_d) + M(K) - H(C_1, \ldots, C_d). \end{aligned}$$

On va devoir poser une hypothèse : En générale, l'entropie d'une langue est difficile à estimer, mais tout de même approchable avec une hypothèse raisonnable. Normalement, chaque symbole a une influence plus forte sur ses voisins que sur le symbole 10 places plus loin.

Ici on va dire que $\frac{H(M_1,\ldots,M_i)}{i} \to H$, et on obtient :

$$0 = Hd + H(K) + H(C_1, \ldots, C_d).$$

Dans l'inégalité suivante, on a presque une égalité pour d petit, mais on général on a

$$H(C_1, \ldots, C_d) \leq d \log(\operatorname{card} \mathcal{C})$$

ce que donne directement

$$Hd + H(K) - d \log(\operatorname{card} \mathcal{C}) \leq 0 \Longleftrightarrow d \geq \frac{H(K)}{\log(\operatorname{card} \mathcal{C}) - H}$$

Exemple [2.15] *Alphabet avec 26 lettres. On a alors :* $\log(\operatorname{card} \mathcal{C}) = \log 26$, $H(K) = \log(26!)$, $H \approx 2$ *bits par symbole. (Ceci est une approximation pour la langue anglaise qui reste valable pour le français).*

On calcule donc que $d \approx 30$. *Ça veut dire que, avec un chiffrement de substitution, on a besoin d'au moins trente lettres pour arriver à remonter à la clé sans ambiguïté ; en-dessous de ce seuil il restera de l'ambiguïté sur le message clair.*

Pour aggrandir la sécurité, c'est-à-dire aggrandir d, on choisit un $\mathcal{C}$ plus petit. Comme cela, $\log(\operatorname{card} \mathcal{C}) - H$ devient plus petit et donc d plus grand.

CHAPITRE 3

COMPRESSION, CODAGE DE SOURCE

I Préliminaires ...

Supposons qu'on ait une source $\mathcal{X} = \{1, 2, \dots, m\}$ et une suite $X_1, \dots, n$ indépendante avec la même loi. (Encore une fois on a dû idéaliser puisque dans la réalité ce n'est pas le cas.) Un codage est une application de $\mathcal{X}$ dans $\{0,1\}^*$ l'ensemble des suites composées des 0 et 1 et $\mathcal{C}(\mathcal{X}) =: C$ est appelé un **code**. $C^* := \{c_1 \dots c_k \mid k \in \mathbb{N}, c_i \in C, i = 1, \dots, k\}$ est l'ensemble des mots qu'on peut créer en concatenant C, $\mathcal{C}^* : x_1, \dots, x_k \mapsto \mathcal{C}(x_1) \dots \mathcal{C}(x_k)$.

Définition [3.1] *C est appelé **uniquement déchiffrable**. $\Longleftrightarrow$[$(c_1, \dots, c_k) \neq (c'_1, \dots, c'_k) \Rightarrow c_1 \dots c_k \neq c'_1 \dots c'_k$]. En mots, quand il n'existe pas plusieurs manières de trouver un mot en concatenant les c_i.*

Exemple [3.2] *(a) $C = \{00,\ 01,\ 10,\ 11\}$ est uniquement déchiffrable (car on ne considère que des couples).*

(b) $C = \{0,\ 01\}$ est uniquement déchiffrable (ex. : 0\|0\|01\|01\|0).

(c) $C = \{0,\ 00,\ 001\}$ n'est pas uniquement déchiffrable (ex. : 0\|0\|001 ou 00\|001)

On a pour **but** de maximiser la longueur de $\mathcal{C}^*(x_1, \dots, c_k)$ pour k grand !

Exemple [3.3] *$\mathcal{X} = \{1, 2, 3, 4\}$ avec $P(X = 1) = \frac{1}{2}$, $P(X = 2) = \frac{1}{4}$, $P(X = 3) = P(X = 4) = \frac{1}{8}$.*
Regardons d'abord le code $C = \{00, 01, 10, 11\}$ qui ne prend pas en compte la loi de X. La longueur moyenne $\overline{\ell} = 2$ bits par symbole.
$C' = \{0, 10, 110, 111\}$ est uniquement déchiffrable car aucun mot n'a comme préfixe un autre mot :

$$1 \mapsto 0,\ 2 \mapsto 10,\ 3 \mapsto 110,\ 4 \mapsto 111.$$

La longueur moyenne $\overline{\ell}' = \frac{7}{4}$.

Remarque [3.4] *$H(X) = \frac{7}{4} \overset{ici}{=} \overline{\ell}'$. Nous comprendrons mieux pourquoi grâce au ...*

Théorème [3.5] *La longueur moyenne $\overline{\ell}$ du codage optimal de X vérifie*

$$H(X) \leq \overline{\ell} \leq H(X) + 1.$$

II Types de code

Définition [3.6] *Un code C est dit **préfixe** si* aucun mot $c \in C$ *n'est préfixe de $c' \in C$.*

Il existe deux types de code : les codes préfixes et les codes uniquement déchiffrable.

Proposition [3.7] *Soit C un code préfixe. Alors C est uniquement déchiffrable.*

Théorème [3.8] (Kraft) *Soient $\ell_1, \dots, \ell_n \in \mathbb{N}^*$.*
$\exists$ *C préfixe, $C = \{c_1, \dots, c_n\}$, $\ell(c_i) = \ell_i \Longleftrightarrow \sum_{i=1}^{n} 2^{-\ell_i} \leq 1$.*

Théorème [3.9] (MacMillan) *Si C est uniquement déchiffrable, $C = \{c_1, \dots, c_n\}$, $\ell_i = \ell(c_i)$, alors*

$$\sum_{i=1}^{n} 2^{-\ell_i} \leq 1.$$

Théorème [3.10] *Soit C un code uniquement déchiffrable associé à $\mathcal{X} = \{x_1, \dots, x_n\}$. Alors $H(X) \leq \overline{\ell}$.*
On a :

$$\overline{\ell} = E[\ell(C(X))] = \sum_i p_i \ell_i$$

où $p_i = P(X = x_i)$

Théorème [3.11] *Le code* optimal *vérifie $H(X) \leq \overline{\ell} \leq H(X) + 1$.*

III Recherche du code optimal

Exemple [3.12] *On considère l'ensemble $\mathcal{X} = \{1,\ 2,\ 3,\ 4,\ 5,\ 6\}$ avec les probabilités respectives $p = (0.4;\ 0.04;\ 0.14;\ 0.18;\ 0.18;\ 0.06\}$.*

A chaque étape on choisit les deux éléments avec les probabilités les plus faibles et on les remplace par un seul nouvel élément dont les deux éléments remplacés "hériteront" dans un modèle d'arbre.

Dans notre cas, la première étape est donc de remplacer 2 et 6 avec les probabilités respectives de 0.04 et 0.06. On nomme notre nouvel élément abritrairement 7 qui aura une probabilité de $0.04 + 0.06 = 0.1$.

On trouve alors $\mathcal{X}' = \{1,\ 3,\ 4,\ 5,\ 7\}$ et $p' = (0.4;\ 0.14;\ 0.18;\ 0.18;\ 0.1)$

Dans la prochaine étape, on remplace 7 ($p_7 = 0.1$) et 3 ($p_3 = 0.14$) par 8 ($p_8 = p_3 + p_7 = 0.24$).
$\Rightarrow \mathcal{X}'' = \{1,\ 4,\ 5,\ 8\}$, $p'' = (0.4;\ 0.18;\ 0.18;\ 0.24)$.

Et ainsi de suite :
$\Rightarrow \mathcal{X}''' = \{1,\ 8,\ 9\}$, $p''' = (0.4;\ 0.24;\ 0.36)$.
$\Rightarrow \mathcal{X}^{(4)} = \{1,\ 10\}$, $p^{(4)} = (0.4;\ 0.6)$.

Alors, on a la représentation suivante :

Théorème [3.13] *Le (ou les) code(s) de Huffman est (sont) optimal(s).*

Lemme [3.14] *Il existe un code optimal vérifiant :*

(a) préfixe,

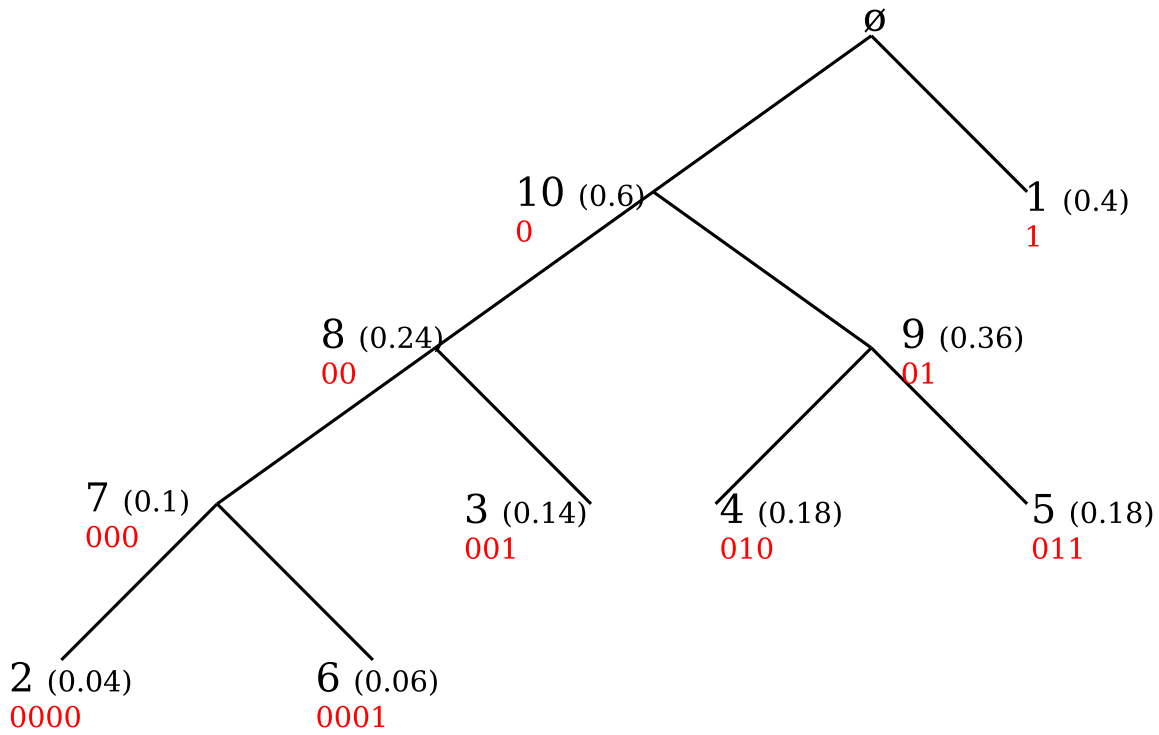
(b) $p_1 \geq \dots \geq p_n \Rightarrow \ell_1 \leq \dots \leq \ell_n$,

(c) les deux probabilités les plus faibles sont associées à des mots de longueur maximale, ne différant que sur le dernier bit.

IV Interpretation de $D(p\|q)$

On considère $p = (p_1, \dots, p_n)$, $q = (q_1, \dots, q_n)$ deux lois de probabilité et un code avec les longueur $(\ell_1, \dots, \ell_n)$. On code $X \in \mathcal{X} = \{x_1, \dots, x_n\}$ avec la loi p. Le choix optimal du code C vérifie $\ell_i \approx \log \frac{1}{p_i}$ et, donc, $\overline{\ell} = H(X) + \varepsilon$, $\varepsilon \geq 0$.

On peut se poser la question ce qui se passera si on se trompe de loi de X. Notons la nouvelle loi q. (Ceci est le cas p. ex. quand X change de nature (changement de type de fichier).)



Ce nouveau code adapté à la loi q vérifie :

$$\begin{aligned}
\ell_i &\approx \log \frac{1}{q_i} \\
\overline{\ell} &= \sum_{i=1}^{n} p_i \log \frac{1}{q_i} \\
&= \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i p_i} \\
&= \sum_{i=1}^{n} p_i \log \frac{1}{p_i} + \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} \\
&= H(p) + D(p\|q).
\end{aligned}$$

$D(p\|q)$ est en quelque sorte le prix à payer pour l'inadaptation de la loi q par rapport à X.

V Pari

Course de chevaux

On assiste à une course de chevaux. Se confrontent les chevaux $1, 2, \ldots, m$ qui ont respectivement une probabilité $p_1, p_2, \ldots, p_m$ de gagner.

Pour une mise de 1 € on reçoit g_i € si c'est le cheval i qui gagne.

Nous avons S € à notre disposition.

Pour gagner le plus possible, on ne maximisera ni p_i (les gains ne sont pas attractifs) ni g_i (trop improbable), mais $p_i g_i$ qui est l'espérance du gain si on mise sur un cheval.

Quand on gagne, on aura beaucoup d'argent. Mais qu'est-ce qui se passe quand on perd tout le temps ? On n'aura plus d'argent ! Il faut revoir la stratégie.

Nous allons miser tout notre argent sur tous les chevaux, à savoir $S\pi_i$ € sur le cheval i. ($\sum \pi_i$ évidemment.)

Soit X le numéro du cheval gagnant. Son gain correspond à $S\pi_X g_X = S\pi(X)g(X)$. Au bout de n courses alors, on aura : $S \prod_{i=1}^{n} \pi(X_i)g(X_i)$.

On réécrit ce terme : $S \prod_{i=1}^{n} \pi(X_i)g(X_i) = S \cdot 2^{\sum \log \pi(X_i)g(X_i)}$. Avec la loi des grands nombres on trouve :

$$\sum \log \pi(X_i)g(X_i) = n\tfrac{1}{n}\sum \log \pi(X_i)g(X_i) \longrightarrow n \underbrace{E[\log \pi(X)g(X)]}_{\sum p_i \log \pi_i g_i =: W(p,g,\pi)} .$$

$\Longrightarrow$ au bout de n courses on aura $S \cdot 2^{nW(p,g,\pi)}$ €.
$W(p,g,\pi)$ est le *taux d'augmentation* que l'on veut optimiser :

$$\begin{aligned}\sum_{j=1}^{m} p_j \log g_j + \sum_{j=1}^{m} p_j \log \pi_j &= \sum_{j=1}^{m} p_j \log g_i + \sum_{j=1}^{m} p_j \log \tfrac{\pi_j p_j}{p_j} \\ &= \sum_{j=1}^{m} p_j \log g_j + \sum_{j=1}^{m} p_j \log p_j + \sum_{j=1}^{m} p_j \log \tfrac{\pi_j}{p_j}\end{aligned}$$

Alors, $W(p,g,\pi) = \sum_{j=1}^{m} p_j \log g_j - H(p) - D(p\|\pi)$. W est maximal quand $\pi = p$.
$\Longrightarrow W_{\max} = \sum_{j=1}^{m} p_j \log g_j - H(p)$.

On se place maintenant dans le cadre (certes irréel) où on connaî les gains. Supposons $\sum_{i=1}^{m} \frac{1}{g_i} = 1$. Idéalement, on aura $p_i = \frac{1}{g_i}$. Dans ce cas, on trouve

$$\begin{aligned}W &= \sum_{j=1}^{m} p_j \log \frac{p_j}{\frac{1}{g_j}p_j} - H(p) - D(p\|\pi) \\ &= H(p) + D(p\|\tfrac{1}{g}) - H(p) - D(p\|\pi) \\ &= D(p\|\tfrac{1}{g}) - D(p\|\pi).\end{aligned}$$

Ça veut dire que le parieur ne gagne que s'il trouve un π qui s'approche plus que l'estimation de l'organisateur de la loi de X.

Supposons qu'on a les résultats d'une autre course Y. Est-ce que cela change notre stratégie ? On cherche alors la loi $\pi(x|y)$. En analogie on a

$$\begin{aligned}W &= \sum_x p(x) \log g(x) + \sum_x p(x) \log \pi(x) \\ W^* &= \sum_{x,y} p(x,y) \log g(x) + \sum_{x,y} p(x,y) \log \pi(x|y) \\ &= \sum_x p(x) \log g(x) + \sum_y p(y) \sum_x p(x|y) \log \pi(x|y).\end{aligned}$$

Comme avant, W^* prend sa valeur maximale pour $\pi(x|y) = p(x|y)$.
Ensemble on aura

$$\begin{aligned}W_{\max} &= \sum_x p(x) \log g(x) - H(X) \\ W^*_{\max} &= \sum_x p(x) \log g(x) - H(X|Y)\end{aligned}$$

$$\begin{aligned}W^*_{\max} - W_{\max} &= H(X) - H(X|Y) \\ &= I(X,Y)\end{aligned}$$

CHAPITRE 4

NOTION DE CANAL DISCRET SANS MÉMOIRE

I Préliminaires ...

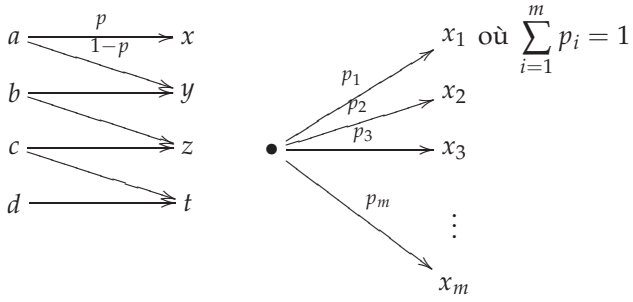


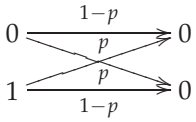
FIG. 4.1 – premier exemple de canal

Soient $X_1, \ldots, X_n$ indépendants et de même loi.

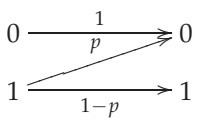
$$X_n \cdots X_1 \longrightarrow \boxed{\text{canal}} \longrightarrow Y_m \cdots Y_1$$

Exemple [4.1]

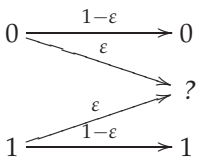
Canal binaire symétrique



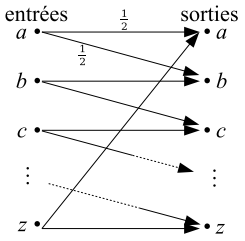
Canal binaire en Z



canal binaire à effacement



Exemple [4.2] *On considère le canal suivant :*



Les transitions ont une probabilité de $\frac{1}{2}$.

Transmission sans ambiguïté de 13 lettres.

En sacrifiant une lettre sur deux, on arrive à retrouver le message envoyé sans ambiguïté. On a ainsi une transmission de $\log_2 13$ *bits/symbole.*

(Astuce : Il faut toujours penser à une réalisation de n *symboles : Dans ce cas, on aurait* 13^n *messages possibles, alors* $\log_2 13^n = n \log_2 13$ *bits d'information.)*

II Capacité d'un canal

Définition [4.3] *La* ***capacité*** C *d'un canal discret sans mémoire est définie par*

$$C := \max_{\text{loi de } X} H(X) - H(X|Y) \Longleftrightarrow \boxed{C = \max_{\text{loi de } X} I(X,Y)}.$$

Exemple [4.4] *On regarde le canal de l'exemple [4.2].*

On rappelle que $H(X) - H(X|Y) = H(Y) - H(Y|X)$.

Or, supposons que le canal est sans ambiguïté, donc qu'on ne considère que 13 lettres d'arrivée. Donc, l'incertidude sur la lettre envoyée en recevant une lettre est

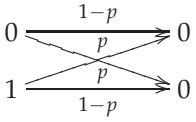
$$H(Y|X) = \tfrac{1}{2}\log_2 2 + \tfrac{1}{2}\log_2 2 = 1.$$

Alors, on a :

$$\begin{aligned} H(X) - H(X|Y) &= H(Y) - H(Y|X) \\ &= H(Y) - 1 \\ &\leq \log_2 26 - 1 \\ &= \log_2 13 \end{aligned}$$

On ne peut transmettre plus de $\log_2 13$ *bits par symbole. On retrouve donc le même résultat qu'en [4.2].*

Exemple [4.5] *On regarde un autre exemple, celui d'un CBS (canal binaire symétrique).*



On sait que $I(X,Y) = H(Y) - H(Y|X)$. *On calcule d'abord l'incertitude sur* Y *sachant* X.

$$\begin{aligned} H(Y|X) &= \sum_{x=0;1} P(X=x) H(Y|X=x) \\ &= P(X=0)h(p) + P(X=1)h(p) \\ &= h(p) \end{aligned}$$

où $h(p) = H(p, 1-p) = p\log_2 \frac{1}{p} + (1-p)\log_2 \frac{1}{1-p}$ *comme défini dans le premier chapitre.*

On a alors que $I(X,Y) = H(Y) - h(p) \leq 1 - h(p)$ *(car* $H(Y)$ *ne peut prendre que des valeurs inférieures ou égales à 1 ici). On a égalité dans le cas où* X *suit une loi uniforme. Donc :*

$$C = 1 - h(p).$$

Un exemple sur le canal à effacement est traité dans le TD6.

CHAPITRE 5

CODAGE DE L'INFORMATION

Appelons code binaire de longueur n $\mathcal{C} \subseteq \{0;1\}^n$ code correctif/de canal ou code compressif/de source. (??????)
$\mathcal{C} \subseteq \mathcal{A}^n$, $\mathcal{A}$ alphabet d'entrée.
$\mathcal{M} = \{\text{messages}\} \leftrightarrow \mathcal{C}$.

Exemple [5.1] (du canal d'effacement) *Prenons n grand : $\mathcal{C} = \{(0,\dots,0);\ (1,\dots,1)\}$.*

Définition [5.2] *On appelle **rendement d'un code** $\mathcal{C}$ la valeur du quotient*

$$R := \frac{\log_2 |\mathcal{C}|}{n}.$$

Supposons que l'on a reçu le mot $\underbrace{0\dots0}_{(1-p)n}\underbrace{\varepsilon\dots\varepsilon}_{pn}$.
La probabilité de commettre une erreur correspond à un effacement total. Donc on trouve

$$\begin{aligned} P(\text{erreur}) &= P(\text{tout effacé}) \\ &= p^n \\ &\xrightarrow{n\to\infty} 0 \end{aligned}$$

Proposition [5.3] (dans le cas du canal d'effacement) *Si $R > C + \varepsilon \Rightarrow$ problème : avec une probabilité proche de 1 le nombre d'effacement $\in\ ](p-\varepsilon)n;\ (p+\varepsilon)n[$. C'est à dire que si le rendement est supérieur à la capacité, quel que soit le code, on ne pourra reconstituer le message.*

Proposition [5.4] *$\forall \varepsilon > 0$, $\exists \mathcal{C} \subseteq \{0,1\}^n$ code de rendement $R \geq C - \varepsilon$ tel que, si on choisit le mot émis $c \in \mathcal{C}$ avec probabilité uniforme, alors le n-uple reçu permet de reconstituer c avec une probabilité qui tend vers 1.*

Pour atteindre la capacié, Shannon introduit trois idées :
– codes de longueur n, $n \to \infty$;
– autoriser une petite erreur de décodage qui tend vers 0 pour n grand ;
– utiliser du codage aléatoire, c'est-à-dire choisir $\mathcal{C}$ au hasard dans $\{0,1\}^n$ avec $R < C$ fixé.

I Théorème de Shannon (dans le cas du CBS)

$\mathcal{C} = \{0,1\}^n$, $c \in \mathcal{C}$ émis, $y \in \{0,1\}^n$ reçu. On suppose que

$$y = c +_{\text{bin}} e,$$

$e \in \{0,1\}$ *vecteur erreur*.
Faire du *décodage* veut dire trouver c à partir du y.

Définition [5.5] (distance de Hamming) $x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n)$.

$$\mathrm{d}(x,y) = |\{i \mid x_i \neq y_i,\ i = 1, \ldots, n\}|.$$

Proposition [5.6] *Le décodeur optimum (à maximum de vraisemblance) choisit le (un) mot de code le plus proche de y. Sous l'hypothèse, le mot émis c est choisi uniformement dans $\mathcal{C}$ avec une probabilité inférieur à $\frac{1}{2}$.*

Remarque et définition [5.7] *La distance de Hamming (contrairement à celle de Kullbach) est une distance géométrique et vérifie donc toutes les propriétés d'une distance :*
- $\mathrm{d}(x,x) = 0$;
- $\mathrm{d}(x,y) = \mathrm{d}(y,x)$;
- $\mathrm{d}(x,z) \leq \mathrm{d}(x,y) + \mathrm{d}(y,z)$.

On peut alors définir la boule de rayon t et de centre c :

$$B(c,t) := \{y \mid \mathrm{d}(c,y) \leq t\}.$$

De même, on définit la sphère

$$S(c,t) := \{y \mid \mathrm{d}(c,y) = t\} \subseteq B(c,t).$$

Cas typique

t erreurs avec $t \approx pn$: $(p-\alpha)n \leq t \leq (p+\alpha)n$.

Géométriquement, cela veut dire que le vecteur reçu $y \in B(c, t = (p+\alpha)n)$.

On se place dans y et on cherche dans $B(y,t)$ pour trouver le mot émis. S'il y a deux éléments dans cette boule, il risque d'y avoir des ambiguïté.

$|S(x,t)| = \binom{n}{t} = \binom{n}{\tau n} \approx 2^{nh(\tau)}$ avec $h : x \mapsto x \log \frac{1}{x} + (1-x) \log \frac{1}{1-x}$.

$$\begin{aligned}
|B(x,t)| &= 1 + \binom{n}{1} + \binom{n}{2} + \ldots + \binom{n}{t} \\
&\approx \sum_{i=0}^{t} 2^{nh\left(\frac{i}{n}\right)} \\
&\leq (t+1) \max_i 2^{nh\left(\frac{i}{n}\right)} \\
&\leq n 2^{nh(\tau)}.
\end{aligned}$$

Donc, d'après la majoration un peu brutal, on voit que le nombre d'éléments de la boule est à un facteur près celui de la sphère !

II Codage aléatoire

Soit $C = 1 - h(p)$ la capacité d'un code. Choisissons un rendement $R = 1 - h(p) - \varepsilon$. Alors $|\mathcal{C}| = 2^{n(1-h(p)-\varepsilon)}$ mots.

$$c \longrightarrow \boxed{\text{canal}} \longrightarrow y$$

Quelle est la probabilité que dans $B(y,t)$ il y ait un autre "candidat" ? Autrement dit :

$$\begin{aligned}
P\left[(\mathcal{C} \cap B(y,t)) = \{c\}\right] &= \underbrace{\left(1 - \frac{|B(y,t)|}{2^n}\right)}_{\text{pr un mot}}^{|\mathcal{C}|} \\
&= \underbrace{\frac{1}{1 - \frac{|B(y,t)|}{2^n}}}_{=:\varphi} \exp\left(|\mathcal{C}| \ln\left(1 - \frac{|B(y,t)|}{2^n}\right)\right) \\
&= \varphi \exp\left(|\mathcal{C}| \left(-\frac{|B|}{2^n} - \frac{1}{2}\left(\frac{|B|}{2^n}\right)^2 + \mathbf{o}\left(\frac{|B|^2}{2^{2n}}\right)\right)\right) \\
&= \varphi \exp\left(-\frac{|\mathcal{C}|\,|B|}{2^n}\right)
\end{aligned}$$

Or, $|B| \approx 2^{nh(p+\alpha)}$, $|\mathcal{C}| = 2^{n(1-h(p)-\varepsilon)}$, alors

$$|B|\,|\mathcal{C}| = 2^{1-\varepsilon/2} \text{avec } \alpha \text{ bien choisi.}$$

Donc, on a

$$\varphi \exp\left(-\frac{2^{n(1-\varepsilon/2)}}{2^n}\right) = \varphi \underbrace{\exp\left(-2^{-\frac{\varepsilon n}{2}} + \cdots\right)}_{\longrightarrow 1}.$$

$\Longrightarrow$ Le décodeur à vraisemblance maximale ne se trompe presque jamais !

Si $|\mathcal{C}| > 2^{n(1-h(p)+\varepsilon)}$, quel que soit le code choisi, dès que l'on au-dessus de la capacité on ne pourra reconstituer le code émis. Dans la formule précédente on trouverait

$$|\mathcal{C}|\,|B| = 2^{n(1+\varepsilon)}$$

$\Rightarrow$ « dans l'espace, on arrive pas à caser toutes les boules ». Partant d'un mot de codes, on va forcément rentrer dans une autre boule.

Théorème [5.8] (Théorème de Shannon) – *Un code aléatoire de longueur n ($\to \infty$) permet de décoder avec erreur $\to 0$ à des rendements arbitrairement proche de la capacité.*

– *Au-delà de la capacité, la probabilité de faire des erreurs de décodage* $\xrightarrow{n\to\infty} 1$.

III Lien avec le codage de source

$$\mathcal{M} \xrightarrow{\text{loi qcq}} \boxed{\text{compression}} \xrightarrow{\text{loi uniforme}} M \text{ avec loi uniforme} \xrightarrow{\text{codage}} \boxed{\text{canal}}$$

M s'écrit avec I bits d'informations $\Rightarrow 2^I$ messages.

CHAPITRE 6

CODES CORRECTEURS, CODES LINÉAIRES

Rappel : D'après Shannon, il existe des « *bons* » codes. Presque tous les longs codes sont bons, approchent la capacité. Par contre, si $R = \frac{1}{2}$, alors $|\mathcal{C}| = 2^{Rn} = 2^{\frac{n}{2}}$. On ne peut pas écrire la liste des mots de code !

I Codes linéaires

Définition [6.1] *Un code $\mathcal{C}$ est appelé* ***linéaire*** *s'il est stable par addition (pour $\mathbb{F} = \{0,1\}$)/s'il est sous-espace vectoriel de $\mathbb{F}^n = \{0,1\}^n$.*

$\mathcal{C}$ admet une base $g_1, \ldots, g_k \in \{0,1\}^n \iff \forall c \in \mathcal{C}, \exists! I \subseteq \{1,2,\ldots,k\}$ tel que $c = \sum_{i \in I} g_i$.

Définition [6.2] *On appelle* ***matrice génératrice*** *du code $\mathcal{C}$ une matrice*

$$G := \underbrace{\begin{bmatrix} g_1 \\ \vdots \\ g_k \end{bmatrix}}_{n \text{ colonnes}}$$

dont les lignes forment une base de $\mathcal{C}$.
On a de plus que $\#(\text{lignes de } G) = \dim \mathcal{C} =: k$.

Proposition [6.3] *$|\mathcal{C}| = 2^k$. Le rendement devient $R = \frac{k}{n}$.*

I.1 Décodage d'effacements

Encodage

$\mathcal{M} = \{0,1\}^k \to \mathcal{C}$, $x = (x_1, \ldots, x_k) \mapsto x_1 g_1 + \ldots + x_k g_k = xG$

Exemple [6.4] *$c = (1\,1\,0\,1\,0\,1\,0\,1\,1)$ émis, $v = (1\,\varepsilon\,0\,1\,\varepsilon\,\varepsilon\,0\,1\,1)$ reçu.*
Comment reconstituer c à partir de v ? On a un code linéaire, alors c admet une écriture unique

$$c = \sum_{i=1}^{k} x_i g_i \qquad x_i \in \{0,1\}$$

avec

$$c_j = \sum_{i=1}^{k} x_i G_{ij} \tag{$*$}$$

où G_ij désigne la $j^{\text{ème}}$ colonne de la matrice G.

Les vecteurs de $\{0,1\}^k$ $(x_1,\dots,x_k)$ possibles sont les solutions du système linéaire $(*)$, $j=1,\dots,n$, à k inconnues et $(n-\#(\text{effacement}))$ équations. Un code est bon s'il n'y a presque toujours qu'une solution. S'il reste des ambiguïté, il n'est pas bon.

I.2 Quels sont les bons codes linéaires ?

Une réponse partielle peut être donnée : Un bon code a une bonne distance de Hamming minimale

$$d_{\min} := \min_{\substack{x,y\in\mathcal{C}\\ x\neq y}} \mathrm{d}(x,y).$$

Proposition [6.5] *Si $\mathcal{C}$ est linéaire, alors d_{min} s'écrit*

$$d_{min} = \min_{\substack{c\in\mathcal{C}\\ c\neq 0}} \text{poids}(c)$$

où $\text{poids}(c) = |\{i \mid c_i \neq 0\}| = \mathrm{d}(c,0)$.

Proposition [6.6] *Notons $d_{min}(\mathcal{C}) =: d$. $\mathcal{C}$ corrige n'importe quel motif de*
- *$d-1$ effacements*
- *e erreurs avec $e < \frac{d}{2}$.*

Ceci est valable pour tous les codes !

(1) $\underbrace{\varepsilon\varepsilon\dots\varepsilon}_{\leq d-1}1010\dots$. On ne peut pas corriger équivaut à dire qu'il existe c_1, c_2 qui coïncident en dehors de l'ensemble effacé. $\Longrightarrow \mathrm{d}(c_1,c_2) \leq d-1$.

(2) c émis, v reçu. erreur = changement ! (pas effacement)
$\mathrm{d}(c,v) = e$. Soit $c' \in \mathcal{C}$, $c' \neq c$. Quelle est la distance entre c' et v ?

$$d_{\min}(\mathcal{C}) = d \leq \mathrm{d}(c,c') \leq \underbrace{\mathrm{d}(c,v)}_{<\frac{d}{2}} + \mathrm{d}(v,c').$$

Donc $\mathrm{d}(c',v) > \frac{d}{2}$. Dit autrement, c est le mot le plus proche de v (en distance de Hamming).

Comment corriger *une* erreur ? On efface une position, puis on applique le correcteur d'effacements. Si aucun mot de code ne correspond, on continue en effaçant la position suivante. On fait ceci jusqu'à ce que l'on trouve la « bonne » position.

I.3 Code dual

Soient $x,y \in \{0;1\}^n$, $x=(x_1,\dots,x_n)$, $y=(y_1,\dots,y_n)$. On définit un « produit scalaire » de x et y par

$$\langle x,y\rangle = x_1y_1 + \dots + x_ny_n \in \{0;1\}.$$

I.4 Code dual de $\mathcal{C}$

Définition [6.7] *Le **dual** ou **orthogonal** $\mathcal{C}^\perp$ du code $\mathcal{C}$ est défini par*

$$\mathcal{C}^\perp := \{x \in \{0;1\} \mid \forall c \in \mathcal{C},\ \langle x,c\rangle = 0\}.$$

Proposition [6.8] $\underbrace{\dim\mathcal{C}}_{=k} + \dim\mathcal{C}^\perp = n.$

Définition [6.9] *On appelle **matrice de parité** (ou de contrôle) de $\mathcal{C}$ une matrice génératrice du dual $\mathcal{C}^\perp$:*

$$H = \left[\begin{array}{ccc} h_1 & \dots & h_n \end{array}\right] \Bigg\} r = n-k \text{ lignes}$$

Proposition [6.10] *Soit* $x \in \{0;1\}^n$. $x \in \mathcal{C} \iff \forall y \in \mathcal{C}^\perp, \langle x, y\rangle = 0$.

Vérifier tous les mots du dual, ce serait trop de travail et ce serait trop coûteux. Mais on a la proposition suivante qui dit qu'il suffit de vérifier les vecteurs de la base du dual :

Proposition [6.11] $x \in \mathcal{C} \iff Hx = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} =: \mathbb{0}$.

Définition [6.12] *On appelle* ***fonction syndrome*** σ *de* $\mathcal{C}$

$$\sigma : \{0;1\}^n \to \{0;1\}^r$$
$$x = (x_1, \ldots, x_n) \mapsto \sigma(x) := \sum_{i=1}^{n} h_i x_i = Hx$$

où $r := n - k = n - \dim \mathcal{C}$.

Proposition [6.13] $x \in \mathcal{C} \iff \sigma(x) = \mathbb{0}$.

Exemple [6.14] $H = \begin{bmatrix} 1 & \ldots & 1 \end{bmatrix}$.
$\Rightarrow \mathcal{C} = \{x \mid \sum x_i = 0\} = \{x \mid \text{poids}(x) \textit{ est pair}\}$.

Proposition [6.15] *On ne regarde que des codes linéaires.*

(1) $d_{min} \geq 2 \iff \forall i\ h_i \neq 0$.

(2) $d_{min} \geq 3 \iff \forall i, j, i \neq j, h_i \neq h_j$

Exemple [6.16] (codes de Hamming) $H = \begin{bmatrix} h_1 & \ldots & h_n \end{bmatrix}$. *On met le nombre maximal de colonnes de longueur r :*

$$n = 2^r - 1 \qquad (\text{“}-1\text{”} \textit{ car on ne peut mettre la colonne nulle})$$
$$k = n - r = 2^r - r - 1$$
$$d = 3$$

Par exemple, $r = 3$ *:*

$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

$$h_1 + h_2 = h_4 \iff h_1 + h_2 + h_4 = \mathbb{0}$$
$$\iff \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \in \mathcal{C}$$

II Formes systématiques de G et de H

Définition [6.17] *On dit que G est sous forme systématique si*

$$G = \left[\ I_k \ \middle|\ A\ \right].$$

On a alors que

$$x = (x_1, \ldots, x_k) \mapsto xG = (\underbrace{x_1, \ldots, x_k}_{\textit{bits d'info}}, \underbrace{x_{k+1}, \ldots, x_n}_{\substack{\textit{bits de parité,} \\ \textit{dép. de } A}}) = c$$

Définition [6.18] *On dit que H est sous forme systématique si*

$$H = \left[\ B \ \middle|\ I_r\ \right].$$

Proposition [6.19] *Si* $[I_k \mid A]$ *génératrice de* $\mathcal{C}$*, alors* $\left[A^{\mathrm{T}} \mid I_{n-k}\right] = H$ *matrice de parité de* $\mathcal{C}$.

Définition [6.20] *On appelle* ***paramètre d'un code linéaire*** *le triplet constituant de la longueur, de la dimension et de la distance minimal du code :*

$$[n,\ k,\ d]$$

Définition [6.21] *La* ***borne de Hamming sur*** d_{min} *est définie par*

$$t := \texttt{floor}\left(\frac{d_{min}-1}{2}\right)$$

Si $\mathrm{d}(c,x) \leq t$, *alors* $\forall c' \in \mathcal{C}, c \neq c', \mathrm{d}(c',x) > t$.

La quantité $|\mathcal{C}| \cdot |B| \leq 2^n$ *est appelée* ***borne de Hamming*** *où, rappelons-le,* $|B| = 1 + \binom{n}{1} + \binom{n}{2} + \ldots \binom{n}{t}$. *Par conséquent,*

$$|\mathcal{C}| \leq \frac{2^n}{1 + \binom{n}{1} + \binom{n}{2} + \ldots \binom{n}{t}}$$

Ceci est une borne pour *tous les codes* !

Pour un code de Hamming, on a en particulier :

$$\begin{aligned} |\mathcal{C}| &\leq \frac{2^n}{1+n} & &= \frac{2^n}{1+2^r-1} \\ &= 2^k \\ &= |\mathcal{C}| \ ; \end{aligned}$$

cela veut dire que pour le code de Hamming, la borne est atteinte. Dans ce cas, on dit que $\mathcal{C}$ est **parfait**.

Codes parfaits linéaires

- $\mathcal{C} = \{0;1\}^n$;
- $\mathcal{C} = \{(0,\ldots,0);\ (1,\ldots,1)\}$, n impair ;
- les codes de Hamming ;
- **le** code de Holay avec les paramètres $[23,\ 12,\ 7]$

III Décodage d'une erreur

Soit $H = \begin{bmatrix} h_1 & \ldots & h_n \end{bmatrix}$ la matrice de parité et $x = c + \varepsilon$ un vecteur erronné, c étant le message envoyé et ε le vecteur d'erreur.

On a alors :

$$\begin{aligned} \sigma(x) &= \underbrace{\sigma(c)}_{=0 \text{ car } c \in \mathcal{C}} + \sigma(\varepsilon) \\ &= \sigma(\varepsilon) \\ &= h_i \end{aligned}$$

où i est la coordonnée où x contient l'erreur.

Plus généralement : $\sigma(x) = \sigma(c) + \sigma(\varepsilon) = \sigma(\varepsilon) = \sum_{i \in I} h_i$ où I est l'ensembles des indices où il y a des erreurs.

Problème du décodage : Le problème, c'est écrire $s = \sigma(x)$ comme somme du plus petit nombre possible de colonnes de H.

Trouver de « bons » codes linéaires

Un bon code ...

- a une grande distance minimale,
- est proche de la capacité de Shannon ce qui minimalise la probabilité de faire une erreur de décodage et
- a un bon algorithme de décodage.

CHAPITRE 7

CODES LINÉAIRES ALÉATOIRES

Soit $\mathcal{C}$ un code et $H \in \mathbb{F}_2^{n\times r}$ une matrice aléatoire uniforme.
$\mathcal{C}$ s'écrit donc sous la forme suivante :

$$\begin{aligned}\mathcal{C} &= \{x \in \{0;1\}^n \mid Hx = \mathbb{0}\} \\ &= \left\{ x \in \{0;1\}^n \;\middle|\; \sum_{i=1}^{n} h_i x_i = \mathbb{0} \right\}\end{aligned}$$

et $\dim \mathcal{C} \geq n - r$ (car toutes les lignes ne sont éventuellement pas libres).

Soit A_i le nombre de mots de $\mathcal{C}$ de poids i. Avec une grande probabilité, les premiers A_i valent zéro. On veut calculer la valeur moyenne des A_i :

$$A_i = \sum_{\substack{x\in\{0;1\}^n \\ \text{poids}(x)=i}} X_x \qquad \text{où } X_x = \mathbb{1}_{\mathcal{C}}(x)$$

$$\begin{aligned}E(A_i) &= \sum_x E(X_x) \\ &= \sum_x P(X_x = 1) \\ &= \sum_x P(x \in \mathcal{C}) \\ &= \sum_x P(\sigma(x) = \mathbb{0}) \\ &= \sum_x P\left(\sum_{j \text{ tq } x_i=1} h_j \right) \qquad h_j \text{ aléatoire !} \\ &= \frac{1}{2^r}\binom{n}{i}\end{aligned}$$

Avec $k = \dim \mathcal{C} = Rn \geq n - r \iff r \geq (1-R)n$, on trouve

$$\begin{aligned}E(A_i) &= \frac{\binom{n}{i}}{2^{n(1-R)}} \\ &\approx \frac{2^{nh(i/n)}}{2^{n(1-R)}} \\ &= 2^{n[h(\lambda)-(1-R)-\mathbf{o}(1)]}\end{aligned}$$

Pour i petit, $E(A_i)$ est très proche de 0.

Conclusion

(1) Pour n grand, on ne trouvera jamais de mot de poids **inférieur** à $n[h^{-1}(1-R)-\varepsilon]$, car avec $P(A_i > 0) \leq E(A_i)$, on a pour

$$\begin{aligned} P(A_1 > 0 \text{ ou } A_2 > 0 \text{ ou } \ldots \text{ ou } A_i > 0) &\leq \sum_i P(A_i) \\ &\leq \sum_i E(A_i) \\ &\leq iE(A_i) \\ &\leq n\binom{n}{i}2^{-r}. \end{aligned}$$

(2) Soit $d = \delta n$, $0 < \delta < \frac{1}{2}$, code aléatoire de rendement $R < 1 - h(\delta)$. Quand n tend vers ∞, $R \approx 1 - h(\delta)$.

(Pour comprendre ce qui se passe : http://fr.wikipedia.org/wiki/Borne_de_Gilbert-Varshamov.)

Comparaison avec la borne de Hamming

$$\begin{aligned} 2^{Rn} &= |\mathcal{C}| \\ &\leq \frac{2^n}{1 + \binom{n}{1} + \ldots + \binom{n}{\frac{\delta}{2}n}} \\ &\approx \frac{2^n}{\binom{n}{\frac{\delta}{n}n}} \\ &= 2^n\,(1 - h(\delta/2)) \end{aligned}$$

Donc, $R \leq 1 - h(\delta/2)$.

Remarque [7.1] *Prenons $p = \delta$. Alors, $C = 1 - h(p)$. D'après Shannon : $\exists$ codes avec $R \approx 1 - h(p) = 1 - h(\delta)$ qui corrigent sans erreurs presque toutes les configurations de pn erreurs.*
Presque tous les codes linéaires atteignent la capacité de CBS avec $p = \delta - \varepsilon$.
Autrement dit : on choisit $\mathcal{C}$ au hasard avec un rendement R fixe.

Exemple *On prend $R = \frac{1}{2}$.*
- *$\delta = h^{-1}(1 - R) \approx 0.11$.*
- *$\mathcal{C}$ corrige presque tous les motifs de $(\delta - \varepsilon)n$ erreurs ($\approx 0.11r$).*

Remarque *D'après un théorème précédent, on peut corriger $\frac{d}{2}$ erreur dans* tous les cas. *On peut espérer d'après le résultat ci-dessus en corriger plus, notamment presque $\delta n = d$ erreurs !*

CHAPITRE 8

CODAGE PAR SYNDROME – QUESTIONS DE PARTAGE, DE COMMUNICATION DE SECRET

I Réconciliation

(i) se mettre d'accord sur un code correcteur, p.ex.

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad (*)$$

(ii) envoyer le syndrome, p.ex.

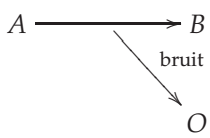
$$\sigma(x) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

(iii) comparer syndrome reçu et trouver l'erreur (s'il y en a), p.ex.

$$\sigma(x') = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

On sait qu equand $x' = x + \varepsilon$, alors $\sigma(x) + \sigma(x') = \sigma(x + x') = \sigma(\varepsilon)$.

II Communication confidentielle



Un tel canal est appelé **Wiretap channel** (ou **canal à jarretière**) : Il y a une communication entre A et B avec un observateur O.

II.1 Wiretap II

O intercepte t bits.

Stratégie pour transmettre un secret s : choisir x aléatoirement tel que $\sigma(x) = s$. envoyer x sur le canal entre A et B.

Exemple : $x = \begin{bmatrix}1 & 0 & 1 & 1 & 0 & 1 & 0\end{bmatrix}$. Si O intercepte les 4 premiers bits, il aurait comme information $\begin{bmatrix}1 & 0 & 1 & 1 & a & b & c\end{bmatrix}$, alors en décodant avec le code correcteur H dans $(*)$, O trouve

$$s = \begin{bmatrix}1\\1\\1\end{bmatrix} + \begin{bmatrix}1\\0\\1\end{bmatrix} + \begin{bmatrix}1\\1\\0\end{bmatrix} + a\begin{bmatrix}1\\0\\0\end{bmatrix} + b\begin{bmatrix}0\\1\\0\end{bmatrix} + c\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}1\\0\\0\end{bmatrix} + \begin{bmatrix}a\\b\\c\end{bmatrix}$$

Autrement dit, il ne connaît rien sur le secret s envoyé ! On va choisir les quatres symboles interceptés plus astucieusement, à savoir les quatres symboles correspondant au support de la première ligne de H : O intercepte donc

$$\begin{bmatrix}1 & a & 1 & 1 & 0 & b & c\end{bmatrix}$$

et en déduit (dans un calcul analogue au précédent)

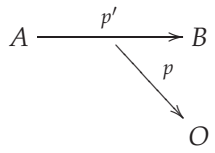
$$s = \begin{bmatrix}1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\a+b\\a+c\end{bmatrix}$$

Comme ça, O peut connaître le premier bit du secret s !

Par contre, si on n'interceptait que 3 bits, on n'aurait **aucune** information sur le secret s :

Théorème [8.1] (Wyner) *Si H est une matrice de parité d'un code $\mathcal{C}$, si $t < d_{min}\left(\mathcal{C}^{\perp}\right)$, alors l'interception de t symboles quelconques de x ne donne **aucune** information sur $s = \sigma(x)$.*

II.2 Wiretap I



Cas simple : $p' = 0$.

Stratégie : s à transmettre entre A et B. envoyer x tel que $\sigma(x) = s$. x aléatoire uniforme parmi les x tels que $\sigma(x) = s$.

Choix de H ? Quelle information obtient O sur x ? $1 - h(p)$ bits par symbole intercepté. Donc taille maximale du secret :

$$n - n(1 - h(p)) = nh(p)$$

Donc $r \leq nh(p)$.

Si on écrit $s = \sigma(x)$, O obtient $x + b$. Alors, $\sigma(x + b) = s + \sigma(b)$. Un b "typique" est un vecteur de poids pn. Il y en a $\binom{p}{np} \approx 2^{nh(p)} = 2^r$.

« L'observateur n'a rien d'autre à faire que calculer $\sigma(x + b)$ et "jeter" $x + b$ », car on a la proposition suivante :

Proposition [8.2] *On a*

$$H(s|x + b) = H(s|\sigma(x + b)).$$

Proposition [8.3] *Si X est une variable aléatoire de S et si $H(X) \geq r - \varepsilon$, alors $H(S|S + X) \geq H(S) - \varepsilon$.*

La proposition [8.2] dit que l'observateur n'apprend pas plus de ε bits d'information sur le secret s si $H(\sigma(b)) > r - \varepsilon$.

Problème : Comment assurer cela ? Comment choisir σ ?

Pour avoir $H(\sigma(b)) < H(b)$, on doit choisir $r \leq H(b)$.

[...]

Pour étudier $H(\sigma(b))$, on introduit l'**entropie de Renyi** d'une variable aléatoire réelle X qui prend ses valeurs x dans $\mathcal{X}$ avec une loi de probabilité $P(X = x) =: p_x$:

Shannon	**Renyi**
$H(X) = -\sum_x p_x \log p_x$	$R(X) = -\log\left(\sum_x p_x^2\right)$
moyenne des logs	log des moyennes
$H(X) = E[-\log P_x]$	$R(X) = -\log E[P_X]$

$\sum_x p_x^2$ est aussi appelé **probabilité de collision**.
Supposons X et Y deux variables aléatoires indépendants de même loi.

$$P(X = Y) = \sum P(X = x \text{ et } Y = x) = \sum p_x^2.$$

Comme la fonction log est convexe, on a

Proposition [8.4] $R(X) \leq H(X)$.

Exemple [8.5] *Si X suit une loi uniforme dans $\mathcal{X} = \{1, 2, \ldots, n\}$, alors $R(X) = \log_2 n$.*

Si on choisit σ au hasard dans un ensemble Σ, on a

$$E[R(\Sigma(b))] > r - \varepsilon.$$

Propriétés de Σ

Soient b et b' fixés, $b \neq b'$.

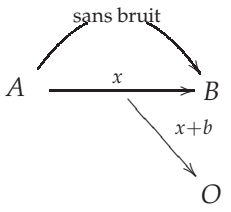
$$P(\Sigma(b) = \Sigma(b')) = \frac{1}{2^r}.$$

Hypothèse : B est un bruit aléatoire dans $\{0;1\}^r$. Alors $\Sigma : \{0;1\}^n \to \{0;1\}^r$ linéaire. Alors, on trouve :

$$\begin{aligned}
E[(R(\Sigma(b))] &= \sum_\sigma P(\Sigma = \sigma) P(\sigma(B)) \\
&= P(\Sigma = \sigma)(-\log P(\sigma(B_1) = \sigma(B_2))) \\
&\geq -\log \sum_\sigma P(\Sigma = \sigma) P(\Sigma(B_1) = \Sigma(B_2) | \Sigma = \sigma) \\
&\geq -\log P(\Sigma(B_1) = \Sigma(B_2)) && \leftarrow \text{Bayes} \\
&\geq -\log \big[\underbrace{P(B_1 = B_2}_{=2^{R(B}} + \underbrace{P(\Sigma(B_1) = \Sigma(B_2) | B_1 \neq B_2)}_{2^{-r}} \\
&\geq -\log 2^r (1 + 2^{r - R(B)}) \\
&\geq r - \log(1 + 2^{r - R(B)})
\end{aligned}$$

Remarque [8.6]
- $H(\Sigma(B)) \geq R(\Sigma(B))$
- *Pour des lois uniformes, on a $R = H$.*
- *Pour b obtenu par CBS, $R(B) < H(B)$.*

III Cas général d'un partage de secret inconditionnel



reçoit $y = x + e$

Trois étapes

(1) « *Advantage distillation* »
Conversation entre A et B sur un canal sans bruit.

(2) Réconciliation
A envoie sur le canal sans bruit un $\sigma_\varepsilon(x) \in \{0;1\}^{r_\varepsilon}$ qui permet de corriger ε et qui ne donne que r_ε bits d'information à O.

(3) Extraction de secret
De nouveau $s = \sigma(x) \in \{0;1\}^r$, $r < H(\sigma(b)) - r_\varepsilon$. Taille maximale du secret et $H(b) - H(\varepsilon)$.