QUELQUES ÉLÉMENTS DE THÉORIE DE L'INFORMATION

FRANÇOIS CHAPON

Université de Toulouse

2024–2025

Table des matières

1. Introduction et motivations

La théorie de l'information est née suite à l'article fondateur de Shannon [Shannon, A Mathematical Theory of Communication (1948)].

Le schéma de communication de la figure 1.1 correspond au modèle proposé par Shannon. S'il peut paraître évident aujourd'hui, il représentait une avancée novatrice à son époque. Pour la première fois, les rôles distincts de la source, de l'émetteur, du canal, du récepteur et du bruit y étaient clairement définis. Ce modèle constitue encore aujourd'hui la base des systèmes de communication et du numérique.

La théorie de Shannon est de plus de nature probabiliste, le récepteur étant incertain du message réellement transmis, on considère la source du message comme étant aléatoire. On cherche ainsi à quantifier la quantité d'information contenu dans un message aléatoire, représenté par une variable aléatoire X : c'est la notion d'entropie introduite par Shannon qui va mesurer la quantité d'information fournie en moyenne par X. Tant qu'on ne connaît pas X, il s'agit plutôt d'une quantité d'incertitude.

Pour mesurer l'information ou l'incertitude liée à la réalisation d'un état x d'une variable aléatoire, il est naturel de la définir comme étant une fonction F de sa probabilité p de réalisation, et d'imposer que F vérifie les axiomes suivants :

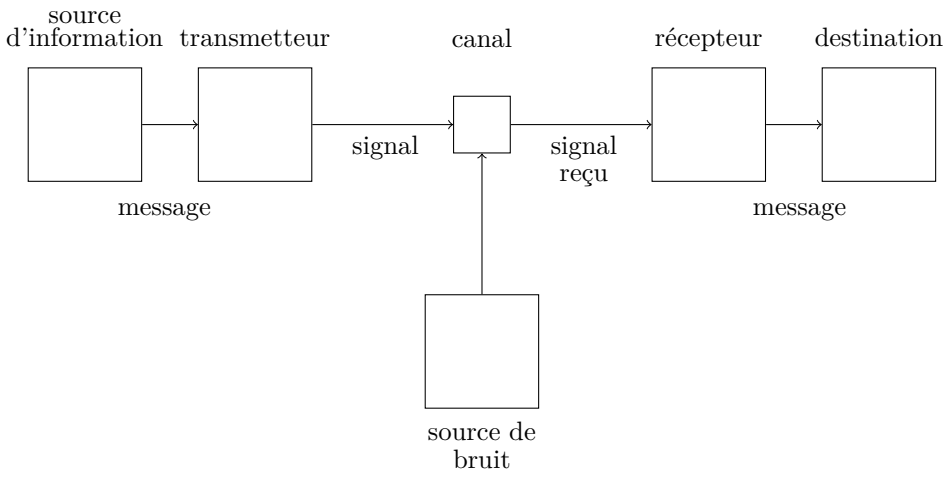


FIGURE 1.1. Schéma d'un système de communication issu de l'article de Shannon.

- F est une quantité positive ;
- F est une fonction décroissante en p. Si $p = 1$, l'information est sûre, et son incertitude vaut 0 ;
- Si A et B sont deux événements indépendants, de probabilité de réalisation p et q respectivement, on doit avoir que l'incertitude liée à la réalisation de $A \cap B$ est la somme de l'incertitudes liée à A et de celle liée à B, c'est-à-dire, la probabilité de réalisation de $A \cap B$ étant pq, on doit avoir
$$F(pq) = F(p) + F(q) ;$$
- F est continue en p. Une petite variation sur la réalisation de p doit entraîner une petite variation sur l'incertitude liée à x.

Les seules fonctions continues sur $]0, 1]$ et satisfaisant l'équation fonctionnelle $F(pq) = F(p) + F(q)$ étant les fonctions logarithmes, on prend comme définition pour l'incertitude liée à la réalisation d'un état de probabilité p : $F(p) = -\log p$ (à une constante multiplicative près). Si X est une v.a. sur un ensemble fini, de loi $(p_i)_{1 \leq i \leq k}$, son incertitude moyenne, est alors définie par

$$-\sum_{i=1}^{k} p_i \log_2(p_i).$$

C'est l'entropie telle que définie par Shannon. On prend le logarithme en base 2 pour une question de normalisation qu'on justifiera par la suite.

La notion d'entropie tire en fait son origine dans la physique statistique, où elle mesure la quantité de désordre d'un système. La formule de Boltzmann (1877) définit l'entropie microcanonique d'un système à l'équilibre macroscopique par la relation

$$S = k_B \log W,$$

où k_B est la constante de Boltzmann et W le nombre de configurations du système. Si on suppose que le système est composé de N particules indiscernables pouvant être dans k états différents de façon équiprobable, en notant N_i le nombre de particules dans l'état i, on a alors que le nombre de configurations possibles est le coefficient multinomial

$$W = \frac{N!}{N_1! \cdots N_k!},$$

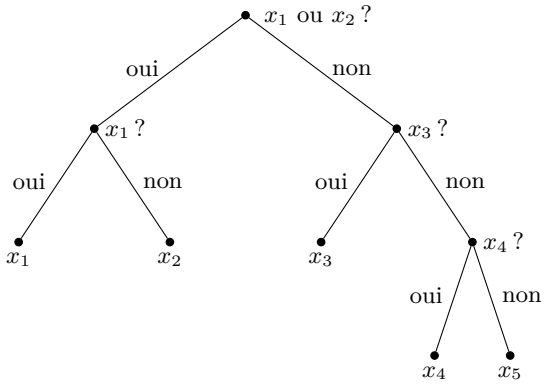
avec $N = N_1 + \cdots + N_k$. L'entropie moyenne du système s'écrit alors, en utilisant l'équivalent $\log(N!) \sim N \log N$, quand N est grand,

$$S = \frac{1}{N} k_B \log \left(\frac{N!}{N_1! \cdots N_k!} \right) \sim -k_B \sum_{i=1}^{k} p_i \log p_i,$$

où on a noté $p_i = \frac{N_i}{N}$ la proportion de particules dans l'état i. On retrouve exactement la définition de l'entropie de Shannon (à une constante multiplicative près).

La notion d'entropie intervient dans différents domaines des mathématiques, de la physique, de l'informatique, et possède de nombreuses applications en théorie de l'information, du codage, de la compression de données, etc... Nous allons voir quelques éléments de base de la théorie de l'information et quelques applications à la théorie du codage. Donnons d'abord deux autres exemples où l'entropie apparaît naturellement.

Exemple 1.1. Considérons une v.a. X à valeurs dans $\{x_1, x_2, x_3, x_4, x_5\}$ de probabilités respectivement $0.3, 0.2, 0.2, 0.15, 0.15$. On cherche à déterminer le résultat de X à l'aide de questions à deux réponses possibles oui/non. Un exemple d'un tel questionnaire est représenté par l'arbre suivant :



Avec ce questionnaire, en 2 ou 3 questions on détermine quelle valeur de X s'est produite, et le nombre moyen de questions à poser est alors :

$$2 \times 0.3 + 2 \times 0.2 + 2 \times 0.2 + 3 \times 0.15 + 3 \times 0.15 = 2.3.$$

Ceci est ici très légèrement supérieur à l'entropie de X, donnée par

$$-(0.3 \log_2(0.3) + 0.2 \log_2(0.2) + 0.2 \log_2(0.2) + 0.15 \log_2(0.15) + 0.15 \log_2(0.15)) = 2.27.$$

On verra que le théorème de Shannon nous donne que l'entropie minimise le nombre moyen de questions à poser pour déterminer X.

Exemple 1.2. On considère un dé truqué. On note $p(k)$, pour $k = 1, \ldots, 6$, la probabilité que le dé tombe sur k. On lance successivement n fois le dé truqué de façon indépendante. Un résultat est donc une suite $(x_1, \ldots, x_n)$ de $\{1, \ldots, 6\}^n$. On s'intéresse à la probabilité d'obtenir une suite « typique », au sens de « toutes les suites sauf une proportion qui tend vers 0 quand n tend vers l'infini ».

La probabilité d'obtenir une suite $(x_1, \ldots, x_n)$ est, par indépendance des lancers,

$$p(x_1) \cdots p(x_n) = \prod_{l=1}^{6} p(k)^{N(k)},$$

où $N(k)$ est le nombre de fois où on est tombé sur k. Or par la loi des grands nombres, $N(k)$ est proche de $np(k)$, quand n est grand, avec forte probabilité. Donc, quand n est

grand, pour la plupart des tirages, la probabilité cherchée est proche de

$$\prod_{l=1}^{6} p(k)^{np(k)} = \left(\prod_{l=1}^{6} p(k)^{p(k)}\right)^{n} = 2^{-nH},$$

où $H = -\sum_{k=1}^{6} p(k)\log_2(p(k))$ est l'entropie. Ainsi, 2^{nH} donne le nombre de suites « typiques ».

On suit essentiellement le début du livre [Cover-Thomas, Elements of Information Theory].

2. Entropie

2.1. Définitions.

Notation 2.1. On note $\log_2$ le logarithme en base 2, donc $\log_2(x) = \frac{\log(x)}{\log 2}$, pour $x > 0$, où log désigne le logarithme népérien. On a donc

$$\log_2(x) = y \Leftrightarrow x = 2^y,$$

pour $x > 0$, et $y \in \mathbb{R}$. Par convention, on posera $0 \times \log 0 = 0$ (qui provient du prolongement par continuité de la fonction $x \mapsto x \log x$ en 0).

Dans cette partie, on se placera dans le cadre de variables aléatoires à valeurs dans un ensemble **fini**. Soit X une v.a. à valeurs dans E. On notera souvent p sa loi, c'est-à-dire que pour tout $x \in E$, $p(x) = \mathbb{P}(X = x)$, donc pour tout $x \in E$, $p(x) \in [0,1]$ et $\sum_{x\in E} p(x) = 1$. À ne pas confondre avec la notation usuelle $p \in [0,1]$ pour désigner une probabilité !

Définition 2.1. *Soit X une v.a. à valeurs dans un ensemble fini E de loi p. L'entropie de p (ou de X), notée $H(p)$ (ou $H(X)$), est le nombre positif*

$$H(p) = -\sum_{x\in E} p(x)\log_2(p(x)).$$

Elle est exprimée en bits.

Remarque 2.1. En voyant p la loi de X comme une fonction $p\colon E \to [0,1]$, on peut exprimer l'entropie comme une espérance :

$$H(X) = \mathbb{E}(-\log_2(p(X))).$$

Exemple 2.1. Soit X une v.a. de Bernoulli sur $\{0,1\}$ de paramètre α. Alors,

$$H(X) = -\alpha\log_2(\alpha) - (1-\alpha)\log_2(1-\alpha).$$

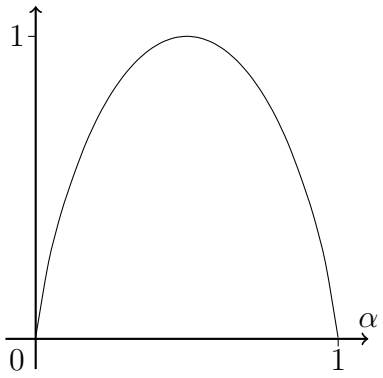


FIGURE 2.2. Entropie d'un v.a. de Bernoulli de paramètre α : $H(\alpha) = -\alpha \log_2(\alpha) - (1-\alpha)\log_2(1-\alpha)$. L'entropie vaut 0 en $\alpha = 0$ ou $\alpha = 1$. L'information transmise étant sûre, il n'y a aucune incertitude sur l'information transmise. L'entropie est maximale en $\alpha = \frac{1}{2}$, et vaut 1 bit. La plus incertaine des variables de Bernoulli est celle pour laquelle $\alpha = \frac{1}{2}$.

Exemple 2.2. Soit μ la loi uniforme sur E. Alors,

$$H(\mu) = -\sum_{x \in E} \frac{1}{|E|} \log_2 \left(\frac{1}{|E|} \right) = \log_2(|E|).$$

Par exemple, si $|E| = 256 = 2^8$, l'entropie vaut 8 bits.

Exemple 2.3. Soit X une v.a. à valeurs dans $\{a, b, c, d\}$ de probabilités respectives $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}$. L'entropie de X est alors

$$\begin{aligned} H(X) &= -\frac{1}{2}\log_2 \frac{1}{2} - \frac{1}{4}\log_2 \frac{1}{4} - \frac{1}{8}\log_2 \frac{1}{8} - \frac{1}{8}\log_2 \frac{1}{8} \\ &= \frac{7}{4} \text{ bits.} \end{aligned}$$

Définition 2.2. *Soit (X, Y) un couple de v.a. à valeurs dans $E \times F$ fini. On note $p_{(X,Y)}$ la loi jointe de (X, Y), i.e. $p_{(X,Y)}(x, y) = \mathbb{P}(X = x, Y = y)$, pour tout $(x, y) \in E \times F$. On appelle entropie mutuelle de (X, Y), notée $H(X, Y)$, l'entropie de $p_{(X,Y)}$, c'est-à-dire*

$$H(X, Y) = -\sum_{x \in E} \sum_{y \in F} p_{(X,Y)}(x, y) \log_2 \left(p_{(X,Y)}(x, y) \right).$$

Définition 2.3. *Soit X et Y deux v.a. à valeurs dans E et F respectivement. On appelle entropie conditionnelle de X sachant Y la quantité*

$$H(X|Y) = -\sum_{x \in E} \sum_{y \in F} p_{(X,Y)}(x, y) \log_2 \left(\frac{p_{(X,Y)}(x, y)}{p_Y(y)} \right),$$

où $p_{(X,Y)}$ désigne la loi du couple (X, Y) et p_Y la loi de Y.

Lorsque que $Y = X$, on a $H(X|X) = 0$. Si on connaît X, il n'y a aucune incertitude sur X, et donc son entropie conditionnelle est nulle. Si X et Y sont indépendantes, alors $H(X|Y) = H(X)$. La connaissance de Y n'apporte aucune information supplémentaire sur X. En général, $H(X|Y) \neq H(Y|X)$.

Proposition 2.1. *Soit X et Y deux variables aléatoires. On a*

$$H(X, Y) = H(X) + H(Y|X).$$

En particulier, si X et Y sont indépendantes, alors $H(X, Y) = H(X) + H(Y)$.

Par symétrie, on a aussi : $H(X,Y) = H(Y) + H(X|Y)$.

Démonstration. Laissée en exercice. □

Exemple 2.4. Soit (X,Y) un couple de v.a. de loi jointe :

Y \ X	1	2	3	4
1	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{32}$	$\frac{1}{32}$
2	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{32}$	$\frac{1}{32}$
3	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
4	$\frac{1}{4}$	0	0	0

Les lois marginales sont alors : $\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right)$ pour X et $\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$ pour Y. On a donc $H(X) = \frac{7}{4}$ et $H(Y) = 2$. On calcule de plus

$$H(X|Y) = \frac{11}{8}, \quad H(Y|X) = \frac{13}{8}, \quad H(X,Y) = \frac{27}{8}.$$

2.2. **Entropie relative.**

Définition 2.4. *Soit p et q deux lois de probabilité sur le même ensemble fini E. Leur entropie relative (ou divergence de Kullback-Leibler) est la quantité*

$$D(p||q) = \sum_{x \in E} p(x) \log_2 \left(\frac{p(x)}{q(x)} \right).$$

On va voir dans le théorème suivant que l'entropie relative entre deux distributions p et q est une sorte de distance entre p et q, mais pas au sens mathématique du terme car elle n'est pas symétrique (elle ne vérifie pas non plus l'inégalité triangulaire). De plus, elle peut prendre la valeur $+\infty$ si $q(x) = 0$ et $p(x) \neq 0$.

Théorème 2.1. *Soit p et q deux lois de probabilité sur le même ensemble fini E. Alors $D(p||q) \geq 0$ avec égalité si et seulement si $p = q$.*

Démonstration. On peut supposer que $p(x) > 0$ pour tout $x \in E$, sinon on ajoute des termes nuls. On peut aussi supposer que $q(x) > 0$ pour tout $x \in E$, sinon $D(p||q) = +\infty$, et il n'y a rien à montrer. En utilisant le fait que $\log(x) \leq x - 1$ pour tout $x > 0$, avec égalité si et seulement si $x = 1$, on a :

$$\begin{aligned} D(p||q) &= -\sum_{x \in E} p(x) \log_2 \left(\frac{q(x)}{p(x)} \right) \\ &\geq -\frac{1}{\log(2)} \sum_{x \in E} p(x) \left(\frac{q(x)}{p(x)} - 1 \right) \\ &= -\frac{1}{\log(2)} \left(\sum_{x \in E} q(x) - \sum_{x \in E} p(x) \right) \\ &= 0. \end{aligned}$$

L'égalité a lieu si et seulement si $\frac{q(x)}{p(x)} = 1$ pour tout $x \in E$, i.e. $p = q$ (exo, par l'absurde). □

Corollaire 2.1. *L'entropie est maximale pour la loi uniforme sur E, i.e. pour toute loi de probabilité p sur E, on a*

$$H(p) \leq \log_2 |E|,$$

avec égalité si et seulement si p est la loi uniforme sur E.

Démonstration. Soit μ la loi uniforme sur E. On a alors pour toute loi de probabilité p sur E,

$$0 \leq D(p||\mu) = \log_2(|E|) - H(p),$$

donc $H(p) \leq \log_2(|E|)$, avec égalité si et seulement si $p = \mu$. □

Définition 2.5. *Soit X et Y deux v.a. de loi jointe $p_{(X,Y)}$, et de loi marginales p_X et p_Y respectivement. L'information mutuelle entre X et Y est la quantité*

$$I(X;Y) = \sum_{x\in E}\sum_{y\in F} p_{(X,Y)}(x,y)\log_2\left(\frac{p_{(X,Y)}(x,y)}{p_X(x)p_Y(y)}\right).$$

C'est donc l'entropie relative entre la loi $p_{(X,Y)}$ et la loi produit $p_X \otimes p_Y$, définie par $p_X \otimes p_Y(x,y) = p_X(x)p_Y(y)$ pour tout $(x,y) \in E \times F$ (i.e. la loi d'un couple de v.a. indépendantes de loi marginales p_X et p_Y),

$$I(X;Y) = D(p_{(X,Y)}||p_X \otimes p_Y).$$

Les propriétés suivantes sont faciles à démontrer et laissées en exercice :

Proposition 2.2. *On a :*

(i) $I(X;Y) = H(X) - H(X|Y)$;

(ii) $I(X;Y) = H(X) + H(Y) - H(X,Y)$;

(iii) $I(X;X) = H(X)$;

(iv) $I(X;Y) = H(X,Y) - H(X|Y) - H(Y|X)$.

Par l'égalité (i), on peut interpréter l'information mutuelle $I(X;Y)$ comme la réduction de l'incertitude de X due à la connaissance de Y. On remarquera de plus que I est symétrique.

Proposition 2.3. *Soit X et Y deux variables aléatoires. Alors,*

$$I(X;Y) \geq 0,$$

avec égalité si et seulement si X et Y sont indépendantes.

Démonstration. Comme $I(X;Y) = D(p_{(X,Y)}||p_X \otimes p_Y)$ où $p_{(X,Y)}$ est la loi jointe de (X,Y) et p_X et p_Y les lois marginales, on a

$$I(X;Y) = D(p_{(X,Y)}||p_X \otimes p_Y) \geq 0,$$

avec égalité si et seulement si $p_{(X,Y)} = p_X \otimes p_Y$, i.e. X et Y sont indépendantes. □

On en déduit aussi :

Proposition 2.4.

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

Ainsi, conditionner réduit l'entropie : plus on a d'information, plus l'incertitude est faible.

Démonstration. Comme $I(X;Y) \geq 0$ et $I(X;Y) = H(X) - H(X|Y)$, l'inégalité est immédiate. □

On peut résumer ces différentes quantités d'information par le diagramme de Venn suivant :

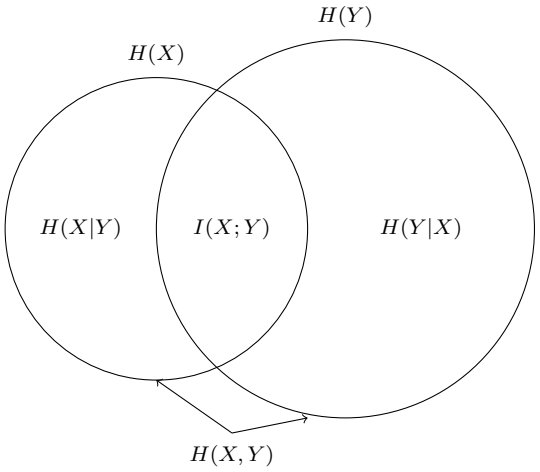


FIGURE 2.3. Relations entre les différentes quantités d'information

3. Codes et compression de données

3.1. **Définitions.** De façon informelle, un code est une règle de transcription qui à tout symbole d'un alphabet (l'alphabet source) assigne de manière unique une chaîne de caractères pris dans un autre alphabet (l'alphabet cible). On peut par exemple penser au code morse international qui établit un codage entre les lettres de l'alphabet latin et des séquences de sons courts et long, ou encore au code UTF-8 qui permet de coder l'ensemble des caractères en caractères informatiques.

Un code peut être de longueur fixe, tous les modes code ont même longueur. Par exemple, pour coder 2^n symboles en binaire, c'est-à-dire avec des suites de 0 et de 1, il faut n bits. Il peut être plus optimal de considérer des codes à longueur variable, et par exemple, de coder les symboles les plus fréquents par des mots code de longueurs les plus courtes.

La compression de données est aussi un problème de codage, où l'on cherche à coder un source de manière fiable avec un débit d'information minimal de sorte à comprimer au maximum la source (par exemple, le format MP3 qui permet de coder des sources audio ou le format jpeg pour la compression d'images).

Commençons par quelques définitions naturelles en théorie du codage.

Définition 3.1. *Soit $\mathcal{D}$ un ensemble fini de symboles, qu'on appellera alphabet. Un mot sur l'alphabet $\mathcal{D}$ est une suite finie $(a_1, \dots, a_n) \in \mathcal{D}^n$, qu'on notera $a_1 \cdots a_n$. La longueur du mot $a_1 \cdots a_n$ est n. Un mot peut être vide, c'est l'unique mot de longueur 0, noté ε. L'ensemble de tous les mots sur l'alphabet $\mathcal{D}$ est noté $\mathcal{D}^*$ (autrement dit $\mathcal{D}^* = \cup_{n\geq 0}\mathcal{D}^n$ est l'ensemble des suites de longueur finie de lettres prises dans $\mathcal{D}$). La concaténation de deux mots $a = a_1 \cdots a_n$ et $a' = a'_1 \cdots a'_k$ de $\mathcal{D}^*$ est le mot $aa' = a_1 \cdots a_n a'_1 \cdots a'_k$. Un mot a est un préfixe d'un mot a' s'il existe un mot w tel que $aw = a'$.*

Exemple 3.1. Sur l'alphabet $\mathcal{D} = \{0, 1\}$, l'ensemble $\mathcal{D}^*$ est donc l'ensemble des suites finies de 0 et de 1. Par exemple, 0010111 est un mot de $\mathcal{D}^*$ de longueur 7. La concaténation des mots 011 et 101 est le mot 011101. Les mots préfixes non vide de 001101 sont 0, 00, 001, 0011, 00110, et 001101.

Définition 3.2. *Soit $\mathcal{D}$ un ensemble fini ordonné. On ordonne $\mathcal{D}^*$ suivant l'ordre lexicographique : soit $a = a_1 \cdots a_r$ et $b = b_1 \cdots b_s$ deux mots dans $\mathcal{D}^*$. Alors, $a < b$ si et seulement si il existe un entier $1 \leq t \leq \min(r, s)$ tel que pour tout $i < t$, $a_i = b_i$ et*

$$a_t < b_t \quad ou \quad (t = r \text{ et } t < s).$$

C'est exactement l'ordre utilisé dans un dictionnaire. Par exemple, sur $\mathcal{D} = \{0, 1\}$ muni de l'ordre $0 < 1$, on a $0 < 00 < \cdots < 00\cdots 0 < 01 < 010 < \cdots < 1 < 10 < \cdots < 11 < \cdots$.

Passons maintenant à la définition de code.

Définition 3.3. *Soit X une v.a. à valeurs dans un ensemble E fini de loi p. Un code (source) est une application $C\colon E \to \mathcal{D}^*$. Le mot $C(x)$ est appelé le mot code correspondant à $x \in E$. On note $l(x)$ la longueur de $C(x)$, et l est appelée la fonction longueur de C (notons qu'elle dépend de C). La longueur moyenne du code C est*

$$L(C) := \mathbb{E}(l(X)) = \sum_{x\in E} p(x)l(x).$$

Exemple 3.2. Soit X une v.a. à valeurs dans $\{x_1, x_2, x_3, x_4\}$ de probabilités respectives $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}$. Un code C est par exemple donné par $C(x_1) = 0$, $C(x_2) = 10$, $C(x_3) = 110$, $C(x_4) = 111$ sur l'alphabet $\mathcal{D} = \{0, 1\}$. La longueur moyenne de C est ici

$$L(C) = \frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = \frac{7}{4}.$$

On remarquera que $L(C)$ est ici exactement égale à l'entropie $H(X)$.

Définition 3.4. *L'extension C^* d'un code C est définie comme l'application $C^*\colon E^* \to \mathcal{D}^*$ qui envoie une suite de mots $x_1 \cdots x_n$ de E^* sur la concaténation $C(x_1)\cdots C(x_n)$.*

L'extension d'un code permet donc de coder n'importe quel mot de E^* à l'aide du code C défini pour chaque symbole de E.

Définition 3.5. *On dit qu'un code est uniquement décodable si l'application $C^*\colon E^* \to \mathcal{D}^*$ est injective.*

Ainsi, une séquence de mots code d'un code uniquement décodable a une seule séquence source possible. Mais il peut être possible d'avoir à attendre la fin de la séquence de mots code pour le déchiffrer. On introduit alors le concept de :

Définition 3.6. *Un code est appelé code préfixe ou code instantané si aucun mot code n'est le préfixe d'un autre mot code.*

Il est alors clair que :

Proposition 3.1. *Tout code préfixe est uniquement décodable.*

En effet, un code préfixe est immédiatement décodable sans avoir à attendre la fin de la séquence. On lit $C(x_{i_1})C(x_{i_2})\cdots C(x_{i_k})$ dans l'ordre et comme le code est préfixe, $C(x_{i_1})$ est le mode code de x_{i_1}, puis on recommence avec $C(x_{i_2})\cdots C(x_{i_k})$, etc...

Exemple 3.3. Quelques exemples de code :

	Code 1	Code 2	Code 3
x_1	0	10	0
x_2	010	00	10
x_3	01	11	110
x_4	10	110	111

Le code 1 n'est pas uniquement décodable, par exemple 010 peut se décoder en x_2 ou x_3x_1 ou x_1x_4 ;

Le code 2 n'est pas préfixe (11 est un préfixe de 110), mais tout de même uniquement décodable : il suffit de considérer le nombre de 0 tout de suite après 11. Par exemple, le mot 101100110 code $x_1x_3x_2x_4$;

Le code 3 (c'est l'exemple 3.2) est un code préfixe (et donc uniquement décodable) : le mot 11001110 est forcément composé des mots code 110, 0, 0, 111 et 0, soit le codage du mot $x_3x_1x_1x_4x_1$.

3.2. **Inégalité de Kraft.**

Théorème 3.1 (Inégalité de Kraft-McMillan)**.** *Soit $C\colon E \to \mathcal{D}^*$ un code uniquement décodable dans un alphabet de taille $D = |\mathcal{D}|$, et soit l la fonction longueur de C. Alors,*

$$\sum_{x\in E} D^{-l(x)} \leq 1.$$

Réciproquement, toute fonction $l\colon E \to \mathbb{N}^*$ vérifiant l'inégalité précédente est la fonction longueur d'un code préfixe.

Remarque 3.1. En pratique, on peut donc ne considérer que des codes préfixes puisque si on considère un code uniquement décodable dont les longueurs des mots code sont données par la fonction l, alors il existe un code préfixe ayant la même fonction longueur.

Démonstration. Soit $k \geq 1$ un entier. En développant, on a

$$\begin{aligned}\left(\sum_{x\in E} D^{-l(x)}\right)^k &= \sum_{(x_1,\dots,x_k)\in E^k} D^{-(l(x_1)+\cdots+l(x_k))} \\ &= \sum_{x\in E^k} D^{-l(x)},\end{aligned}$$

où on a noté, pour $x \in E^k$, $l(x) = \sum_{i=1}^k l(x_i)$ la longueur du mot code $C(x)$ correspondant au mot $x = x_1 \cdots x_k$. Posons $L = \max\{l(x) \,|\, x \in E\}$. Soit $N(p)$ le nombre de mots de E^k codés par des codes de longueur p, i.e. $N(p) = \#\{x \in E^k \,|\, l(x) = p\}$. Comme C^* est injective, et qu'il y a au plus D^p mots de longueur p sur l'alphabet $\mathcal{D}$, on a $N(p) \leq D^p$. Alors,

$$\sum_{x\in E^k} D^{-l(x)} = \sum_{p=1}^{kL} \sum_{x\in E^k \,|\, l(x)=p} D^{-l(x)} = \sum_{p=1}^{kL} N(p)D^{-p} \leq \sum_{p=1}^{kL} D^p D^{-p} = kL.$$

Ainsi, $\sum_{x\in E} D^{-l(x)} \leq (kL)^{1/k}$, et en faisant $k \to \infty$, on obtient $\sum_{x\in E} D^{-l(x)} \leq 1$.

Réciproquement, on va encoder un code préfixe par un arbre. Donnons d'abord quelques définitions.

Définition 3.7. *En informatique théorique, un arbre $\mathcal{T}$ désigne une structure de données récursive : on part d'un sommet, appelé racine, qui possède des enfants reliés à la racine par une arête, et chaque enfant a lui même éventuellement des enfants, reliés à son parent par une arête, etc... Un sommet possédant des enfants est appelé un noeud, un sommet sans enfant est appelé une feuille. Un arbre se lit de haut en bas. On ordonne les enfants de gauche à droite. La profondeur d'un sommet est le nombre d'arêtes qui le séparent de la racine. Les sommets de profondeur n sont appelés la génération n. On dit qu'un arbre est k-aire si chaque sommet a au plus k enfants (on dit binaire pour $k = 2$).*

D'un point de vue de la théorie des graphes un arbre correspond donc à un graphe connexe acyclique enraciné et planaire.

L'ensemble des mots $\mathcal{D}^*$ sur l'alphabet $\mathcal{D}$ peut être représenté par un arbre étiqueté. On suppose que $\mathcal{D}$ est ordonné. On part de la racine qui représente le mot vide ε. On trace une arête issue de la racine pour chaque élément de $\mathcal{D}$, de gauche à droite, suivant l'ordre de $\mathcal{D}$. Chaque noeud représente alors les mots à 1 lettre sur $\mathcal{D}$. On itère pour chacun des noeuds. La 2^{e} génération représente donc les mots à 2 lettres sur $\mathcal{D}$, et on itère... La n^{e} génération représente donc les mots à n lettres sur $\mathcal{D}$. On obtient ainsi un arbre (infini) $|\mathcal{D}|$-aire dans lequel tout sommet a exactement $|\mathcal{D}|$ enfants (on parle d'arbre complet). La figure 3.4 ci-dessous montre un exemple avec $|\mathcal{D}| = 2$.

Un code préfixe C dans un alphabet $\mathcal{D}$ peut alors être représenté par un arbre $|\mathcal{D}|$-aire. On part de l'arbre infini codant $\mathcal{D}^*$, on parcourt chacune des branches dans l'ordre

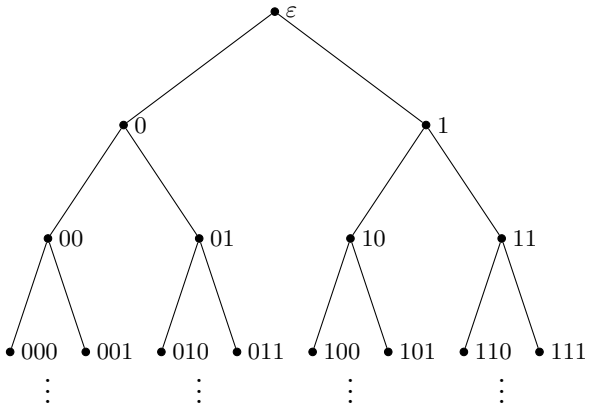


FIGURE 3.4. L'ensemble $\{0,1\}^*$ encodé par un arbre binaire infini.

lexicographique, et on marque chaque noeud correspondant à un mode code de C. On efface de l'arbre tous les noeuds qui ne sont ni marqués, ni des ancêtres de noeuds marqués. On obtient ainsi un arbre $|\mathcal{D}|$-aire, et les mots code de C sont alors donnés par les feuilles de l'arbre. La condition préfixe assure quand à elle qu'aucun mot code n'est le descendant d'un autre mot code. De plus, $l(x)$ est la profondeur de la feuille de l'arbre correspondant à $C(x)$. Voir les exemples des figures 3.5 et 3.6.

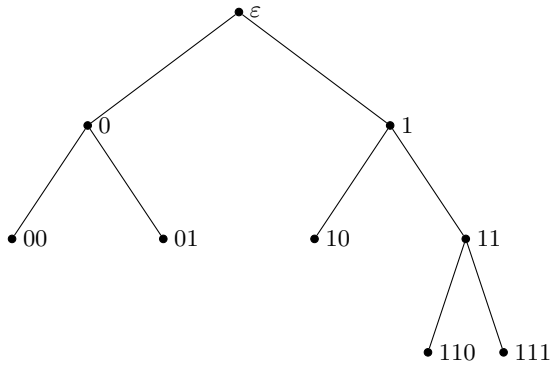


FIGURE 3.5. Le code préfixe $\{00, 01, 10, 110, 111\}$ représenté par un arbre binaire.

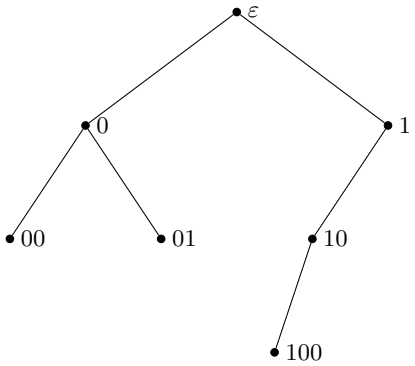


FIGURE 3.6. Autre exemple. Le code préfixe $\{00, 01, 100\}$ représenté par un arbre binaire.

Fin de la démonstration du Théorème 3.1. Soit $l\colon E \to \mathbb{N}^*$ une fonction vérifiant l'inégalité de Kraft. On ordonne les éléments de E par $x_1 < \cdots < x_n$ de telle sorte que l soit

croissante sur E. On part de l'arbre $\mathcal{T}$ encodant $\mathcal{D}^*$ qu'on arrête à la génération $l(x_n)$. On a donc $D^{l(x_n)}$ sommets à la génération $l(x_n)$. Soit $C(x_1)$ le premier élément (dans l'ordre lexicographique) de $\mathcal{T}$ de profondeur $l(x_1)$. On efface tous ses descendants de l'arbre. Soit $C(x_2)$ le premier élément de ce nouvel arbre de profondeur $l(x_2)$. On efface de même tous ses descendants de l'arbre. On itère le procédé jusqu'à $C(x_n)$. Il s'agit alors de s'assurer qu'on efface pas plus de sommets qu'il y en a de disponible. À chaque itération j, on efface donc $D^{l(x_n)-l(x_j)}$ sommets à la génération $l(x_n)$, et comme l vérifie l'inégalité de Kraft, pour tout $j \leq n$, on a

$$\sum_{i=1}^{j} D^{l(x_n)-l(x_i)} \leq D^{l(x_n)},$$

avec inégalité stricte pour $j < n$. Ceci assure qu'à chaque itération j, il reste des éléments de longueur $l(x_n)$, et donc de longueur inférieur à $l(x_n)$ en remontant les branches de l'arbre, permettant ainsi de construire le prochain mot code, et ce jusqu'à la dernière itération. □

3.3. **Théorème du codage de source de Shannon.**

Théorème 3.2 (Théorème de Shannon)**.** *Soit X une v.a. à valeurs dans un ensemble E fini. Soit $C\colon E \to \mathcal{D}^*$ un code uniquement décodable dans un alphabet à D lettres. Alors,*

$$\mathbb{E}(l(X)) \geq \frac{H(X)}{\log_2(D)}.$$

L'égalité est atteinte si et seulement si les valeurs $p(x)$ de la loi de X sont des puissances (négatives) de D.

Exemple 3.4. Si $D = 2$, le théorème 3.2 dit alors que l'entropie minimise la longueur moyenne de tout code binaire uniquement décodable de source X. Si D est quelconque, on a la même conclusion en remplaçant dans la définition de l'entropie le logarithme en base 2 par le logarithme en base D.

Reprenons l'exemple du questionnaire de l'exemple 1.1. Un questionnaire oui/non peut être vu comme un codage préfixe de X sur un alphabet binaire. Ainsi, l'entropie minimise le nombre moyen minimum de questions oui/non à poser pour déterminer le résultat d'une expérience aléatoire.

Démonstration. On définit une loi de probabilité q sur E en posant $q(x) = \frac{1}{c} D^{-l(x)}$, avec $c = \sum_{x\in E} D^{-l(x)}$. Soit p la loi de X. Alors,

$$\begin{aligned} D(p||q) &= \sum_{x\in E} p(x) \log_2 \frac{p(x)}{q(x)} \\ &= \sum_{x\in E} p(x) \log_2(p(x)) - \sum_{x\in E} p(x) \log_2(q(x)) \\ &= -H(X) + \log_2(c) + \log_2(D) \sum_{x\in E} p(x) l(x) \\ &= -H(X) + \log_2(c) + \log_2(D)\, \mathbb{E}(l(X)). \end{aligned}$$

Or $D(p||q) \geq 0$, et $c \leq 1$ par l'inégalité de Kraft, d'où $-H(X) + \log_2(D)\, \mathbb{E}(l(X)) \geq 0$, et donc l'inégalité

$$\mathbb{E}(l(X)) \geq \frac{H(X)}{\log_2(D)}.$$

Si l'égalité est vérifiée, comme $D(p||q) \geq 0$ et $c \leq 1$, on a forcément $c = 1$, et $D(p||q) = 0$, donc $p = q$, et donc les valeurs de p sont des puissances de D. Réciproquement, si p est à valeurs dans les puissances de D, on pose $l(x) = -\frac{\log_2(p(x))}{\log_2(D)}$, pour tout $x \in E$. Donc l est une

fonction sur E à valeurs entières strictement positives, et $\sum_{x\in E} D^{-l(x)} = \sum_{x\in E} p(x) = 1$. Par le théorème 3.1, il existe un code préfixe de fonction longueur l, et de longueur moyenne

$$\mathbb{E}(l(X)) = \sum_{x\in E} p(x)l(x) = -\sum_{x\in E} p(x)\frac{\log_2(p(x))}{\log_2(D)} = \frac{H(X)}{\log_2(D)}. \qquad \square$$

Remarque 3.2. En général, il existe un code C vérifiant

$$\frac{H(X)}{\log_2(D)} \leq \mathbb{E}(l(X)) < \frac{H(X)}{\log_2(D)} + 1.$$

En effet, pour tout $x \in E$, on pose

$$l(x) = \left\lceil -\frac{\log_2(p(x))}{\log_2(D)} \right\rceil,$$

où $\lceil t \rceil$ est le plus petit entier plus grand ou égal à t (donc $\lceil t \rceil$ est l'unique entier vérifiant $\lceil t \rceil - 1 < t \leq \lceil t \rceil$). Alors l vérifie l'inégalité de Kraft, car

$$\sum_{x\in E} D^{-l(x)} \leq \sum_{x\in E} D^{\frac{\log_2(p(x))}{\log_2(D)}} = \sum_{x\in E} p(x) = 1,$$

et donc il existe par le théorème 3.1 un code préfixe de fonction longueur donnée par l, et on a

$$\mathbb{E}(l(X)) = \sum_{x\in E} p(x)\left\lceil -\frac{\log_2(p(x))}{\log_2(D)} \right\rceil < \sum_{x\in E} p(x)\left(-\frac{\log_2(p(x))}{\log_2(D)} + 1 \right) = \frac{H(X)}{\log_2(D)} + 1.$$

On appelle code de Shannon, le code préfixe de fonction longueur $l(x) = \left\lceil -\frac{\log_2(p(x))}{\log_2(D)} \right\rceil$.

3.4. **Un exemple de code optimal : le code de Huffman.** On va présenter un algorithme permettant de produire un code optimal, c'est-à-dire de longueur moyenne minimale, pour les codes binaires (sur un alphabet $\{0, 1\}$).

Définition 3.8 (Algorithme de Huffman)**.** *Soit X une v.a. à valeurs dans un ensemble fini ordonné E.*

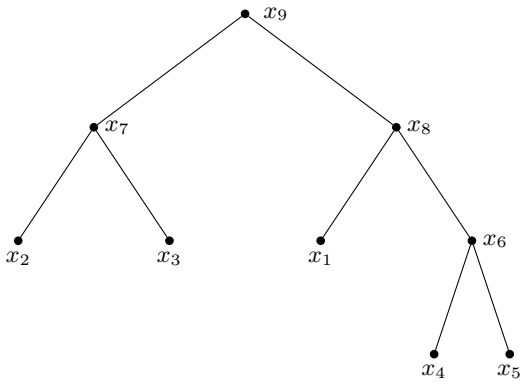
Soit x et x' les éléments de E dont les probabilités sont les plus petites (s'il y a plusieurs choix, on choisit les derniers selon l'ordre sur E). On les fusionne en leur donnant un parent commun y auquel on affecte la probabilité $p(y) = p(x) + p(x')$. On réitère la procédure sur l'espace $E \setminus \{x, x'\} \cup \{y\}$ jusqu'à obtenir un espace à un seul élément.

L'algorithme ainsi défini produit une bijection d'un espace fini ordonné E muni d'une probabilité sur les feuilles d'un arbre binaire.

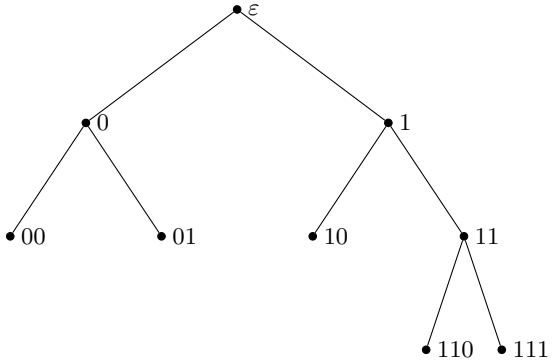
Exemple 3.5. Soit X sur $\{x_1, x_2, x_3, x_4, x_5\}$ de probabilités $0.25, 0.25, 0.2, 0.15, 0.15$.

- Itération 1 : on fusionne x_4 et x_5 en un noeud appelé x_6, de probabilité $0.15+0.15 = 0.3$. On obtient $E = \{x_1, x_2, x_3, x_6\}$ avec probabilités $0.25, 0.25, 0.2, 0.3$;
- Itération 2 : on fusionne x_2 et x_3 en un noeud appelé x_7, de probabilité $0.25+0.2 = 0.45$. On obtient $E = \{x_1, x_6, x_7\}$ avec probabilités $0.25, 0.3, 0.45$;
- Itération 3 : on fusionne x_1 et x_6 en un noeud appelé x_8, de probabilité $0.25+0.3 = 0.55$. On obtient $E = \{x_7, x_8\}$ avec probabilités $0.45, 0.55$;
- Itération 4 : on fusionne x_7 et x_8 en un noeud appelé x_9, de probabilité $0.45+0.55 = 1$.

On obtient l'arbre :



de codage :



c'est-à-dire que le codage C obtenu est :

x	x_1	x_2	x_3	x_4	x_5
$C(x)$	10	00	01	110	111

On remarquera que ce codage n'est pas unique.

Exemple 3.6. Exercice : Déterminer un codage d'Huffman du questionnaire oui/non de l'exemple 1.1.

Soit $\mathcal{T}$ un arbre binaire encodant un code préfixe C de source la v.a. X sur E. La profondeur d'une feuille étant égale à la longueur du mode code associé, on appelle profondeur moyenne de $\mathcal{T}$, notée $L(\mathcal{T})$, la longueur moyenne de C, i.e.

$$L(\mathcal{T}) = L(C) = \mathbb{E}(l(X)) = \sum_{x \in E} p(x) l(x).$$

On dit qu'un arbre $\mathcal{T}$ est optimal s'il est de profondeur moyenne minimale, i.e. pour tout arbre binaire $\mathcal{S}$ encodant C, on a

$$L(\mathcal{S}) \geq L(\mathcal{T}).$$

Un codage optimal n'est pas unique (loin de là), mais l'algorithme d'Huffman produit bien un arbre optimal, en effet :

Proposition 3.2. *Soit X une v.a. sur un ensemble fini E, de loi p. L'arbre construit par l'algorithme de Huffman est de profondeur moyenne minimale.*

La preuve est basée sur l'observation suivante :

Lemme 3.1. *Soit X une v.a. sur un ensemble fini E, de loi p. Il existe un arbre de codage optimal vérifiant les propriétés suivantes :*

(i) les longueurs des mots code sont en ordre inversé par rapport aux probabilités (i.e. si $p_i \leq p_j$, alors $l(x_i) \geq l(x_j)$) ;

(ii) les deux plus longs mots code ont la même longueur ;

(iii) deux des plus longs mots code différent seulement par le dernier bit (i.e. ont le même parent) et correspondent aux deux symboles les moins probables.

Démonstration. Soit $\mathcal{A}$ un arbre optimal codant la source X.

Pour (i) : il suffit de remarquer que si on échange deux symboles a et b dans $\mathcal{A}$, le nouvel arbre $\mathcal{A}'$ sera de profondeur moyenne

$$L(\mathcal{A}') = L(\mathcal{A}) + (p(a) - p(b))\,(l(b) - l(a))\,.$$

Donc si $p(a) \leq p(b)$, et comme $\mathcal{A}$ est de profondeur moyenne minimale, i.e. $L(\mathcal{A}')-L(\mathcal{A}) \geq 0$, on a donc $l(b) - l(a) \leq 0$, i.e. $l(a) \geq l(b)$.

Pour (ii) : si les deux plus longs mots code n'avaient pas la même longueur, il suffirait de remplacer le plus long des mots code par son parent (ce qui garderait la propriété préfixe), ce qui diminuerai la longueur moyenne de l'arbre, contredisant son optimalité.

Pour (iii) : tous les arbres optimaux ne vérifient pas cette propriété, mais on peut construire un arbre optimal ayant cette propriété en réarrangeant les mots code. Si le mot code de longueur maximale n'avait pas de feuille soeur, de même que précédemment, on pourrait le remplacer par son parent ce qui diminuerai la longueur moyenne de l'arbre, contredisant son optimalité. Ainsi, tout mot code de longueur maximale dans $\mathcal{A}$ possède une feuille soeur. On peut alors échanger les mots code de longueur maximale pour que les deux symboles les moins probables soient issus du même parent, ceci sans changer la profondeur moyenne de $\mathcal{A}$. Ainsi, deux des plus longs mots code différent seulement par le dernier bit. □

Démonstration de la proposition 3.2. Par induction sur $|E|$. Si $|E| = 2$, il n'y a qu'un seul arbre binaire qui encode E, et il n'y a rien à montrer. Supposons que l'algorithme d'Huffman produise un arbre optimal pour les codes source des ensembles finis de cardinal $n-1$.

Soit E un ensemble à n éléments et X une v.a. sur E de probabilité $p_1 \leq \cdots \leq p_n$. Soit $\mathcal{H}$ l'arbre construit par l'algorithme d'Huffman, et soit $\mathcal{T}$ un arbre issu d'un codage préfixe de X.

Par le lemme précédent, on peut supposer que dans $\mathcal{T}$ et $\mathcal{H}$, les deux symboles les moins probables sont à la profondeur maximale de l'arbre et sont issus du même parent.

On applique la 1re itération de l'algorithme d'Huffman à $\mathcal{T}$, c'est-à-dire qu'on efface x_1 et x_2 et on les remplace par leur parent auquel on affecte la probabilité $p' = p_1 + p_2$. On obtient donc un arbre à $n-1$ feuilles, codant $n-1$ symboles de probabilités $p', p_3, \ldots, p_n$. Notons $\mathcal{T}'$ ce nouvel arbre issu de $\mathcal{T}$, et $N = \max_{x\in E} l(x)$. On a donc

$$L(\mathcal{T}') = L(\mathcal{T}) - p_1 N - p_2 N + (p_1 + p_2)(N-1) = L(\mathcal{T}) - (p_1 + p_2).$$

On applique de même la 1re itération de l'algorithme d'Huffman à $\mathcal{H}$. Il s'agit de remarquer (par récurrence) qu'on obtient alors un arbre d'Huffman $\mathcal{H}'$ codant une v.a. de probabilités $p', p_3, \ldots, p_n$ sur un ensemble à $n-1$ symboles. On a de même que précédemment,

$$L(\mathcal{H}') = L(\mathcal{H}) - (p_1 + p_2).$$

Par hypothèse de récurrence, $\mathcal{H}'$ est optimal, donc $L(\mathcal{T}') \geq L(\mathcal{H}')$, et donc

$$L(\mathcal{T}) \geq L(\mathcal{H}),$$

ce qui prouve que $\mathcal{H}$ est optimal. □