Théorie de L'information

Nicolas Sendrier

Chapitre 1

Systèmes de communication

1.1 Introduction

La théorie des communications s'intéresse aux moyens de transmettre une information depuis une source jusqu'à un utilisateur (cf. Figure 1.1). La nature de la *source* peut-être très variée. Il peut s'agir par exemple d'une voix, d'un signal électromagnétique ou d'une séquence de symboles binaires. Le *canal* peut être une ligne téléphonique, une liaison radio ou encore un support magnétique ou optique : bande magnétique ou disque compact. Le canal sera généralement perturbé par un *bruit* qui dépendra de l'environnement et de la nature canal : perturbations électriques, rayures, ... Le *codeur* représente l'ensemble des opérations effectuées sur la sortie de la source avant la transmission. Ces opérations peuvent être, par exemple, la modulation, la compression ou encore l'ajout d'une redondance pour combattre les effets du bruit. Elles ont pour but de rendre la sortie de la source compatible avec le canal. Enfin, le *décodeur* devra être capable, à partir de la sortie du canal de restituer de façon acceptable l'information fournie par la source.

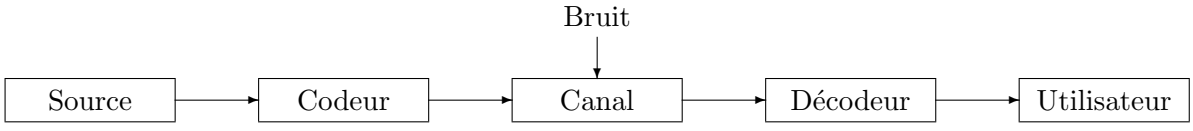


Fig. 1.1 – Shéma d'un système de communication.

Dans les années 40, C. E. Shannon a développé une théorie mathématique appelée *théorie de l'information* qui décrit les aspects les plus fondamentaux des systèmes de communication. Cette théorie s'intéresse à la construction et à l'étude de modèles mathématiques à l'aide essentiellement de la théorie des probabilités. Depuis ce premier exposé, la théorie de l'information s'est faite de plus en plus précise et est devenue aujourd'hui incoutournable dans la conception tout système de communication.

Dans ce cours, nous étudierons certains de ces modèles mathématiques, qui, bien que

considérablement plus simple que les sources et les canaux physiques, permettent de donner une bonne intuition de leur comportement.

Dans le but de simplifier l'étude des systèmes de communication, nous étudierons séparément les modèles de sources et les modèles de canaux. Ceci peut se schématiser en séparant le codeur et le décodeur de la Figure 1.2 en deux parties. Le but du codeur de source est de représenter la sortie de la source, ou information, en une séquence binaire, et cela de la façon la plus économique possible. Le but du codeur de canal et de son décodeur est de reproduire le plus fidèlement possible cette séquence binaire malgré le passage à travers le canal bruité.

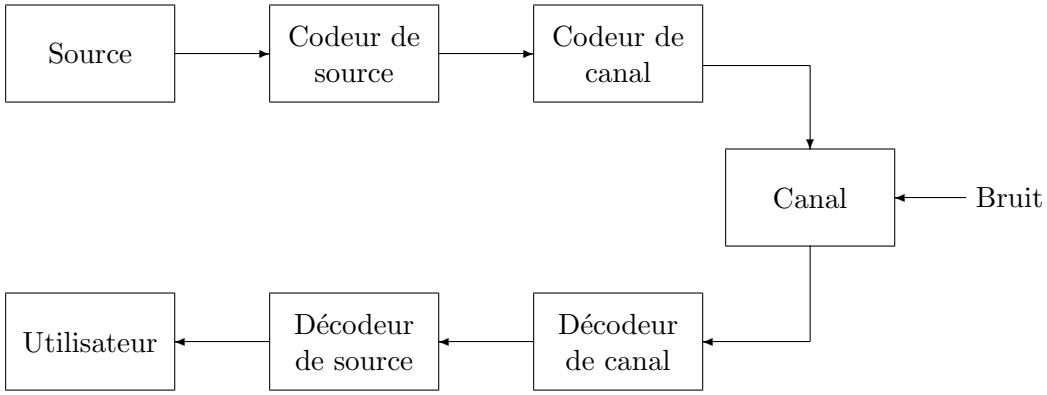


FIG. 1.2 – Système de communication avec codeurs de source et de canal séparés.

Cette séparation rendra notre étude commode, car nous pourrons traiter indépendemment la source et le canal. De plus la théorie montre que dans la pluspart des cas, cela n'implique aucune limitation sur les performances du système. Nous poursuivrons donc ce chapitre d'introduction par une brève description des différentes classes de modèles de sources et de canaux.

1.2 Sources et codage de source

1.2.1 Sources discrètes sans mémoire

Parmi les classes possibles de modèles de sources, nous nous intéresserons plus particulièrement aux *sources discrètes sans mémoire*. La sortie d'une telle source est une séquence de lettres tirées dans un alphabet fini $\{a_1, \ldots, a_K\}$. Chaque lettre de la séquence est choisie aléatoirement d'après une loi de probabilité $P(a_1), \ldots, P(a_K)$ indépendante du temps, c'est-à-dire indépendante de la position de la lettre dans la séquence. Pour toute lettre a_k, $1 \leq k \leq K$, $P(a_K)$ est la probabilité pour que cette lettre soit choisie, on aura donc $\sum_{k=1}^{K} P(a_k) = 1$.

Il peut sembler étonnant de modéliser une source d'information à l'aide d'une variable aléatoire. Nous allons essayer de nous convaincre sur un exemple qu'il s'agit bien de la

bonne démarche.

Exemple : Soit une source d'information qui fournit comme information l'une des quatre lettres a_1, a_2, a_3 et a_4. Supposons que le codage transforme cette information en symboles binaires. Dans la Table 1.1, nous donnons deux codages différents de cette source. Dans la première méthode, deux symboles binaires sont générés pour chaque

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

TAB. 1.1 – Deux codages d'un alphabet de quatre lettres.

lettre émise, alors que dans la seconde le nombre de symboles est variable.

Si les quatre lettres sont équiprobables, alors la première méthode est la meilleure : 2 symboles par lettre en moyenne au lieu de 2,25. En revanche si l'on a

$$P(a_1) = \frac{1}{2}, \quad P(a_2) = \frac{1}{4}, \quad P(a_3) = P(a_4) = \frac{1}{8},$$

alors la méthode 1 nécessite toujours 2 symboles binaires par lettre en moyenne alors que la méthode 2 qui n'en nécessite que 1,75. Elle est dans ce cas la plus économique.

Il est donc important pour coder correctement une source de connaitre son comportement statistique.

1.2.2 Entropie d'une source discrète

Il apparait qu'il existe un lien entre l'information fournie par une source et la distribution de probabilité de la sortie de cette source. Plus l'évènement donné par la source est probable, moins la quantité d'information correspondante est grande. Plus précisement, si une lettre a_k a pour probabilité $P(a_k)$ d'être tirée, son *information propre* sera $I(a_k) = -\log_2 P(a_k)$. Cette définition parait conforme à l'idée intuitive que l'on peut se faire de l'information, et en particulier on a $I(a_k) = 0$ si $P(a_k) = 1$, c'est-à-dire que l'occurence d'un évènement certain ne peut fournir aucune information.

La valeur moyenne de l'information propre calculée sur l'ensemble de l'alphabet revet une grande importance. Elle est appelée *entropie* de la source et vaut

$$\sum_{k=1}^{K} -P(a_k) \log_2 P(a_k).$$

L'entropie d'une source est le nombre moyen minimal de symboles binaires par lettre nécessaires pour représenter la source.

Par exemple, si un alphabet contient 2^L lettres équiprobables, il est immédiat que l'entropie de la source correspondante vaut L. Or il est bien clair que pour représenter 2^L lettres distinctes, L symboles binaires sont nécessaires.

L'entropie d'une source est parfois donnée en bits par seconde, si l'entropie d'une source discrète est H, et si les lettres sont émises toutes les τ_s secondes, son entropie en bits/s sera H/τ_s.

1.2.3 Autres modèles de sources

On peut citer également parmi les classes de modèles de source, les *sources discrètes avec mémoire*, pour lesquelles une entropie peut être définie de façon analogue.

Enfin, les sources non discrètes, ou *sources continues*, ont une grande importance pour les applications. La sortie d'une telle source sera une fonction continue du temps, par exemple une tension, qu'il faut coder en une séquence binaire. La fonction continue devra être décrite le plus fidèlement possible par la séquence binaire générée par le codeur de source. Le problème essentiel dans ce cas consiste à minimiser le nombre de symboles transmis pour un niveau de distorsion donné.

1.3 Canaux et codage de canal

1.3.1 Canaux discrets

Pour modéliser correctement un canal de transmission il est nécessaire de spécifier l'ensemble des entrées et l'ensemble des sorties possibles. Le cas le plus simple est celui du *canal discret sans mémoire* : l'entrée est une lettre d'un alphabet fini $\{a_1, \ldots, a_K\}$, et la sortie est une lettre d'un autre alphabet fini, éventuellement identique, $\{b_1, \ldots, b_J\}$. Ces lettres sont émises en séquence, et pour que le canal soit sans mémoire, il faut que chaque lettre de la séquence reçue ne dépende statistiquement que de la lettre émise de même position. Ainsi un canal discret sans mémoire est entièrement décrit par la donnée des probabilités conditionnelles $P(b_j \mid a_k)$, pour tout a_k dans l'alphabet d'entrée et tout b_j dans l'alphabet de sortie.

Par exemple le canal binaire symétrique, représenté dans la Figure 1.3, est un canal discret sans mémoire, dont l'alphabet d'entrée et l'alphabet de sortie sont égaux tous deux à $\{0, 1\}$. La probabilité pour qu'un symbole soit inchangé est $1 - \epsilon$, et la probabilité pour qu'il soit transformé en son opposé est ϵ.

On peut également considérer des canaux discrets à mémoire dans lesquels chaque lettre de la séquence de sortie peut dépendre de plusieurs lettres de la séquence d'entrée.

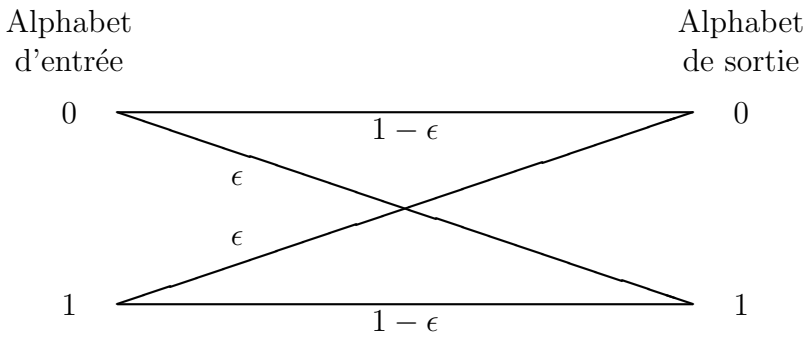


FIG. 1.3 – Canal binaire symétrique.

1.3.2 Canaux continus

Il existe une classe de modèles de canaux, appelé *canaux continus*, beaucoup plus proche des canaux physiques, dans lesquels l'entrée et la sortie sont des fonctions continues du temps.

Pour les canaux de cette classe, il est possible, et commode, de séparer le codeur et le décodeur en deux parties, comme le montre la figure 1.4. La première partie du codeur, que nous appelerons codeur de canal discret, transforme une séquence binaire en une séquence de lettres d'un alphabet fini $\{a_1, \ldots, a_K\}$, la seconde partie du codeur, le modulateur de données digitales (MDD) envoie pendant un temps τ_c sur le canal une des fonctions du temps prédéfinies $s_1(t), \ldots, s_K(t)$. La durée τ_c sera l'intervalle de temps séparant l'émission de deux lettre par le codeur de canal discret. L'ensemble de ces fonctions du temps mises bout à bout sera convertie à la sortie du canal par le démodulateur de données digitales (DDD) en une séquence de lettres d'un alphabet de sortie $\{b_1, \ldots, b_J\}$ au rythme, là encore, d'une lettre toutes les τ_c secondes.

On voit que de cette manière l'ensemble MDD-canal-DDD peut être considéré comme un canal discret dont l'alphabet d'entrée est $\{a_1, \ldots, a_K\}$ et l'alphabet de sortie $\{b_1, \ldots, b_J\}$. Si de plus le bruit est indépendant entre chaque intervalle de τ_c secondes, alors ce canal sera sans mémoire. L'étude des canaux discrets nous permettra donc de déduire des résultats sur les canaux continus.

1.3.3 Capacité d'un canal

L'un des paramètres les plus important d'un canal est sa *capacité*, nous verrons que cette capacité peut s'interpréter comme une mesure de la quantité d'information, exprimée en bits par seconde par exemple, pouvant être transmise à travers ce canal. L'intéret de cette grandeur nous vient essentielement du théorème sur le codage des canaux bruités : grossiérement, ce théorème dit que dans un canal de capacité C bits/s, si l'on veut faire transiter une quantité d'information à un *taux utile* de R bits/s, tel que $R < C$, alors il existe une procédure de codage et une procédure de décodage telles que le *taux d'erreur*

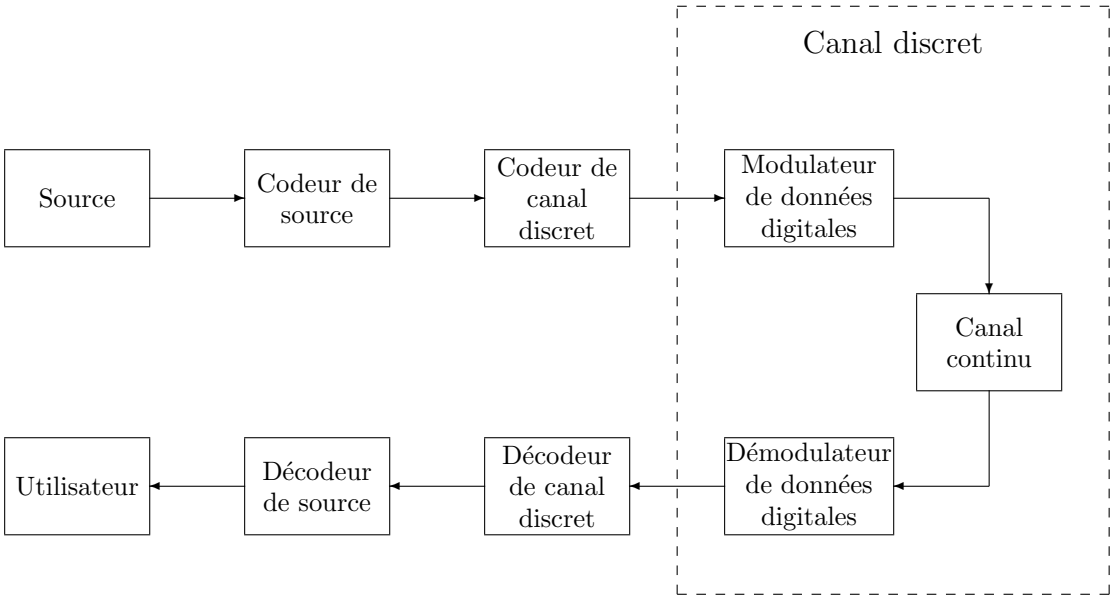


FIG. 1.4 – Canal continu et canal discret.

résiduel soit arbitrairement faible.

La réciproque de ce théorème nous dit par contre que si $R > C$ alors pour toute procédure de codage et de décodage, le taux d'erreur résiduel est supérieur à une constante strictement positive qui dépend de R et C.

Chapitre 2

Une mesure de l'information

Nous donnons dans ce chapitre une mesure de la quantité d'information adaptée à la description statistique des sources et des canaux. Les énoncés qui en résultent faisant largement appel aux probabilités discrètes, nous commencerons l'exposé par le rappel de quelques éléments de théorie des probabilités.

2.1 Rappels de théorie des probabilités discrètes

2.1.1 Espace probabilisé discret

Une expérience aléatoire se décrit mathématiquement par la donnée de l'ensemble des résultats, ou issues, possibles de cette expérience, cet ensemble est appelé espace des epreuves.

Nous noterons par une majuscule, ici $U = \{a_1, \ldots, a_K\}$, l'espace des épreuves, et par la minuscule correspondante, ici u, la variable aléatoire dont la valeur est égale à l'issue de l'expérience.

Les différents résultats de l'expérience aléatoire sont alors : "$u = a_1$", "$u = a_2$", ..., "$u = a_K$". Une loi de probabilité sur U est la donnée des probabilités de chacuns des résultats possibles : $Pr(u = a_k)$, $1 \leq k \leq K$, que nous noterons $P_U(a_k)$, ou plus simplement, lorsqu'aucune confusion sera possible $P(a_k)$. On a bien sùr, $P(a_k) \geq 0$ pour tout k, et $P(a_1) + \ldots + P(a_K) = 1$.

L'espace U muni d'une loi de probabilité P_U est appelé espace probabilisé.

L'espace probabilisé U pourra représenter par exemple une source discrète dont l'alphabet de sortie sera $a_1, \ldots, a_K$, et où $P(a_k)$ sera la probabilité pour que la lettre a_k soit tirée.

2.1.2 Variable aléatoire

Une variable aléatoire (v.a. en abrégé) d'un espace probabilisé $U = \{a_1, \ldots, a_K\}$ est définie comme une application dont l'ensemble de départ est U et dont l'ensemble d'arrivée

est quelconque.

Par exemple la v.a. u dont la valeur est égale à l'issue de l'expérience associée à U est l'application identité $U \to U$.

Une v.a. v à valeur dans l'ensemble des réels est appelée variable aléatoire réelle. On peut définir la moyenne d'une telle v.a. par

$$\bar{v} = \sum_{k=1}^{K} P(a_k) v(a_k).$$

2.1.3 Espace probabilisé joint – Probabilités conditionnelles

a - Espace probabilisé joint

Pour modéliser un canal discret, nous considérons un espace des épreuves $X \times Y$, produit des deux ensembles $X = \{a_1, \ldots, a_k\}$, et $Y = \{b_1, \ldots, b_J\}$. Le produit $X \times Y$ est l'ensemble des couples (a_k, b_j) pour tout k, $1 \leq k \leq K$ et tout j, $1 \leq J$. Le cardinal de $X \times Y$ est KJ.

Nous pouvons considérer cet ensemble comme un espace des epreuves et le munir d'une loi de probabilité, notée P_{XY}, appelée loi de probabilité jointe de X et Y. L'espace probabilisé joint ainsi défini est noté XY.

L'issue de l'expérience aléatoire est ici un couple, nous noterons x et y les variables aléatoires égales respectivement à la première et la seconde coordonnée de l'issue de l'expérience. La probabilité $P_{XY}(a_k, b_j)$ est donc la probabilité d'avoir simultanément $x = a_k$ et $y = b_j$.

La probabilité d'un évènement étant égale à la somme des probabilités des issues réalisant cet évènement, la probabilité de l'évènement $x = a_k$ est donc

$$P_X(a_k) = \sum_{j=1}^{J} P_{XY}(a_k, b_j).$$

Cela définit une loi de probabilité P_X sur X. De même la probabilité de l'évènement $y = b_j$ vaut

$$P_Y(b_j) = \sum_{k=1}^{K} P_{XY}(a_k, b_j).$$

Les deux lois de probabilité P_X et P_Y ainsi définies sont appelées lois marginales de P_{XY}.

Pour un espace probabilisé joint XY, avec $X = \{a_1, \ldots, a_K\}$ et $Y = \{b_1, \ldots, b_J\}$, la moyenne de la v.a. réelle v se définit par

$$\bar{v} = \sum_{k=1}^{K} \sum_{j=1}^{J} P(a_k, b_j) v(a_k, b_j).$$

b - Probabilité conditionnelle

On suppose que $P(a_k) > 0$, la probabilité conditionnelle pour que $y = b_j$ sachant que $x = a_k$, est définie par

$$P_{Y|X}(b_j \mid a_k) = \frac{P_{XY}(a_k, b_j)}{P_X(a_k)}.$$

De façon symétrique, nous définissons la probabilité conditionnelle de $x = a_k$ sachant $y = b_j$ par

$$P_{X|Y}(a_k \mid b_j) = \frac{P_{XY}(a_k, b_j)}{P_Y(b_j)}.$$

Les évènement $x = a_k$ et $y = b_j$ sont dit statistiquement indépendant si

$$P_{YX}(a_k, b_j) = P_X(a_k)P_Y(b_j).$$

Si cette égalité est vraie pour tout couple de XY, alors les espaces X et Y sont dit statistiquement indépendant. Nous parlerons alors d'espace probabilisé produit.

2.2 Une définition de l'information

2.2.1 Incertitude et information

a - Description qualitative et quantitative de l'information

Avant de trouver un moyen de mesurer quantitativement l'information, nous allons essayer de préciser le concept d'information.

Nous l'avons vu, la façon la plus appropriée de décrire un système de communication est d'en donner un modèle probabiliste. Fournir une information à un utilisateur consiste à choisir un évènement parmi plusieurs possibles. Qualitativement, fournir une information consiste donc à lever une incertitude sur l'issue d'une expérience aléatoire.

En fait, la notion d'information est déjà inhérente à celle de probabilité conditionnelle.

Considérons les évènements x et y. La probabilité conditionnelle $P(x \mid y)$ peut être interprétée comme la modification apportée à la probabilité $P(x)$ de l'évènement x lorsque l'on reçoit l'information que l'évènement y est réalisé.

L'information "y est réalisé" modifie la probabilité de x, c'est-à-dire l'incertitude sur la réalisation de x, de $P(x)$ à $P(x \mid y)$. Plus précisément,

– si $P(x \mid y) \leq P(x)$, l'incertitude sur x augmente,
– si $P(x \mid y) \geq P(x)$, l'incertitude sur x diminue.

Pour mesurer la variation de l'incertitude, il faudra choisir une fonction décroissante de la probabilité. Le logarithme permet d'exprimer commodément les variations d'incertitude, nous noterons $I(x)$ l'incertitude sur x, qui sera définie par

$$I(x) = -\log P(x).$$

Ainsi, l'information "y est réalisé" diminue l'incertitude sur x de la quantité

$$I(x) - I(x \mid y) = \log \frac{P(x \mid y)}{P(x)}.$$

Cette dernière quantité sera défine plus loin comme l'information mutuelle de x et y, alors que $I(x)$ sera l'information propre de x.

b - Choix du logarithme comme fonction de mesure de l'incertitude

Pour mesurer l'incertitude sur un évènement x, c'est-à-dire l'information fournie par la réalisation de x, nous désirons une fonction décroissante de $P(x)$. Le choix du logarithme n'est pas arbitraire ; en effet, il est souhaitable que la mesure d'information choisie soit additive. La quantité d'information fournie par la réalisation de deux évènements x et y statistiquement indépendants doit être égale à la somme des quantités d'information fournies par les réalisations de x et y pris séparément. On doit donc avoir $I(x, y) = I(x) + I(y)$, et puisque $P(x, y) = P(x)P(y)$ lorsque x et y sont indépendants, il est naturel d'utiliser le logarithme.

Nous devrons donc choisir une fonction de la forme $I(x) = \lambda \log P(x)$, avec $\lambda < 0$ pour assurer la décroissance par rapport à $P(x)$. Le choix de λ va dépendre de l'unité d'information que nous choisirons ; dans ce cours nous utiliserons le bit.

> *Un bit est égal à la quantité d'information fournie par le choix d'une alternative parmi deux équiprobables.*

En clair, cela signifie que si x est choisi dans l'espace des épreuves $\{0, 1\}$ muni d'une loi uniforme (i.e. $P(x = 0) = P(x = 1) = 1/2$), alors la quantité d'information fournie par la réalisation de l'évènement $x = 0$ (ou $x = 1$) est de 1 bit. On a

$$I(x = 0) = \lambda \log P(x = 0) = -\lambda \log 2 = 1,$$

donc $\lambda = -1/\log 2$ ce qui revient à choisir le logarithme en base 2 pour la définition de $I(x)$:

$$I(x) = -\log_2 P(x).$$

Le "bit" est à comprendre ici dans son sens originel de "binary unit", et non comme un symbole de l'alphabet $\{0, 1\}$. Cette confusion commune entre "binary unit" et "binary digit" s'explique par le fait que pour représenter une information de n bits, il faut généralement utiliser n symboles binaires.

Exemple : Soit une source dont l'aphabet de sortie $\{a_0, \ldots, a_{15}\}$ comprend 16 lettres equiprobables, c'est-à-dire que pour tout k, $0 \leq k < 16$, $P(a_k) = 1/16$. L'information propre de l'une de ces sorties a_k sera égale à $I(a_k) = -\log_2(1/16) = 4$ bits.

Dans ce cas particulier, l'information va consister à choisir un entier k dans {0,1,...,15}, et pour représenter cette information il faut disposer de 4 symboles binaires.

Il faut prendre garde que ce résultat n'est vrai que parceque les lettres de la source sont équiprobables, en effet, si ce n'est pas le cas, l'information propre de l'évènement a_k, $I(a_k) = -\log_2 P(a_k)$, sera généralement différente de 4, et nous verrons même plus loin que l'information propre moyenne peut être inférieure strictement à 4 bits.

2.2.2 Information mutuelle – Information propre

Nous considérons un espace probabilisé joint XY, avec $X = \{a_1, \ldots, a_K\}$ et $Y = \{b_1, \ldots, b_J\}$. Les variables aléatoires x et y sont associées respectivement aux espaces X et Y.

Nous désirons ici donner une mesure quantitative de ce que nous apporte la réalisation d'un évènement $y = b_j$ sur la possibilité de réalisation d'un autre évènement $x = a_k$. En termes mathématiques, l'occurence de l'évènement $y = b_j$, transforme la probabilité à priori $P(a_k)$ de l'évènement $x = a_k$ en la probabilité à postériori $P(a_k \mid b_j)$.

Définition 2.1 (Information mutuelle) *L'information mutuelle entre les évènements $x = a_k$ et $y = b_j$ est définie par*

$$I(a_k; b_j) = \log_2 \frac{P(a_k \mid b_j)}{P(a_k)}.$$

Remarquons que cette définiton est symétrique, en effet, on a par définition de la probabilité conditionnelle, $P(a_k, b_j) = P(a_k \mid b_j)P(b_j) = P(b_j \mid a_k)P(a_k)$, et donc

$$I(a_k; b_j) = I(b_j; a_k) = \log_2 \frac{P(a_k, b_j)}{P(a_k)P(b_j)}.$$

Discussion sur le signe de $I(x; y)$

- $I(x; y) > 0$ signifie que si l'un des deux évènements se réalise, alors la probabilité d'occurence de l'autre augmente.
- $I(x; y) < 0$ signifie que si l'un des deux évènements se réalise, alors la probabilité d'occurence de l'autre diminue.
- $I(x; y) = 0$ signifie que les deux évènements sont statistiquement indépendants.

Exemple : Considérons le canal binaire symétrique de probabilité de transition ϵ, avec des entrées a_1 et a_2 équiprobables (cf. Figure 2.1). Afin d'éviter une possible confusion, nous utiliserons les lettres a_1 et a_2 pour les entrées, et les lettres b_1 et b_2 pour les sorties au lieu des symboles binaires 0 et 1. Le canal binaire symétrique est défini par les probabilités conditionnelles $P(b_1 \mid a_1) = P(b_2 \mid a_2) = 1 - \epsilon$, $P(b_1 \mid a_2) = P(b_2 \mid a_1) = \epsilon$. Puisque $P(a_1) = P(a_2) = 1/2$, on en déduit

$$P(a_1, b_1) = P(a_2, b_2) = \frac{1 - \epsilon}{2} \quad \text{et} \quad P(a_1, b_2) = P(a_2, b_1) = \frac{\epsilon}{2},$$

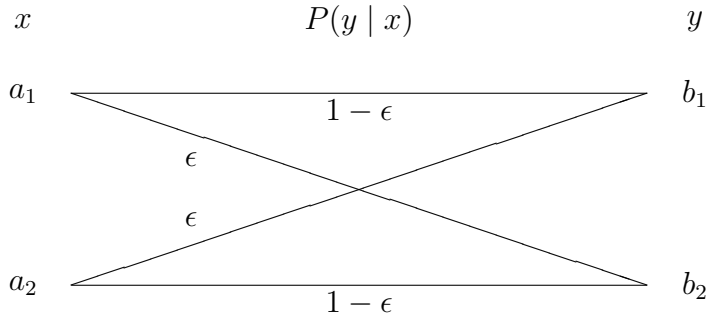


FIG. 2.1 – Canal binaire symétrique

et $P(b_1) = P(b_2) = 1/2$. Cela nous permet de calculer l'information mutuelle de chaque couple (a_k, b_j)

$$I(a_1; b_1) = I(a_2; b_2) = \log_2 2(1-\epsilon) \quad \text{et} \quad I(a_1; b_2) = I(a_2; b_1) = \log_2 2\epsilon.$$

On constate que pour $\epsilon < 1/2$, $I(a_1; b_1)$ est positif et $I(a_1; b_2)$ est négatif. Cela signifie que lorsqu'on observe à la sortie du canal la lettre b_1, la probabilité pour que a_1 ait été émise augmente. Et que au contraire si b_2 est observée, la probabilité pour que la lettre émise ait été a_1 diminue.

Enfin lorsque $\epsilon = 1/2$, toutes les informations mutuelles sont nulles, et donc les alphabets d'entrée et de sortie sont statistiquement indépendants, ce qui n'est évidemment pas souhaitable.

Considérons à présent un cas particulier intéressant ; quelle est l'information mutuelle entre l'évènement $x = a_k$ et lui-même ? Rigoureusement, cela consiste à calculer $I(a_k; b_j)$ lorsque l'évènement $y = b_j$ spécifie de façon unique l'évènement $x = a_k$, c'est-à-dire $P(a_k \mid b_j) = 1$; on a alors

$$I(a_k; b_j) = \log_2 \frac{P(a_k \mid b_j)}{P(a_k)} = \log_2 \frac{1}{P(a_k)}.$$

Il s'agit en fait de la quantité maximale d'information que peut fournir l'évènement $x = a_k$.

Définition 2.2 (Information propre) *L'information propre de l'évènement $x = a_k$ est définie par*

$$I(a_k) = -\log_2 P(a_k).$$

L'information propre s'interprète comme la "quantité d'information fournie par la réalisation d'un évènement".

Notons que l'information propre est toujours positive ou nulle, et que plus un évènement est improbable, plus son information propre est grande. À l'inverse, lorsque $P(a_k) = 1$, on a $I(a_k) = 0$, c'est-à-dire que la réalisation d'un évènement certain n'apporte aucune information, ce qui semble conforme à l'intuition.

On peut également définir dans l'espace probabilisé joint XY l'information propre conditionnelle qui est égale à la quantité d'information fournie par un évènement $x = a_k$ sachant que l'évènement $y = b_j$ s'est realisé.

Définition 2.3 *L'information propre conditionnelle de l'évènement $x = a_k$, sachant $y = b_j$ est définie par*

$$I(a_k \mid b_j) = -\log_2 P(a_k \mid b_j).$$

Cette dernière définition nous permet de donner une nouvelle interprétation de l'information mutuelle entre deux évènements. En effet d'après la relation

$$I(x; y) = I(x) - I(x \mid y), \tag{2.1}$$

l'information mutuelle entre x et y est égale à la quantité d'information fournie par x moins la quantité d'information que fournirait x lorsque y est réalisé.

2.3 Information mutuelle moyenne – Entropie

2.3.1 Définitions

Dans l'espace probabilisé joint XY, où $X = \{a_1, \ldots, a_K\}$ et $Y = \{b_1, \ldots, b_J\}$, l'information mutuelle peut être considérée comme une variable aléatoire réelle.

Définition 2.4 (Information mutuelle moyenne) *L'information mutuelle moyenne de X et Y dans l'espace probabilisé joint XY est définie par*

$$I(X; Y) = \sum_{k=1}^{K} \sum_{j=1}^{J} P(a_k, b_j) I(a_k; b_j) = \sum_{k=1}^{K} \sum_{j=1}^{J} P(a_k, b_j) \log_2 \frac{P(a_k, b_j)}{P(a_k) P(b_j)}.$$

On peut également définir la moyenne de l'information propre d'un espace probabilisé $X = \{a_1, \ldots, a_K\}$, cette moyenne porte le nom d'entropie.

Définition 2.5 (Entropie) *L'entropie d'un espace probabilisé X est défini par*

$$H(X) = \sum_{k=1}^{K} P(a_k) I(a_k) = \sum_{k=1}^{K} -P(a_k) \log_2 P(a_k).$$

Enfin, l'information propre conditionnelle est également une v.a. réelle et nous pouvons définir sa moyenne.

Définition 2.6 *L'entropie conditionnelle de X sachant Y dans l'espace probabilisé joint XY est définie par*

$$H(X \mid Y) = \sum_{k=1}^{K} \sum_{j=1}^{J} -P(a_k, b_j) \log_2 P(a_k \mid b_j).$$

L'équation (2.1) peut se reécrire en moyenne

$$I(X;Y) = H(X) - H(X \mid Y).$$

Cette relation nous sera utile pour interpréter la capacité d'un canal.

2.3.2 Propriétés de l'entropie

Théorème 2.1 *Soit X un espace probabilisé discret de cardinal K. Alors son entropie $H(X)$ vérifie*

$$H(X) \leq \log_2 K$$

avec égalité si et seulement si la loi de probabilité de X est uniforme.

preuve : La fonction $\log_2$ vérifie

$$\begin{array}{l} \log_2 z < (z-1)\log_2 e \quad \text{si } z > 0, z \neq 1 \\ \log_2 z = (z-1)\log_2 e \quad \text{si } z = 1 \end{array} \tag{2.2}$$

Nous allons montrer que $H(X) - \log_2 K \leq 0$. On a

$$\begin{aligned} H(X) - \log_2 K &= \sum_x P(x) \log_2 \frac{1}{P(x)} - \sum_x P(x) \log_2 K \\ &= \sum_x P(x) \log_2 \frac{1}{KP(x)}. \end{aligned}$$

En appliquant (2.2) à $z = 1/(KP(x))$, on obtient

$$\begin{aligned} H(X) - \log_2 K &\leq \log_2 e \sum_x P(x) \left(\frac{1}{KP(x)} - 1 \right) \\ &\leq \log_2 e \left(\sum_x \frac{1}{K} - \sum_x P(x) \right) = 0, \end{aligned}$$

avec égalité si et seulement si $1/(KP(x)) = 1$ pour tout x.

L'inégalité est donc vérifiée, et il y a égalité si et seulement si $P(x) = 1/K$ pour tout x, c'est à dire si la loi de probabilité de X est uniforme. □

Cette proposition est fondamentale, en effet c'est elle qui nous dit que l'entropie d'un espace probabilisé de taille 2^L muni d'une loi uniforme est égale à L bits.

Nous verrons dans le chapitre sur le codage de source que cela nous donne une borne inférieure sur le nombre de symboles binaires nécessaires pour coder une source discrète à l'aide d'un code de taille fixe. Cette borne sera égale au plus petit entier supérieur ou égal à $\log_2 K$.

Théorème 2.2 *Soit un espace probabilisé joint discret XY. L'information mutuelle moyenne $I(X;Y)$ de X et de Y vérifie*

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

avec égalité si et seulement si X et Y sont statistiquement indépendants.

preuve : Montrons que $-I(X;Y) \leq 0$. On a, en utilisant (2.2)

$$\begin{aligned} -I(X;Y) &= \sum_{x,y} P(x,y) \log_2 \frac{P(x)P(y)}{P(x,y)} \\ &\leq \log_2 e \sum_{x,y} P(x,y) \left(\frac{P(x)P(y)}{P(x,y)} - 1 \right) \\ &\leq \log_2 e \left(\sum_{x,y} P(x)P(y) - \sum_{x,y} P(x,y) \right) \\ &\leq \log_2 e \left(\sum_{x} P(x) \sum_{y} P(y) - 1 \right) = 0, \end{aligned}$$

et il y a égalité si et seulement si $P(x)P(y)/P(x,y) = 1$ pour tout couple (x,y).

L'inégalité est donc prouvée et puisque x et y sont indépendants si et seulement si $P(x,y) = P(x)P(y)$, l'egalité $I(X;Y) = 0$ est vraie si et seulement si X et Y sont statistiquement indépendants. □

Ce résultat signifie essentiellement que en moyenne le fait de connaitre la valeur de y dans Y diminue toujours l'incertitude sur X, sauf si X et Y sont indépendants, auquel cas aucune information n'est apportée.

2.4 Le cas continu

2.4.1 Espaces probabilisés continus

Les notions d'information mutuelle et d'information propre peuvent également être définie dans le cas continu. Il faut prendre en compte le fait que l'espace des épreuves est continu. En pratique on le prendra égal à l'intervalle $X =]-\infty, +\infty[$, et la loi de probabilité sur X sera définie par une densité, donnée par une fonction réelle notée $p(x)$, dont l'intégrale sur X vaut 1.

Enfin comme on peut le voir dans la Table 2.1, la moyenne d'une variable aléatoire réelle se calcule à l'aide d'une intégrale, au lieu d'être une somme sur un ensemble discret.

De même nous pouvons définir un ensemble probabilisé joint XY continu à l'aide d'une densité de probabilité $p_{XY}(x,y)$ à deux variables. Comme dans le cas des lois discrètes,

		cas discret	cas continu
espace des épreuves	X	$\{a_1, \ldots, a_K\}$	$]-\infty, +\infty[$
loi de probabilité	P_X	$P(a_1), \ldots, P(a_K)$ $\sum_{k=1}^{K} P(a_k) = 1$	$p(x),\ x \in]-\infty, +\infty[$ $\int_{-\infty}^{+\infty} p(x)dx = 1$
moyenne d'une v.a. réelle	$\bar{v}$	$\sum_{k=1}^{K} v(a_k)P(a_k)$	$\int_{-\infty}^{+\infty} v(x)p(x)dx$

TAB. 2.1 – Correspondance entre espaces probabilisés discret et continu

nous définissons les lois marginales sur X et Y par

$$\begin{aligned} p_X(x) &= \int_{-\infty}^{\infty} p(x,y)dy, \\ p_Y(y) &= \int_{-\infty}^{\infty} p(x,y)dx, \end{aligned}$$

ainsi que les densités de probabilités conditionnelles

$$p_{Y|X}(y \mid x) = \frac{p_{XY}(x,y)}{p_X(x)},$$

et

$$p_{X|Y}(x \mid y) = \frac{p_{XY}(x,y)}{p_Y(y)},$$

chaque fois que les dénominateurs seront non nuls.

2.4.2 Entropie et information dans le cas continu

À l'aide des densités de probabilités, nous sommes à même de définir par analogie avec le cas discret l'information mutuelle et l'information propre dans le cas continu.

Définition 2.7 *Soit XY un espace probabilisé joint continu.* L'information mutuelle *entre x et y est définie par*

$$I(x;y) = \log_2 \frac{p(x,y)}{p(x)p(y)}.$$

L'information propre *de x est définie par*

$$I(x) = -\log_2 p(x).$$

L'information propre conditionnelle *de x sachant y est définie par*

$$I(x) = -\log_2 p(x \mid y).$$

De même, $I(x, y)$ et $I(x)$ et $I(x \mid y)$ pouvant être considérée comme des v.a. réelles, on peut calculer leur moyenne.

Définition 2.8 *Soit XY un espace probabilisé joint continu.* L'information mutuelle moyenne *entre X et Y est définie par*

$$I(X;Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(x,y)\log_2 \frac{p(x,y)}{p(x)p(y)}dxdy.$$

L'entropie *de X est définie par*

$$H(X) = \int_{-\infty}^{\infty} -p(x)\log_2 p(x)dx.$$

L'entropie conditionnelle *de X sachant Y est définie par*

$$H(X \mid Y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} -p(x,y)\log_2 p(x \mid y).$$

Chapitre 3

Codage des sources discrètes

Nous nous intéresserons dans ce chapitre au codage de la sortie d'une source discrète sans mémoire en une séquence binaire. Ce codage devra permettre de retrouver la séquence de lettre de la source à partir du codage binaire. Nous verrons que le nombre minimale moyen de symboles binaires par lettre est égal à l'entropie de la source.

Dans de nombreux exemples pratique de codage de source, comme le code Morse, les lettres les plus fréquentes se voient attribuer les mots de code les plus courts. En Morse, la lettre "e" est représentée par le mot "·", alors que la lettre "q", beaucoup moins fréquente, est codée par le mot "$\cdot\cdot--$". De tels code sont dits de longueur variable. Cependant, ces codes peuvent amener des problèmes d'attente dans les applications où les mots de codes doivent être transmis à un rythme régulier. Nous devrons donc également considérer le cas des codes de longueur fixe, bien qu'ils ne permettent pas toujours, comme nous le verrons, un codage optimal.

3.1 Les différents types de codage de source

Dans tout le chapitre nous considérerons une source discrète dont l'alphabet est $X = \{a_1, \ldots, a_K\}$ et dont la loi de probabilité $P(a_1), \ldots, P(a_k)$ est donnée.

3.1.1 Terminologie

Nous considérons une source discrète qui émet à intervalle de temps régulier des lettres choisie dans son alphabet à l'aide d'une loi de probabilité. Le codeur émet à intervalle de temps réguliers des symboles binaires.

Nous désignerons par X^l l'ensemble des l-uplets de lettres de X et par $X^* = \bigcup_{l\geq 1} X^l$ l'ensemble des séquences finies de lettres de X. Nous désignerons par $\{0,1\}^*$ l'ensemble des séquences binaires finies.

Définition 3.1 *Un* codage *d'une source discrète est une procédure qui associe à chaque séquence finie de lettres de la source une séquence binaire finie.*

Un codage est donc une application de X^* dans $\{0,1\}^*$.

Définition 3.2 *Un* code *d'une source discrète est une procédure qui associe à chaque lettre de la source en une séquence binaire appelée* mot de code.

Un code est donc une application de X dans $\{0,1\}^*$, qui à toute lettre a_k de X associe un mot de code m_k. À un code donné, on peut associer le codage

$$\begin{array}{ccc} X^* & \rightarrow & \{0,1\}^* \\ (a_{k_1},\ldots,a_{k_l}) & \mapsto & m_{k_1}\odot\ldots\odot m_{k_l} \end{array} \tag{3.1}$$

où le symbole $\odot$ représente la concaténation.

Définition 3.3 (Code régulier) *Un code sera dit* régulier *si deux lettres distinctes sont codées à l'aide de deux mots de code distincts.*

Dans toute la suite nous ne considerons que des codes réguliers. De plus pour un code régulier le terme de code sera également utilisé pour désigner l'ensemble des mots de code.

Définition 3.4 *L'efficacité d'un code d'une source X d'entropie $H(X)$ est définie par*

$$E = \frac{H(X)}{\bar{n}},$$

où $\bar{n}$ est le nombre moyen de symboles binaires utilisés par lettre de la source, défini par

$$\bar{n} = \sum_{k=1}^{K} n_k P(a_k),$$

où n_k est la longueur du mot de code associé à a_k.

Nous verrons que l'efficacité d'un code régulier ne peut excéder 1.

Efficacité d'un codage Pour définir l'efficacité d'un codage, il faut définir le nombre de symboles moyen utilisés pour représenter une lettre de la source.

Soit $N_L(a_{k_1},\ldots,a_{k_L})$ la longueur de la séquence binaire codant le L-uplet $(a_{k_1},\ldots,a_{k_L})$, *puisque la source est sans mémoire* la probabilité d'avoir ce L-uplet plutôt qu'un autre L-uplet est le produit $P(a_{k_1})\ldots P(a_{k_L})$. La longueur moyenne d'un L-uplet est alors

$$\bar{N}_L = \sum_{k_1=1}^{K}\ldots\sum_{k_L=1}^{K} P(a_{k_1})\ldots P(a_{k_L})N_L(a_{k_1},\ldots,a_{k_L}),$$

le nombre de symboles binaire utilisé par chaque lettre d'un L-uplet est donc égale à

$$\bar{n}_L = \frac{\bar{N}_L}{L}.$$

Le nombre moyen de symboles utilisés pour représenter une lettre de la source est défini par

$$\bar{n} = \lim_{L\to\infty} \bar{n}_L,$$

si cette limite existe.

L'efficacité d'un codage d'une source X peut alors être définie comme pour un code par

$$E = \frac{H(X)}{\bar{n}},$$

où $H(X)$ est l'entropie de X.

Définie ainsi, l'efficacité d'un codage associé à un code par (3.1) est égale à l'efficacité de ce code.

3.1.2 Codes de longueur fixe

Définition 3.5 (Code de longueur fixe) *Un code de longueur fixe est un code dont tous les mots de code ont la même longueur. Si cette longueur est n, nous dirons que le code est de longueur n.*

Proposition 3.1 *Soit une source X dont l'alphabet a pour cardinal K. Il existe un code régulier de X de longueur n telle que*

$$\log_2 K \leq n < 1 + \log_2 K. \tag{3.2}$$

De plus, il n'existe aucun code régulier de longueur $n < \log_2 K$.

preuve : Soit la source $X = \{a_1, \ldots, a_K\}$, si nous choisissons de coder a_k par l'écriture de k en base 2, cela nécessitera n symboles binaires, où n sera le plus petit entier tel que $2^n \geq K$. On en déduit facilement (3.2) en prennant le logarithme.

Réciproquement, soit $n < \log_2 K$, l'ensemble des séquences binaires de longueur n a pour cardinal $2^n < K$, il est donc exclus de faire correspondre à chaque élément de X une séquence différente. □

L'efficacité d'un code de longueur n est égale à

$$E = \frac{H(X)}{n},$$

d'après le Théorème 2.1, on a $H(X) \leq \log_2 K$, donc d'après (3.2), on a $E \leq 1$. De plus $E = 1$ n'est possible qu'à deux conditions :

1. $H(X) = \log_2 K$, c'est-à-dire que les lettres de la source sont équiprobables.
2. $n = \log_2 K$, c'est-à-dire que le cardinal de la source est une puissance de 2.

Exemple : Soit une source dont l'aphabet de sortie est l'ensemble des chiffres décimaux $X = \{0, 1, \ldots, 9\}$ muni de la loi de probabilité uniforme. Le code de longueur fixe d'une telle source a une longueur au moins 4. Par exemple

lettre	0	1	2	3	4	5	6	7	8	9
mot de code	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

L'efficacité de ce code est égale à $H(X)/4 = (\log_2 10)/4 \approx 0,83$. Il est clair que ce code n'est pas optimal, puisque six mots binaires de longueur 4 sont inutilisés, donc ce code pourrait être utilisé pour une source ayant un cardinal 16.

Il est cependant possible d'améliorer l'efficacité du codage en considérant non plus des chiffres isolés mais des paires de chiffres. Ainsi la source peut être vue comme ayant pour alphabet l'ensemble $X^2 = \{00, 01, \ldots, 99\}$ de cardinal 100. Cette source reste munie d'une loi de probabilité uniforme, et son entropie vaut $H(X^2) = \log_2 100 = 2H(X)$.

La puissance de 2 immédiatement supérieure à 100 est $2^7 = 128$. Il existe don un code régulier de X^2 de longueur 7. L'efficacité de ce code est cette fois égal à $H(X^2)/7 = (2\log_2 10)/7 \approx 0,95$ ce qui est meilleur.

En considérant la source X^3 de 1000 lettres, codées en 10 symboles binaires, on obtient une efficacité de $0,996$.

D'une façon générale il est possible de considérer la source X^L des L-uplets de lettres de X, le cardinal de cette source est K^L, et son entropie est $H(X^L) = LH(X)$.

Proposition 3.2 *Soit X une source de cardinal K, soit X^L la source dont l'alphabet est l'ensemble des L-uplets d'éléments de X. Il existe un code régulier de X^L de longueur N telle que*

$$\log_2 K \leq \frac{N}{L} < \frac{1}{L} + \log_2 K. \tag{3.3}$$

preuve : D'après la Proposition 3.1 il existe un code de la source X^L de longueur N telle que

$$\log_2 K^L \leq N < 1 + \log_2 K^L,$$

car le cardinal de X^L est K^L. On en déduit (3.3). □

Pour tout L on pose $\bar{n}(L) = N/L$, où N est la longueur optimale du code donnée par la Proposition 3.2. L'efficacité d'un tel code vaut

$$E(L) = \frac{H(X^L)}{N} = \frac{LH(X)}{N} = \frac{H(X)}{\bar{n}(L)},$$

D'après (3.3) on a

$$\lim_{L\to\infty} \bar{n}(L) = \log_2 K,$$

nous en déduisons que

$$\lim_{L\to\infty} E(L) = \frac{H(X)}{\log_2 K}.$$

Ceci montre que si $H(X) < \log_2 K$, il est impossible de s'approcher d'un codage optimal avec un code régulier. Par contre pour une source munie d'une loi de probabilité uniforme, donc telle que $H(X) = \log_2 K$, l'efficacité du codage peut être arbitrairement proche de 1.

3.1.3 Codes de longueur variable

a - Codes séparables – Codes irréductibles

Il existe deux méthodes aisées pour obtenir un code permettant de séparer deux mots de code consécutifs :

1. Utiliser un code de longueur fixe n, auquel cas la séquence binaire reçue est découpée en blocs de n symboles binaires qui seront décodés séparément.
2. Utiliser, comme pour le Morse, un symbole supplémentaire entre deux mots. Dans un tel cas, tout se passe comme si l'alphabet de sortie du codeur était augmenté d'une unité. Ainsi le code Morse peut être considéré comme un code ternaire et non binaire. Nous traiterons ces codes comme une généralisation des codes binaires.

Le cas des *codes de longueur variable* est plus problématique en ce qui concerne le découpage des mots. Puisque deux mots consécutifs ont à priori des longueurs différentes, il faut concevoir des codes permettant la séparation sans ambiguïté des mots de code.

Définition 3.6 (Code déchiffrable) *Un codage injectif est appelé* déchiffrable.

Un code dont le codage associé est injectif est dit déchiffrable.

Autrement dit, une source $X = \{a_1, \ldots, a_k\}$, muni d'un code tel que $a_k \mapsto m_k$ pour tout k, est dit déchiffrable si l'application

$$\begin{array}{ccc} \bigcup_{l\geq 1} X^l & \to & \{0,1\}^* \\ (a_{k_1}, \ldots, a_{k_l}) & \mapsto & m_{k_1} \odot \ldots \odot m_{k_l} \end{array}$$

réalisant le codage de toutes les séquences de lettres de la source est injective.

Cette définition signifie simplement qu'il existe au plus une séquence de lettres de la source ayant permis d'obtenir une séquence binaire donnée.

Il existe une condition suffisante, dite du prefixe, pour qu'un code soit déchiffrable.

Condition du préfixe. *Un code vérifie la condition du préfixe si aucun mot de code n'est le début d'un autre mot de code.*

Définition 3.7 (Code irréductible) *Un code est dit* irréductible *s'il vérifie la condition du préfixe.*

Proposition 3.3 *Tout code irréductible est déchiffrable.*

Comme le montre l'exemple suivant, la condition du préfixe est une condition suffisante mais non nécessaire pour qu'un code soit déchiffrable.

Exemple : La source $\{a_1, a_2, a_3\}$ est codée par $a_1 \to 0$, $a_2 \to 10$, $a_3 \to 100$. Les mots de codes peuvent être découpés en introduisant une séparation avant chaque "1" dans la séquence binaire, ce code est donc déchiffrable. Il n'est pourtant pas irréductible.

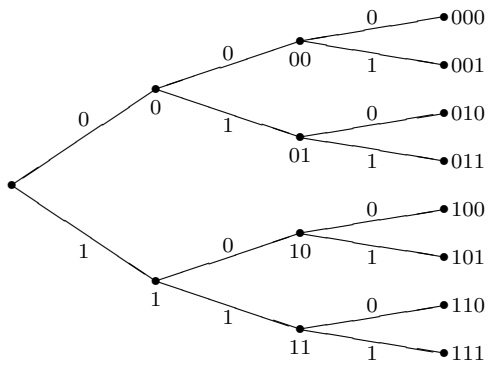
b - Représentation des codes irréductibles par des arbres

Il est commode de représenter un code à l'aide d'un arbre binaire. Nous ne définirons pas ici les termes *branche, nœud, feuille, raçine, fils, père*, qui auront leur signification usuelle. Les arbres que nous considérons ont les propriétés suivantes :

- chaque branche a pour attribut un des symboles binaires 0 ou 1,
- deux branches partant du même nœud ont des attributs différents,
- chaque nœud a pour attribut la concaténation des attributs des branches reliant la raçine à ce nœud.

De plus, nous appellerons *ordre* d'un nœud ou d'une feuille le nombre de branches le séparant de la raçine.

Exemple : Pour représenter l'ensemble des séquences binaire de longueur inférieure ou égale à 3 nous utilisons l'arbre :

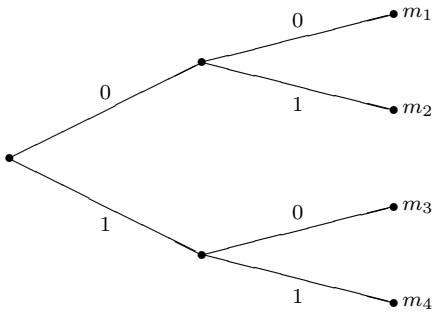


Pour représenter un code régulier, nous utiliserons le plus petit arbre contenant tous les mots de code, appelé *arbre du code*.

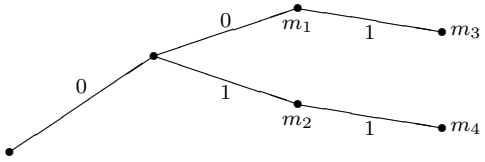
Exemple : Considérons une source d'alphabet $\{a_1, a_2, a_3, a_4\}$, et les trois codage suivants

	code 1	code 2	code 3
a_1	$m_1 = 00$	$m_1 = 00$	$m_1 = 0$
a_2	$m_2 = 01$	$m_2 = 01$	$m_2 = 10$
a_3	$m_3 = 10$	$m_3 = 001$	$m_3 = 110$
a_4	$m_4 = 11$	$m_4 = 011$	$m_4 = 111$

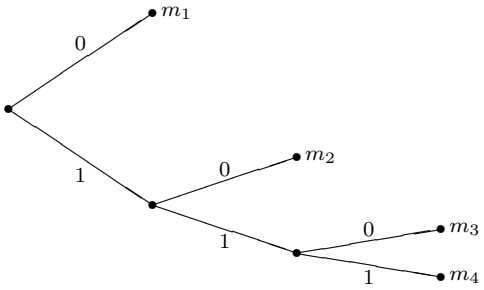
L'arbre du code 1 est donné par



L'arbre du code 2 est donné par



Et l'arbre du code 3 est donné par



Nous constatons que pour les codes 1 et 3, qui sont irréductibles, les mots de code sont exactement les feuilles de l'arbre, alors que pour le code 2 qui n'est pas irréductible, les mots de code m_1 et m_2 sont "à l'intérieur" de l'arbre.

Il existe une caractérisation simple des arbres des codes irréductibles :

Proposition 3.4 *Un code est irréductible si et seulement si les feuilles de son arbre sont exactement ses mots de code.*

preuve : Dire qu'un mot de code est une feuille est équivalent à dire qu'il n'est pas le préfixe d'un autre mot de code. □

Dans un arbre binaire, chaque nœud peut avoir zéro, un ou deux fils. Les arbres dont aucun nœud n'a un seul fils ont une importance particulière.

Définition 3.8 *On appelle* arbre binaire strict *un arbre dont tous les nœuds ont zéro ou deux fils.*

Lemme 3.1 *Soit un arbre binaire dont les K feuilles ont pour ordre $n_1, \ldots, n_k$. Alors*

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

Et si l'arbre est binaire strict, on a

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

preuve : Nous définissons le poids de chaque nœud de l'arbre de la façon suivante :
- le poids d'une feuille d'ordre i est égal à 2^{-i},
- le poids d'un nœud qui n'est pas une feuille est la somme des poids de ses fils.

Montrons par récurrence descendante sur l'ordre des nœuds qu'un nœud d'ordre i a pour poids au plus 2^{-i}.

Supposons que tous les nœuds d'ordre $i+1$ ont un poids $\leq 2^{-(i+1)}$. Un nœud d'ordre i est soit une feuille et son poids est égal à 2^i, soit n'est pas une feuille et admet au plus deux fils de poids $2^{-(i+1)}$. Dans tous les cas sont poids est au plus 2^i.

Il reste à vérifier qui existe un entier i_0 tel que la propriété soit vraie. Soit i_0 l'ordre maximal des nœud de l'arbre, tout nœud d'ordre i_0 est une feuille et a donc un poids 2^{i_0}.

Donc la raçine de l'arbre qui est un nœud d'ordre 0 a un poids ≤ 1.

D'autre part, il est clair que le poids de la raçine de l'arbre est égal à la somme des poids de ces feuilles. On en déduit l'inégalité recherchée.

Dans le cas d'un arbre binaire strict la démonstration est la même, mais chaque nœud ayant exactement deux fils s'il n'est pas une feuille, il a un poids exactement égal à 2^{-i}, où i est son ordre. □

Théorème 3.1 (de Kraft) *Il existe un code irréductible de K mots de longueurs $n_1, \ldots, n_K$ si et seulement si l'inégalité*

$$\sum_{k=1}^{K} 2^{-n_k} \leq 1, \tag{3.4}$$

est satisfaite.

preuve :

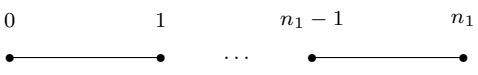
1. Soit un code irréductible de K mots de longueurs $n_1, \ldots, n_K$, les mots de ce code sont exactement les feuilles de son arbre.

   D'autre part la longueur d'un mot est exactement égale à l'ordre du nœud le représentant dans l'arbre.

   Donc à l'aide du Lemme 3.1, on a (3.4).

2. Soit des entiers $1 \leq n_1 \leq \ldots \leq n_K$ vérifiant (3.4), montrons par réccurence sur K qu'il existe un arbre binaire dont les K feuilles ont pour ordre $n_1, \ldots, n_K$.

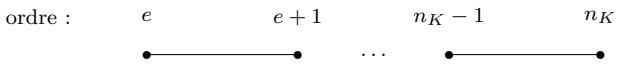
   (a) Si $K = 1$, l'arbre unaire de profondeur n_1 convient.



(b) Supposons que l'on ait construit un arbre binaire tel que ses $K-1$ feuilles ait pour ordre $n_1, \ldots, n_{K-1}$, d'après (3.4), et puisque $2^{-n_K} > 0$, on a

$$\sum_{k=1}^{K-1} 2^{-n_k} < 1,$$

et donc l'arbre à $K-1$ feuilles décrit ci-dessus n'est pas binaire strict. Il existe donc un nœud ayant exactement un fils. L'ordre e de ce nœud est inférieur strictement à n_K par construction, on ajoute donc à ce nœud un second fils égal au sous arbre



De cette façon une seule feuille est ajoutée d'ordre n_K.

D'après la Proposition 3.4 cela suffit à montrer qu'il existe un code irréductible de K mots de longueur $n_1, \ldots, n_K$.

□

Il existe un résultat similaire plus fort du à Mac Millan que nous ne démontrerons pas.

Théorème 3.2 (de Mac Millan) *Il existe un code* **déchiffrable** *de K mots de longueurs $n_1, \ldots, n_K$ si et seulement si l'inégalité*

$$\sum_{k=1}^{K} 2^{-n_k} \leq 1, \tag{3.5}$$

est satisfaite.

Il est essentiel de noter que le Théorème 3.1 comme le Théorème 3.2 sont non constructifs. Ils nous donnent un résultat sur l'existence de codes dont les longueurs des mots de codes vérifient (3.5), mais ne prétendent pas que tout code dont les longueurs vérifient l'inégalité (3.5) est irreductible (ou déchiffrable).

3.2 Le premier théorème de Shannon

Soit une source $X = \{a_1, \ldots, a_K\}$ munie d'une loi de probabilité $P(a_1), \ldots, P(a_K)$.

Théorème 3.3 *1. Pour toute source d'entropie H codée au moyen d'un code déchiffrable de longueur moyenne $\bar{n}$, on a*

$$\bar{n} \geq H.$$

2. Pour toute source d'entropie H, il existe un code irréductible dont la longueur moyenne $\bar{n}$ est telle que

$$H \leq \bar{n} < H + 1. \tag{3.6}$$

preuve :

1. Soit la source $X = \{a_1, \ldots, a_K\}$ muni d'un code déchiffrable dont les mots ont une longueur respectivement de $n_1, \ldots, n_K$. Montrons que $H(X) - \bar{n} \leq 0$.

$$\begin{aligned} H(X) - \bar{n} &= \sum_{k=1}^{K} P(a_k) \log_2 \frac{1}{P(a_k)} - \sum_{k=1}^{K} P(a_k) n_k \\ &= \sum_{k=1}^{K} P(a_k) \left(\log_2 \frac{1}{P(a_k)} + \log_2 2^{-n_k} \right) \\ &= \sum_{k=1}^{K} P(a_k) \log_2 \frac{2^{-n_k}}{P(a_k)} \end{aligned}$$

   D'après (2.2) page 16, nous en déduisons

$$H(X) - \bar{n} = (\log_2 e) \left(\sum_{k=1}^{K} 2^{-n_k} - \sum_{k=1}^{K} P(a_k) \right) \leq 0$$

   en appliquant le Théorème 3.2 de Mac Millan.

2. Nous cherchons à présent à montrer qu'il existe un code irréductible vérifiant (3.6). Pour tout k, $1 \leq k \leq K$, soit n_k l'unique entier vérifiant

$$2^{-n_k} \leq P(a_k) < 2^{-n_k+1}. \tag{3.7}$$

   En sommant l'inégalité de gauche on obtient

$$\sum_{k=1}^{K} 2^{-n_k} \leq 1,$$

   ce qui nous assure d'après le Théorème 3.1 de Kraft qu'il existe un code irréductible dont les mots ont pour longueurs $n_1, \ldots, n_K$.
   En prennant le logarithme de l'inégalité de droite de (3.7), nous obtenons

$$\log_2 P(a_k) < -n_k + 1,$$

   soit encore

$$n_k < \log_2 \frac{1}{P(a_k)} + 1.$$

   En passant à la moyenne, nous obtenons

$$\bar{n} < H(X) + 1.$$

   Donc il existe pour la source X un code irréductible dont la longueur moyenne des mots $\bar{n}$ vérifie (3.6).

□

L'efficacité $E = H(X)/\bar{n}$ d'un code irréductible dont la longueur moyenne vérifie (3.6) sera telle que

$$1 - \frac{1}{H(X)+1} < E \leq 1.$$

Cela nous montre que l'efficacité ne peut pas exceder 1. Cela ne suffit pourtant à prouver l'existence de codages dont l'efficacité s'approche arbitrairement de 1.

Pour obtenir de tels codages, il sera nécessaire, comme pour les codes de longueur fixe de considérer la source X^l des l-uplets de lettres de X. L'entropie de cette source vaut $lH(X)$.

Théorème 3.4 *Soit un source discrète sans mémoire X d'entropie $H(X)$. Pour tout entier $l \geq 1$, il existe un code irréductible de X^l dont la longueur moyenne $\bar{N}$ vérifie*

$$H(X) \leq \frac{\bar{N}}{l} < \frac{1}{l} + H(X). \tag{3.8}$$

preuve : Soit un entier $l \geq 1$. Puisque la source X est sans mémoire, la loi de probabilité de X^l est la loi produit, c'est-à-dire

$$P(a_{i_1}, \ldots, a_{i_l}) = \prod_{m=1}^{l} P(a_{i_m}).$$

La source X^l a donc pour entropie $H(X^l) = lH(X)$.

D'après le Théorème 3.3, il existe un code irréductible de X^l de longueur $\bar{N}$ telle que

$$H(X^l) \leq \bar{N} < 1 + H(X^l).$$

On en déduit (3.8). □

Théorème 3.5 (Premier théorème de Shannon) *Pour toute source discrète sans mémoire, il existe un codage déchiffrable dont l'efficacité est arbitrairement proche de 1.*

preuve : En terme mathématiques le théorème s'énonce :

> *Pour toute source discrète sans mémoire, pour tout réel $\epsilon > 0$, il existe un codage déchiffrable dont l'efficacité est strictement supérieure à $1 - \epsilon$.*

Soit X la source, considèrons la source X^l pour un entier l quelconque. d'après le Théorème 3.4 il existe un code irréductible de X^l, dont la longueur moyenne $\bar{N}_l$ vérifie

$$H(X) \leq \frac{\bar{N}_l}{l} < \frac{1}{l} + H(X). \tag{3.9}$$

Nous allons utiliser de code pour réaliser le codage suivant.

Soit l fixé.Pour coder $L = ml + r$ symboles, où $0 < r \leq l$, nous allons coder m l-uplets de lettres à l'aide du code de X^l et le dernier bloc de r lettres sera complété par $l - r$ symboles arbitraires puis codé à l'aide du même code.

Ce codage va donc nécessiter en moyenne $(m+1)\bar{N}_l$ symboles binaires pour les L lettres, soit une moyenne $\bar{n}_L$ par lettre vérifiant

$$\frac{\bar{N}_l}{l} \leq \bar{n}_L < \frac{m+1}{m}\frac{\bar{N}_l}{l}. \tag{3.10}$$

L'efficacité de ce codage est égale par définition à

$$E = \frac{H(X)}{\bar{n}},$$

où $\bar{n} = \lim_{L\to\infty} \bar{n}_L$ si cette limite existe. D'après (3.10) on a

$$\bar{n} = \lim_{L\to\infty} \bar{n}_L = \frac{\bar{N}_l}{l}.$$

Et d'après (3.9)

$$\frac{1}{E} = \frac{\bar{n}}{H(X)} < 1 + \frac{1}{lH(X)}.$$

Choisissons l tel que $l \geq 1/(\epsilon H(X))$, on aura alors

$$\frac{1}{E} < 1 + \epsilon,$$

donc

$$E > \frac{1}{1+\epsilon} \geq 1 - \epsilon.$$

□

3.3 Une procédure optimale de codage

Dans cette section, lorsque nous parlerons du code d'une source X, les mots de codes associés aux lettres $a_1, .., a_k$ de la source seront noté respectivement $m_1, \ldots, m_k$.

3.3.1 Définition du code de Huffman

Le code de Huffmann d'une source X de cardinal K est un code irréductible qui se définit récursivement sur le cardinal de la source. Le code de Huffman de X sera construit à partir du code de Huffman d'une source de cardinal $K - 1$.

1. $K = 2$. Soit la source $X = \{a_1, a_2\}$, le code irréductible $\{m_1 = 0, m_2 = 1\}$ est le code de Huffman de X.

2. $K > 2$. Soit la source $X = \{a_1, \ldots, a_K\}$ dont on réordonne les lettres de telle façon que $P(a_1) \geq \ldots \geq P(a_K)$. Soit $\{m'_1, \ldots, m'_{K-1}\}$ le code de Huffman de la source $X' = \{a'_1, \ldots, a'_{K-1}\}$ munie de la loi de probabilité P' telle que

$$\begin{cases} P'(a'_k) &= P(a_k) & \text{si } k \leq K-2 \\ P'(a'_{K-1}) &= P(a_{K-1}) + P(a_K) \end{cases}$$

Le code de Huffman de X sera alors défini par

$$\begin{cases} m_k &= m'_k & \text{si } k \leq K-2 \\ m_{K-1} &= m'_{K-1} \odot 0 \\ m_K &= m'_{K-1} \odot 1 \end{cases}$$

où $\odot$ représente la concaténation.

Si le code $\{m'_1, \ldots, m'_{K-1}\}$ verifie la condition du préfixe, alors le code $\{m_1, \ldots, m_K\}$ la vérifiera également. Donc le code de Huffman d'une source de cardinal K est irréductible.

3.3.2 Le code de Huffman est optimal

Définition 3.9 *Un code déchiffrable de longueur moyenne $\bar{n}$ d'une source est dit* optimal *s'il n'existe auncun code déchiffrable de cette source dont la longueur moyenne des mots de code est strictement inférieure à $\bar{n}$.*

Proposition 3.5 *Le code de Huffman d'une source est optimal.*

3.3.3 Exemples

Chapitre 4

Canaux discrets sans mémoire

Dans le chapitre précédent sur le codage de source, aucun élément extérieur ne venait modifier l'information. Au contraire lorsque l'information va transité dans le canal, elle sera pertubée par un bruit.

Le résultat principal de ce chapitre nous assure qu'il est possible de coder l'information de façon à ce que la détérioration soit négligeable. Cela se fera bien entendu au prix d'une redondance de l'information.

En fait pour tout canal de transmission, nous définirons une grandeur caractéristique appelée *capacité* et qui s'interprète comme la quantité maximale d'information pouvant transiter à travers le canal.

4.1 Généralités

4.1.1 Définition du canal discret

Pour définir un canal de transmission, il est nécessaire de décrire l'ensemble des entrées et des sorties possibles du canal, ainsi que le bruit qui perturbera la transmission.

Le canal discret est le modèle le plus simple de canal de transmission. L'ensemble des entrées comme celui des sorties seront des ensembles finis X et Y et le bruit se modélisera par la donnée d'une loi de probabilité conditionnelle de Y sachant X.

Définition 4.1 (Canal discret sans mémoire) *Un* canal discret sans mémoire *est défini par la donnée de :*

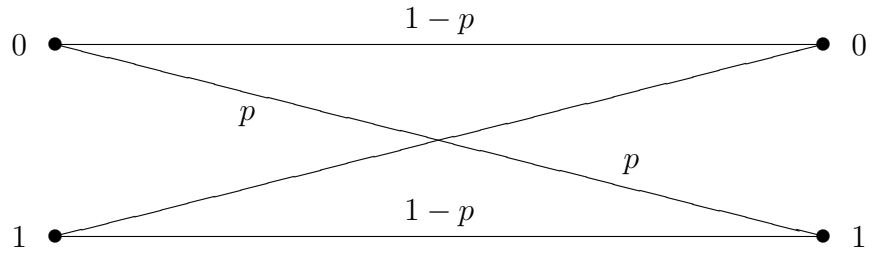
1. *Un alphabet d'entrée* $X = \{a_1, \ldots, a_K\}$.
2. *Un alphabet de sortie* $Y = \{b_1, \ldots, b_J\}$.
3. *Une loi de transition définies par les probabilité conditionnelles* $P(b_j \mid a_k)$.

La matrice $K \times J$ *suivante*

$$\Pi = \begin{pmatrix} P(b_1 \mid a_1) & \ldots & P(b_J \mid a_1) \\ \vdots & \ddots & \vdots \\ P(b_1 \mid a_K) & \ldots & P(b_J \mid a_K) \end{pmatrix}$$

est appelée matrice stochastique *du canal. Nous parlerons d'un canal* (X, Y, Π).

Exemple : Le canal binaire symétrique de probabilité de transition p représenté par le diagramme



a pour matrice stochastique

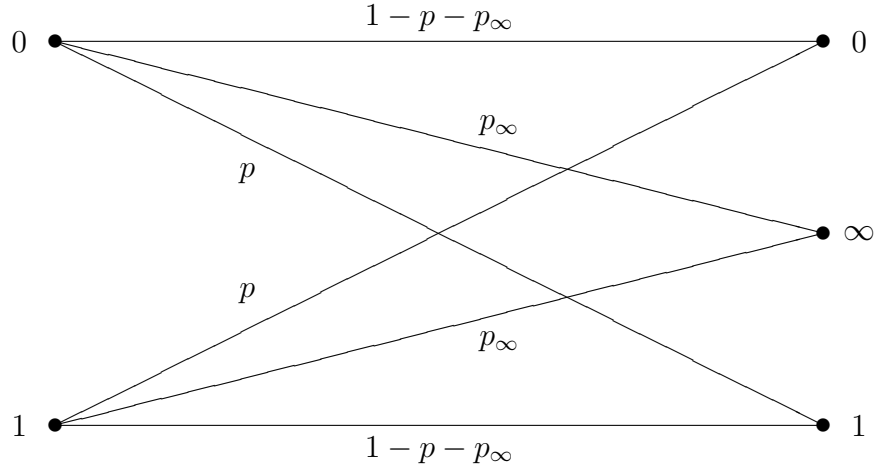
$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}.$$

La probabilité de transition p du canal sera souvent appelée probabilité d'erreur du canal.

Définition 4.2 (Canal symétrique) *Un canal discret est dit* symétrique *si les lignes de sa matrice stochastique sont formées des mêmes éléments à l'ordre près.*

Un canal symétrique n'a pas forcément le même nombre d'entrées et de sorties.

Exemple : Le canal binaire symétrique à effacement est défini par le diagramme



et a pour matrice stochastique

$$\begin{pmatrix} 1-p-p_\infty & p_\infty & p \\ p & p_\infty & 1-p-p_\infty \end{pmatrix}.$$

4.1.2 Canal continu et canal discret

Nous avons vu dans le chapitre d'introduction que nous pouvions ramener l'étude des canaux continus à celle des canaux discrets.

En effet l'ensemble constitué du modulateur de données digitales, du canal continu et du démodulateur de données digitales dans la Figure 4.1 peut être considéré comme un canal discret.

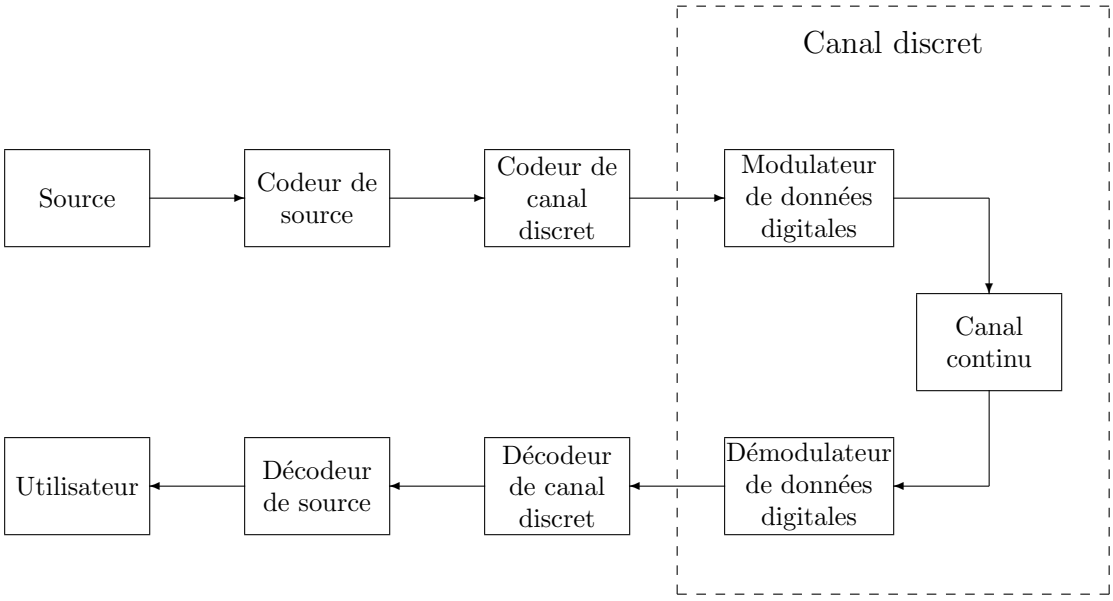


FIG. 4.1 – Canal continu et canal discret.

La principale difficulté dans un tel modèle consiste à "traduire" le bruit du canal continu en un bruit pour le canal discret correspondant. Cette traduction dépendra évidemment du type de modulation choisi.

4.2 Capacité d'un canal

4.2.1 Capacité d'un canal

Soit un canal dont l'ensemble des entrées est X et l'ensemble des sorties Y. La donnée du canal nous fourni également la loi de transition, c'est à dire la loi de probabilité conditionnelle de Y sachant X. Cette loi sera notée P, pour tout x dans X et tout y dans Y, nous connaissons $P(y \mid x)$.

Si nous connaissons également la loi d'émission, c'est-à-dire la loi de probabilité de X, alors nous sommes en mesure de calculer l'information mutuelle $I(X;Y)$ entre X et Y. En

effet

$$I(X;Y) = \sum_x \sum_y P(y \mid x) P(x) \log_2 \frac{P(y \mid x)}{P(y)}$$

et $P(x, y)$ et $P(y)$ peuvent être calculés et valent

$$P(x, y) = P(y \mid x)P(x), \quad P(y) = \sum_x P(y \mid x).$$

L'information mutuelle entre X et Y s'interprète comme l'information transmise à travers le canal. La relation

$$I(X;Y) = H(X) - H(X \mid Y)$$

exprime alors que l'information transmise à travers le canal est égale à l'entropie de X, $H(X)$ diminuée de la quantité $H(X \mid Y)$ que l'on peut interpréter alors comme la perte d'information due au bruit.

Pour un canal donné, seule la loi de transition est connue. La loi d'émission ne dépend que de la source et du codeur de canal. La capacité du canal sera la plus grande valeur que peut atteindre l'information mutuelle de X et Y.

Définition 4.3 (Capacité d'un canal) *Soit un canal d'entrée X et de sortie Y. La* capacité *de ce canal est égal au maximum de l'information mutuelle entre X et Y pour toutes les lois d'émission possible.*

Cette définition ne nécessite pas un canal discret, et reste parfaitement applicable à un canal continu. Nous ne nous intéresserons cependant qu'à la capacité des canaux discret sans mémoire.

4.2.2 Capacité d'un canal discret sans mémoire

Si la définition de la capacité d'un canal peut sembler simple, son calcul pratique est d'une grande compléxité en général. En effet si l'alphabet d'entrée est $X = \{a_1, \ldots, a_K\}$, et que la loi d'émission est $p_1, \ldots, p_K$, où $p_k = P(a_k)$ pour tout k, il faut calculer le maximum de $I(X;Y)$ sur l'ensemble des K-uplets $(p_1, \ldots, p_K)$ avec les contraintes $p_k \geq 0$ pour tout k, et $\sum p_k = 1$.

Ce calcul est cependant possible dans certains cas particuliers.

Capacité d'un canal symétrique. Pour un tel canal, la matrice stochastique

$$\Pi = \begin{pmatrix} P(b_1 \mid a_1) & \ldots & P(b_J \mid a_1) \\ \vdots & \ddots & \vdots \\ P(b_1 \mid a_K) & \ldots & P(b_J \mid a_K) \end{pmatrix}$$

est telle que ses lignes sont toutes égales à l'ordre près. On pose pour tout couple (k, j), $\Pi_{k,j} = P(b_j \mid a_k)$.

L'information mutuelle moyenne s'écrit

$$I(X;Y) = H(Y) - H(Y \mid X),$$

où $H(Y \mid X)$ l'entropie conditionnelle de Y sachant X est égale à

$$\begin{aligned} H(Y \mid X) &= -\sum_k \sum_j P(a_k, b_j) \log_2 P(b_j \mid a_k) \\ &= -\sum_k \sum_j \Pi_{k,j} p_k \log_2 \Pi_{k,j} \\ &= -\sum_k p_k \sum_j \Pi_{k,j} \log_2 \Pi_{k,j}. \end{aligned}$$

Par définition d'un canal symétrique, le terme $-\sum_j \Pi_{k,j} \log_2 \Pi_{k,j}$ est indépendant de k. Notons le $H(\Pi)$. Nous avons alors

$$H(Y \mid X) = \sum_k p_k H(\Pi) = H(\Pi) \sum_k p_k = H(\Pi).$$

Donc pour maximiser $I(X;Y) = H(Y) - H(\Pi)$ il suffira de maximiser $H(Y)$. Or nous savons que l'entropie de Y sera maximale lorsque la loi de Y sera uniforme et nous aurons alors $H(Y) = \log_2 J$.

La capacité d'un canal symétrique de matrice stochastique Π est donc égale à

$$C = \log_2 J - H(\Pi).$$

Exemple : Considérons le canal binaire symétrique. Sa matrice stochastique vaut

$$\Pi = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

et donc $H(\Pi) = -p \log_2 p - (1-p) \log_2 (1-p)$.

La capacité du canal binaire symétrique de probabilité d'erreur p est donc égale à

$$C = 1 + p \log_2 p + (1-p) \log_2 (1-p).$$

4.3 Théorème fondamental

Avant d'énoncer le second théorème de Shannon sur le codage des canaux bruité, nous allons décrire les façons par lesquelles nous pouvons opérer ce codage.

4.3.1 Codage de canal

D'après le hypothèses que nous avons admises jusqu'à présent, l'entrée du codeur de canal est égale à la sortie du codeur de source, c'est-à-dire qu'il s'agit d'une séquence binaire.

Définition 4.4 *Un* codage *d'un canal discret* (X, Y, Π) *est une procédure qui associe à chaque séquence binaire finie une séquence finie de lettres de* X*.*

Bien évidemment, il est hautement désirable qu'après passage dans le canal la séquence de lettres de Y reçue puisse être décodée pour retrouver la séquence binaire, et ce avec une probabilité la plus élevée possible.

Définition 4.5 *Un* code en bloc *d'un canal discret* (X, Y, Π) *est une procédure qui associe à chaque séquence binaire de longueur donnée* l *une séquence finie de* n *lettres de* X *appelée mot de code. L'entier* n *est appelé* longueur *du code,* 2^l *est le cardinal du code.*

Nous désignerons également par code l'ensemble des mots de codes pour tous les l*-uplets binaires possibles.*

À tout codage, il faudra évidemment associer une procédure de décodage. Cette procédure va pendre en entrée une séquece de lettres à la sortie du canal, et donner en sortie une séquence binaire. Ce decodage devra se faire de façon à maximiser la vraisemblance.

Définition 4.6 *Un* algorithme de décodage *d'un codage d'un canal* (X, Y, Π) *sera une procédure qui à toute séquence de lettres de* Y *associe une séquence binaire.*

Un algorithme de décodage *d'un code en bloc d'un canal* (X, Y, Π) *de longueur* n *et de cardinal* 2^l *sera une procédure qui à tout* n*-uplet de* Y *associe un* l*-uplet binaire.*

4.3.2 Canal binaire symétrique

Nous nous contenterons d'exposer le théorème fondamental de Shannon pour le codage de canal dans le cas d'un canal binaire symétrique. Il est aisé à partir de généraliser au cas q-aire, c'est-à-dire lorsque les alphabet d'entrée et de sortie du canal sont égaux et de cardinal q.

Comme nous l'avons déjà vu le canal binaire symétrique de probabilité de transition p à pour capacité $C = 1 + p \log_2 p + (1-p) \log_2 (1-p)$.

Pour un tel canal, nous utiliserons des codes binaires, c'est-à-dire des sous ensemble de $\{0,1\}^n$ pour un certain entier n.

Définition 4.7 *Le taux d'un code binaire de longueur* n *et de cardinal* M *est égal à*

$$R = \frac{\log_2 M}{n}.$$

Un code aura un taux de 1 lorsque $M = 2^n$, c'est-à-dire qu'aucune redondance n'aura été ajoutée. Un code de taux ne 1/2 signifiera que $M = 2^{n/2}$, c'est-à-dire, si n est pair que le code double la longueur des séquences binaires.

Soit $\mathcal{C}$ un code en bloc de longueur n et de cardinal M. Soient $\vec{x}$ un mot code et $\vec{y}$ un n-uplet de lettres de Y.

Soit ϕ un algorithme de décodage nous noterons $\Pr(\phi(\vec{y}) = \vec{x})$ la probabilité pour que le mot $\vec{y}$ de Y^n soit décodé par le mot de code $\vec{x}$. Nous noterons $\Pr(\phi(\vec{y}) \neq \vec{x}) = 1 - \Pr(\phi(\vec{y}) = \vec{x})$ la probabilité de l'évènement complémentaire.

La grandeur $\Pr(\phi(\vec{y}) \neq \vec{x})$ peut être considérée comme une v.a. réelle, donc il est possible de calculer sa moyenne.

Définition 4.8 (taux d'erreur résiduel) *Le* d'erreur résiduel *d'un algorithme de décodage est égale à la moyenne de la variable aléatoire* $\Pr(\phi(\vec{y}) \neq \vec{x})$ *lorsque la loi d'émission est uniforme.*

Si nous notons ce taux d'erreur résiduel P_e, nous avons

$$\begin{aligned} P_e &= \sum_{\vec{x},\vec{y}} P(\vec{x},\vec{y}) \Pr(\phi(\vec{y}) \neq \vec{x}), \\ &= \sum_{\vec{x},\vec{y}} P(\vec{y} \mid \vec{x}) P(\vec{x}) \Pr(\phi(\vec{y}) \neq \vec{x}), \\ &= \sum_{\vec{x},\vec{y}} \frac{P(\vec{y} \mid \vec{x})}{M} \Pr(\phi(\vec{y}) \neq \vec{x}). \end{aligned}$$

La sommation s'effectue pour tout $\vec{x}$ dans $\mathcal{C}$ et tout $\vec{y}$ dans Y^n.

La taux d'erreur P_e est bien un taux d'echec moyen dans le cas ou la loi d'émission est uniforme. Ce n'est pas le cas en général.

Exemple : Considerons le code à répétition de longueur impaire $n = 2t + 1$ utilisé pour lutter contre le bruit dans un canal binaire symétrique sans mémoire de probabilité d'erreur p.

Un code à répétition est un code constitué de deux mots : $0 \ldots 0$ et $1 \ldots 1$ de longueur n qui codent respectivement les symboles "0" et "1".

Un algorithme de décodage possible pour ce code est le décodage dit majoritaire : une séquence de n symboles binaires sera décodée par "0" si elle comporte une majorité de "0" dans son écriture, et par "1" sinon.

La probabilité pour que le mot $0 \ldots 0$ transmis sur le canal soit décodé par le symbole 1 est égale à

$$P_e = \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i},$$

c'est-à-dire la probabilité pour que le nombre de symboles ayant été transformé soit supérieur ou égal à t+1.

Il est possible de montrer que pour une probabilité d'erreur $p < 1/2$ fixée, la grandeur P_e peut être rendu arbitrairement petite.

4.3.3 Le second théorème de Shannon pour un canal binaire symétrique

Théorème 4.1 (Second théorème de Shannon) *Soit un canal binaire symétrique de capacité C. Pour tout $R < C$ il existe une suite $\mathcal{C}_n$ de codes de longueur n de taux R_n d'algorithme de décodage de taux d'erreur résiduel P_{en} telle que*

1. $\lim_{n\to\infty} R_n = R$
2. $\lim_{n\to\infty} P_{en} = 0$

Ce théorème signifie donc qu'il existe des codes en bloc permettant de réaliser un code dont le taux est aussi proche qu'on le désire de la capacité du canal.

Il existe un autre résultat qui est la réciproque de ce théorème.

Théorème 4.2 *Soit un canal binaire symétrique de capacité C. Soit un code de taux $R > C$ alors tout algorithme de décodage de ce code est tel que son taux d'erreur résiduel $P_e > K(R, C) > 0$, où $K(R, C)$ est une constante strictement positive ne dépendant que de R et de C.*

Ce résultat nous indique qu'il est inutile de chercher des codes de taux supérieur à la capacité du canal. La capacité C est donc bien le taux de codage maximal que l'on puisse atteindre pour faire transiter une information dans un canal donné.

Chapitre 5

Codes correcteurs d'erreurs

5.1 Définitions générales

5.1.1 Distance de Hamming

Définition 5.1 *Soient A^n l'ensemble des mots de longueur n sur A, $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1}) \in A^n$ et $\mathbf{y} = (y_0, y_1, \ldots, y_{n-1}) \in A^n$.*

- *La* distance de Hamming *entre* $\mathbf{x}$ *et* $\mathbf{y}$ *est*

$$d_H(\mathbf{x}, \mathbf{y}) = |\{i \mid 0 \leq i \leq n-1,\ x_i \neq y_i\}|.$$

  on vérifie que d_H est bien une métrique et on appelle alors espace de Hamming sur *A l'ensemble A^n muni de la métrique d_H.*
- *Si A est un groupe, le* poids de Hamming $w_H(\mathbf{x})$ *d'un mot $\mathbf{x} \in A^n$ est le nombre de ses coordonnées non nulles*

$$w_H(\mathbf{x}) = d_H(\mathbf{x}, \mathbf{0}),$$

  où $\mathbf{0}$ est le mot de A^n ayant toutes ses coordonnées égales à l'elément neutre de A.

5.1.2 Codes en bloc – Codes linéaires

Définition 5.2 *Un* code *C sur A est une partie non vide de l'espace de Hamming A^n, n est appelé* longueur *du code, les éléments de C sont appelés* mots de code.

Lorsque $A = \{0, 1\}$ on parlera de code binaire.

Définition 5.3 *On appelle* distance minimale *d'un code C sur A, l'entier*

$$d = min\{d_H(\mathbf{x}, \mathbf{y}) \mid \mathbf{x}, \mathbf{y} \in C, \mathbf{x} \neq \mathbf{y}\},$$

si A est un groupe, on appelle poids minimal *d'un code C, l'entier*

$$min\{w_H(\mathbf{x}) \mid \mathbf{x} \in C, \mathbf{x} \neq \mathbf{0}\}.$$

Lorsque l'alphabet A peut être muni d'une structure de corps, l'espace A^n est un espace vectoriel. On a alors la définition suivante.

Définition 5.4 *Un code C est dit* linéaire *sur A, si A est un corps et C un sous-espace-vectoriel de A^n. La dimension de C sur A est appelée la* dimension *du code C.*

Pour un code linéaire le *poids minimal* est égal à la *distance minimale*.

Les paramètres d'un code sont :
- son alphabet A de cardinal q,
- sa longueur n,
- son cardinal M (ou sa dimension k s'il est linéaire),
- sa distance minimale d.

Nous noterons $C(q; n, k, d)$ un code linéaire et $C[A; n, M, d]$ un code quelconque.

Définition 5.5 *Une* matrice génératrice *G d'un code linéaire C est une matrice $k \times n$ dont les lignes forment une base de C.*

Pour un même code il existe de nombreuses matrice génératrices. Parmi celles-ci certaines ont une forme pratique.

Proposition 5.1 *Tout code linéaire possède une matrice génératrice systématique, c'est-à-dire de la forme*

$$G = \left(\begin{array}{c|c} I_k & P \end{array} \right)$$

à une permutation des coordonnées près.

Définition 5.6 *Soit C un code linéaire de matrice génératrice G. Une* matrice de parité *H de C est une matrice $(n-k) \times n$ telle que $H\,{}^tG = 0$, où ${}^tG$ est la transposée de G.*

La matrice de parité H est telle que pour tout x dans C, $H\,{}^tx = 0$. C'est-à-dire que C est le noyau de l'endomorphisme de $\mathbb{F}_q^n$ dans $\mathbb{F}_q^{n-k}$ représenté par la matrice H.

Proposition 5.2 *Si G est une matrice génératrice systématique de C*

$$G = \left(\begin{array}{c|c} I_k & P \end{array} \right)$$

alors

$$H = \left(\begin{array}{c|c} -P & I_{n-k} \end{array} \right)$$

est une matrice de parité de C.

5.1.3 Décodage à vraisemblance maximale

Soit un canal de transmission (A, B, Π).

Définition 5.7 *Soit C un code de longueur n sur A. Un algorithme de décodage de C à* vraisemblance maximale *est une procédure de décodage $B^n \to C \subset A^n$ qui à tout élément de $\vec{y} \in B^n$ associe l'élement de $\vec{x} \in C$ réalisant le maximum de $P(\vec{y} \mid \vec{x})$.*

Proposition 5.3 *Dans un canal binaire symétrique de probabilité de transition $p < 1/2$, $P(\vec{y} \mid \vec{x})$ est une fonction croissante de $d_H(\vec{y}, \vec{x})$.*

preuve : Soient $\vec{x} = (x_1, \ldots, x_n)$ et $\vec{y} = (y_1, \ldots, y_n)$, on a

$$P(\vec{y} \mid \vec{x}) = \prod_{i=1}^{n} P(y_i \mid x_i) = p^{d_H(\vec{y},\vec{x})}(1-p)^{n-d_H(\vec{y},\vec{x})} = (1-p)^n \left(\frac{p}{1-p}\right)^{d_H(\vec{y},\vec{x})}.$$

□

Proposition 5.4 *Soit ϕ un algorithme de décodage à vraisemblance maximale de C dans un canal binaire symetrique. Si $\phi(\vec{y}) = \vec{x}$ alors pour tout $\vec{x}'$ dans C on a $d_H(\vec{y}, \vec{x}) \leq d_H(\vec{y}, \vec{x}')$.*

Proposition 5.5 *Un code de distance minimale d peut corriger $(d-1)/2$ erreurs.*

5.2 Quelques classes de codes

5.2.1 Codes parfaits

Définition 5.8 *On appelle* code parfait *un code tel que l'ensemble des boules de rayon $\lfloor \frac{d-1}{2} \rfloor$ centrées en tous les éléments du code forment une partition de l'espace de Hamming A^n.*

Tous les codes parfaits sont connus.

Exemple : 1. Code à repétition de longueur impaire $n = 2t+1$. Ce code contient deux mots : $0 \ldots 0$ et $1 \ldots 1$. Sa matrice génératrice est

$$G = \begin{pmatrix} 1 & \ldots & 1 \end{pmatrix}$$

et sa matrice de parité est

$$H = \begin{pmatrix} 1 & & 0 & 1 \\ & \ddots & & \vdots \\ 0 & & 1 & 1 \end{pmatrix}$$

Ce code a pour distance minimale $d = n = 2t+1$, et tout mot de l'espace $\{0,1\}^n$ contient soit plus de $t+1$ "0", soit plus de $t+1$ "1", et se trouve donc à distance au plus $t = (d-1)/2$ d'un des deux mots de code.

2. Code de Hamming $\mathcal{H}_m$ est un code binaire linéaire de paramètres $(n = 2^m - 1, k = 2^m - m - 1, d = 3)$. La matrice de parité H_m de ce code est constituée de l'ensemble des $2^m - 1$ vecteurs-colonne non nuls de $\mathbb{F}_2^m$. Par exemple :

$$H_3 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Le nombre de points d'une boule de rayon $(d-1)/2 = 1$ centrée en un mot du code $\mathcal{H}_m$ est $\binom{n}{0} + \binom{n}{1} = 2^m$. La réunion de toutes ces boules a donc un cardinal $2^k 2^m = 2^{k+m} = 2^n$, c'est-à-dire le cardinal de l'espace tout entier. Le code $\mathcal{H}_m$ est donc parfait.

5.2.2 Codes cycliques

a - Corps finis

Théorème 5.1 *Soit F un corps fini de cardinal $q > 1$. Alors*

1. *$q = p^m$, où p est un nombre premier et m un entier positif.*
2. *F est unique à isomorphisme près.*

- Lorsque p est premier le corps $\mathbb{F}_p$ est égal à $\mathbb{Z}/p\mathbb{Z}$.
- Lorsque $q = p^m$, avec $m > 1$, on dira que $\mathbb{F}_q$ est une extension de degré m de $\mathbb{F}_p$.

Proposition 5.6 *Soit $\mathbb{F}_q$ un corps fini. Soit $p_m(x)$ un polynôme irréductible de $\mathbb{F}_q[x]$ de degré m. Soit $(p_m(x))$ l'idéal engendré par le polynôme $p_m(x)$, c'est à dire l'ensemble des polynômes multiple de $p_m(x)$.*

1. *Le quotient $\mathbb{F}_q[x]/(p_m(x))$ est un corps fini de cardinal q^m.*
2. *Il existe toujours un élement α de $\mathbb{F}_{q^m}$ tel que $p_m(\alpha) = 0$.*

Définition 5.9 *Un polynôme $p_m(x)$ irréductible de degré m est* primitif *si l'ensemble des restes de la division de x^i par $p_m(x)$ sont tous distincts pour $0 \leq i < q^m - 1$.*

Un élement α de $\mathbb{F}_{q^m}$ tel que $p_m(\alpha) = 0$ est dit primitf.

Soit α un élement primitif de $\mathbb{F}_{q^m}$, on notera $\mathbb{F}_{q^m} = \mathbb{F}_q[\alpha]/(p_m(\alpha))$.

- Le corps fini à q^m élements est égal à l'ensemble $\{0, 1, \alpha, \alpha^2, \ldots, \alpha^{q^m-2}\}$.
- Le corps fini à q^m élements est égal à l'ensemble des polynômes en α de degré $< m$.
- L'addition deux élements du corps sera l'addition de deux polynômes de $\mathbb{F}_q[\alpha]$.
- La multiplication de deux élements du corps sera le reste de la division par $p_m(\alpha)$ de la multiplication de deux polynômes de $\mathbb{F}_q[\alpha]$.

b - Codes BCH

L'alphabet est égal à $\mathbb{F}_q$, le corps à q éléments. Soit m un entier et $n = q^m - 1$. On note $\mathcal{R}_n = \mathbb{F}_q[X]/(X^n - 1)$ l'anneau des polynômes à coefficients dans $\mathbb{F}_q$ modulo l'idéal engendré par $X^n - 1$. Soit α un élement primitif de $\mathbb{F}_{q^m}$.

Définition 5.10 (codes cycliques) *Un* code cyclique *binaire de longueur n est un idéal de $\mathcal{R}_n$.*

Remarque :

1. Un code cyclique est linéaire.
2. Un code cyclique est invariant par permutation circulaire de ses coordonnées dans la base $(1, X, X^2, \ldots, X^{n-1})$de $\mathcal{R}_n$. La permutation circulaire d'une position à droite correspond à la multiplication par X modulo $X^n - 1$

$$\begin{aligned} & X\left(a_0 + a_1 X + \ldots + a_{n-2}X^{n-2} + a_{n-1}X^{n-1}\right) \\ &= a_0 X + a_1 X^2 + \ldots + a_{n-2}X^{n-1} + a_{n-1}X^n \\ &= a_{n-1} + a_0 X + a_1 X^2 + \ldots + a_{n-2}X^{n-1} \bmod X^n - 1. \end{aligned}$$

Théorème 5.2 *Soit C un code cyclique binaire de longueur n, on a*

i. *Il existe un unique polynôme unitaire $g(X)$ de degré minimal dans C.*

ii. *C est l'idéal engendré par $g(X)$, $g(X)$ est appelé* polynôme générateur *de C.*

iii. *$g(X)$ est un facteur de $X^n - 1$.*

iv. *Tout $c(X) \in C$ s'écrit de façon unique $c(X) = f(X)g(X)$ dans $\mathbb{F}_q[X]$. La dimension de C est $n - r$ où $r = \deg g(X)$.*

Définition 5.11 (codes BCH) *Soit C un code cyclique binaire de longueur $n = q^m - 1$ et soit $g(X)$ son polynôme générateur. C est un code BCH de* distance construite δ *si δ est le plus grand entier vérifiant*

$$\exists\, b \in \mathbb{Z},\ g(\alpha^b) = g(\alpha^{b+1}) = \ldots = g(\alpha^{b+\delta-2}) = 0,$$

où α est un élement primitif de $\mathbb{F}_{q^m}$.

Définition 5.12 *Un code est dit BCH* au sens strict *si $b = 1$. On notera $B(n, \delta)$ le code BCH binaire au sens strict de longueur n et de distance construite δ.*

Théorème 5.3 (borne BCH) *Soit un code BCH de distance construite δ, sa distance minimale d vérifie*

$$d \geq \delta.$$

Exemple : Le code de Hamming est un code BCH binaire de distance construite 3.

c - Code de Reed-Solomon

Définition 5.13 *Un code de Reed-Solomon est un code BCH de longueur $q-1$ sur $\mathbb{F}_q$.*

5.2.3 Codes détecteurs d'erreur

Un code peut être utilisé pour detecter les erreurs au lieu de les corriger. Le problème que l'on souhaite résoudre est le suivant : un mot de code est transmis à travers un canal bruité et l'utilisateur souhaite savoir si ce mot a été modifié.

Pour qu'une erreur non detectée ait lieu, il faut qu'un mot de code soit modifié en un mot de code différent.

Proposition 5.7 *Un code de distance minimale d peut detecter $d-1$ erreurs.*

Exemple :

1. Le code de parité de longueur n est l'ensemble des séquences binaires de poids pair. Sa matrice génératrice est égale à

$$G = \begin{pmatrix} 1 & & 0 & 1 \\ & \ddots & & \vdots \\ 0 & & 1 & 1 \end{pmatrix}$$

   Le code est obtenu en ajoutant à une séquence de longueur $n-1$ un n-ième symbole égal à la somme des autres. Par les les 128 caractéres ASCII sont codés par des mots de longueur 8.

   Ce code permet de décoder une erreur.

2. code CRC (Cyclic Redundancy Check).

   Une séquence de k symboles binaires d'information est représentée par le polynôme $i(x)$. Soit $g(x)$ un polynôme de degré s. Le mot de code correspondant à $i(x)$ est égal à $c(x) = x^s i(x) + r(x)$ où $r(x)$ est le reste de la division de $x^s i(x)$ par $g(x)$.

   Ce code permet de detecter toute rafale (erreurs consécutives) d'erreurs de longueur inférieure ou égale à s.

5.3 Décodage des codes linéaires

5.3.1 Théorie algébrique du décodage

Dans toute cette section on considère un code linéaire $C(n,k,d)$ sur $\mathbb{F}_2$.

a - Position du problème

Etant donné
– que $\mathbf{x} \in C$ est le "message transmis",

– que $\mathbf{x}$ est perturbé dans un canal bruité par l'erreur $\mathbf{e} \in \mathbb{F}_2^n$,
– que $\mathbf{y} = \mathbf{x} + \mathbf{e}$, le "message reçu", est le seul mot auquel le décodeur a accès,
le problème du décodage est de retrouver $\mathbf{x}$ à partir de $\mathbf{y}$.

Un algorithme de décodage de C devra donc prendre comme argument un élément de $\mathbb{F}_2^n$, et s'il se termine, rendre un élément du code C. Il devra également être déterministe, c'est-à-dire qu'un mot de l'espace $\mathbb{F}_2^n$ sera toujours décodé de la même manière. Nous proposons la définition suivante.

Définition 5.14 *Soit un code linéaire $C(n,k,d)$ sur $\mathbb{F}_2$, un* algorithme de décodage d'erreur de C *est une application*

$$\begin{array}{rccc} \gamma & : & \mathbb{F}_2^n & \longrightarrow & C \cup \{\infty\} \\ & & \mathbf{y} & \longmapsto & \gamma(\mathbf{y}) \end{array}$$

telle que $\forall\, \mathbf{x} \in C$, $\gamma(\mathbf{x}) = \mathbf{x}$. Le fait que $\gamma(\mathbf{y}) = \infty$ signifiant que $\mathbf{y}$ n'a pas été décodé.

Soit e un entier positif, on dira que γ est borné par e, *si*

$$\forall\, \mathbf{y} \in \mathbb{F}_q^n,\ \forall\, \mathbf{x} \in C,\quad d_H(\mathbf{x},\mathbf{y}) < \frac{e}{2} \Rightarrow \gamma(\mathbf{y}) = \mathbf{x}$$

où d_H est la métrique de Hamming sur $\mathbb{F}_q$, on dira que γ est borné strictement par e *si on a de plus*

$$\gamma(\mathbf{y}) = \mathbf{x} \neq \infty \Rightarrow d_H(\mathbf{x},\mathbf{y}) < \frac{e}{2}$$

b - Matrice de parité – Syndrome

Définition 5.15 *On appelle* matrice de parité *du code $C(n,k,d)$ une matrice H à $n-k$ lignes et n colonnes sur $\mathbb{F}_2$, telle que $C = \ker H = \{\mathbf{x} \mid H\mathbf{x}^t = 0\}$.*

La matrice H détermine le code C. Remarquons que par un théorème fondamental d'algèbre linéaire, les lignes de H forment une base de l'orthogonal de C dans $\mathbb{F}_2^n$ pour le produit scalaire usuel.

Définition 5.16 *Soit $\mathbf{y} \in \mathbb{F}_2^n$. Le* syndrome *de $\mathbf{y}$ est le vecteur de $\mathbb{F}_2^{n-k}$*

$$S(\mathbf{y}) = H\mathbf{y}^t.$$

l'application $S : \mathbb{F}_2^n \to \mathbb{F}_2^{n-k}$ ainsi définie est $\mathbb{F}_2$-linéaire.

H induit une relation d'équivalence sur $\mathbb{F}_2^n$

$$\mathbf{x}\ \mathcal{R}\ \mathbf{y} \Leftrightarrow H\mathbf{x}^t = H\mathbf{y}^t \Leftrightarrow H(\mathbf{x}-\mathbf{y})^t = 0 \Leftrightarrow \mathbf{x}-\mathbf{y} \in C.$$

Chacune des classes de cette relation d'équivalence est appelée "classe latérale" ou "translaté" de C.

Le code C est la classe de l'élément nul de $\mathbb{F}_2^n$. Plus généralement, deux éléments sont dans un même translaté si et seulement si ils ont le même syndrome.

Proposition 5.8 *Il existe au plus un mot de poids $< d/2$ dans un translaté donné.*

preuve : Soient $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$ tels que $\mathbf{x}\ \mathcal{R}\ \mathbf{y}$, $w_H(\mathbf{x}) < d/2$ et $w_H(\mathbf{y}) < d/2$. Alors $\mathbf{x}-\mathbf{y} \in C$ et $w_H(\mathbf{x}-\mathbf{y}) = d(\mathbf{x},\mathbf{y}) \leq d(\mathbf{x},\mathbf{0}) + d(\mathbf{0},\mathbf{y}) < d/2 + d/2 = d$, en vertu de l'inégalité triangulaire. Donc $\mathbf{x} = \mathbf{y}$. □

c - Décodage des codes linéaires

A partir de la matrice de parité et des syndromes on peut définir l'algorithme de décodage γ suivant :

1. Dans tout translaté $\mathbf{y} + C$, on choisit un élément de plus petit poids, $e(\mathbf{y})$ que l'on met dans une table indexée par le syndrome de $\mathbf{y}$. Remarquons que $e(\mathbf{y}+\mathbf{x}) = e(\mathbf{y})$ pour tout $\mathbf{x} \in C$
2. Si $\mathbf{y}$ est le mot reçu, on calcule son syndrome $S(\mathbf{y})$, et on lit $e(\mathbf{y})$ dans la table.
3. l'algorithme retourne $\gamma(\mathbf{y}) = \tilde{\mathbf{x}} = \mathbf{y} - e(\mathbf{y})$.

Proposition 5.9 *γ est un algorithme à vraisemblance maximale, donc il est borné par sa distance minimale.*

preuve : Soit $\mathbf{y} \in \mathbb{F}_q^n$, $\tilde{\mathbf{e}}$ un élément de plus petit poids de la classe de $\mathbf{y}$

- $\gamma(\mathbf{y}) \in C$, en effet $\mathbf{y}\ \mathcal{R}\ \tilde{\mathbf{e}}$, donc $\gamma(\mathbf{y}) = \mathbf{y} - \tilde{\mathbf{e}}\ \mathcal{R}\ \mathbf{0}$.
- $\forall\, \mathbf{x} \in C$, $\mathbf{y} - \mathbf{x}\ \mathcal{R}\ \tilde{\mathbf{e}}$, donc $d_H(\mathbf{x},\mathbf{y}) = w_H(\mathbf{y}-\mathbf{x}) \geq w_H(\tilde{\mathbf{e}}) = d_H(\gamma(\mathbf{y}),\mathbf{y})$, donc $\gamma(\mathbf{y})$ est un élément du code réalisant le minimum de la distance à $\mathbf{y}$.
- L'algorithme est linéaire. En effet, pour tout $\mathbf{x} \in C$, $\mathbf{y} \in \mathbb{F}_q^n$

$$\gamma(\mathbf{y}+\mathbf{x}) = \mathbf{y}+\mathbf{x}-e(\mathbf{y}+\mathbf{x}) = \mathbf{y}+\mathbf{x}-e(\mathbf{y}) = \gamma(\mathbf{y})+\mathbf{x},$$

puisque $e(\mathbf{y}+\mathbf{x}) = e(\mathbf{y})$.

□

Le décodage des codes linéaires se ramène donc à la recherche d'un mot de plus petit poids d'un translaté donné par son syndrome. Cette recherche est généralement difficile, mais dans certains cas, comme celui des codes BCH, on possède un algorithme qui fonctionne pour certains translatés.

5.3.2 Décodage des codes cycliques

Dans cette section les vecteurs de $\mathbb{F}_q^n$ sont notés en gras.

a - Code de Hamming binaire $d = 3$

Le code de Hamming binaire $\mathcal{H}_m$ est un code linéaire de paramètres $(n = 2^m - 1, 2^m - m - 1, 3)$, les colonnes de sa matrice de parité sont les $2^m - 1$ vecteurs distincts non nuls de $\mathbb{F}_2^m$ (c'est bien une matrice $m \times n$) :

$$H_m = \begin{pmatrix} 0 & 0 & 0 & 0 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 1 & \dots & 1 \\ 0 & 1 & 1 & 0 & \dots & 1 \\ 1 & 0 & 1 & 0 & \dots & 1 \end{pmatrix}$$

le code $\mathcal{H}_m$ est l'ensemble des combinaisons linéaires nulles des colonnes de H_m.

Exemple : $m = 3$, le code de Hamming $\mathcal{H}_3$ a pour paramètres $(7, 4, 3)$, et pour matrice de parité :

$$H_3 = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Proposition 5.10 *Chaque coset du code de Hamming $\mathcal{H}_m$ possède un et un seul mot de poids 1.*

preuve : Nous savons qu'il existe au plus un mot de poids 1 dans chaque coset, de plus il y a exactement $2^m - 1$ mots de poids 1, et $2^m - 1$ cosets différents de $\mathcal{H}_m$. □

Nous pouvons en déduire que l'agorithme de décodage suivant est à vraisemblance maximale.

Soit $\mathbf{y} = (y_0, \dots, y_{n-1}) \in \mathbb{F}_2^n$.

1. On calcule $\mathbf{s} = H_m \mathbf{y}^t \in \mathbb{F}_2^m$
2. si $\mathbf{s} = 0$ alors $\mathbf{y} \in C$ et $\longrightarrow \mathbf{y}$
3. sinon $\mathbf{s}$ est la j-ième colonne de H_m et $\longrightarrow \mathbf{x} = (y_0, \dots, y_{j-1}, 1 + y_j, y_{j+1}, \dots, y_{n-1}) \in C$

(Notons que l'ordre dans lequel les colonnes de H_m sont écrites n'est pas important)

Le code de Hamming cyclique

On peut décider d'écrire la matrice H_m d'une façon légèrement différente, considerons la matrice :

$$H_m = \begin{pmatrix} 1 & \alpha & \alpha^2 & \dots & \alpha^{n-1} \end{pmatrix}$$

où α est un élément générateur de $\mathbb{F}_{2^m}^*$. $\mathcal{H}_m$ est l'ensemble des vecteurs de $\mathbb{F}_2^n$ orthogonaux à H_m (les calculs sont effectués dans $\mathbb{F}_{2^m}$). Ce code est cyclique.

Proposition 5.11 *$\mathcal{H}_m$ est l'ensemble des polynômes de $\mathbb{F}_2[x]/(x^n-1)$ s'annulant en α. C'est aussi l'idéal engendré par le polynôme minimal de α dans $\mathbb{F}_2$.*

Exemple : $m = 3$, on considère le code de Hamming de matrice de parité :

$$H_3 = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \end{pmatrix}$$

où α est un élément générateur de $\mathbb{F}_8$ tel que $\alpha^3+\alpha+1 = 0$ (par exemple). En écrivant H_3 dans la base $1, \alpha, \alpha^2$, on obtient :

$$H_3 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

qui est la même que celle décrite au début à une permutation des colonnes près.

b - Codes 2-correcteur binaires, $d = 5$

Nous considerons à présent le code C suivant : l'ensemble des polynômes de $\mathbb{F}_2[x]/(x^n-1)$ s'annulant en α et en α^3. Il s'agit d'un code linéaire binaire de longueur $n = 2^m - 1$. Sa matrice de parité peut s'écrire :

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \dots & \alpha^{n-1} \\ 1 & \alpha^3 & \alpha^6 & \dots & \alpha^{3(n-1)} \end{pmatrix}$$

Ce code a une distance minimale $d \geq 5$ et peut donc corriger 2 erreurs.

Décodage

Nous faisons les hypothèses suivantes :
- $a(x) \in \mathbb{F}_2[x]/(x^n-1)$, tel que $a(\alpha) = a(\alpha^3) = 0$
- $e(x) = x^{i_1} + x^{i_2}$, avec $0 \leq i_j \leq n-1$, pour $j = 1, 2$ et $i_1 \neq i_2$
- on connait le polynôme $b(x) = a(x) + e(x)$

comment déterminer les valeurs de i_1 et i_2 ?

On pose $X_1 = \alpha^{i_1}$ et $X_2 = \alpha^{i_2}$, qui seront les inconnues, et on connait les valeurs de $\xi_1 = b(\alpha) = e(\alpha)$ et de $\xi_3 = b(\alpha^3) = e(\alpha^3)$. On veut donc résoudre dans $\mathbb{F}_{2^m}$ le système :

$$\begin{cases} X_1 + X_2 &= \xi_1 \\ X_1^3 + X_2^3 &= \xi_3 \end{cases}$$

on a :

$$\xi_1^3 = X_1^3 + X_2^3 + X_1X_2(X_1 + X_2) = \xi_3 + X_1X_2\xi_1$$

on peut donc reécrire le système :

$$\begin{cases} X_1 + X_2 &= \xi_1 \\ X_1X_2 &= \frac{\xi_1^3+\xi_3}{\xi_1} \end{cases}$$

on se ramène donc à la résolution d'une équation du second degré dans $\mathbb{F}_{2^m}$. □

c - Codes BCH de distance construite δ

Soit α un élément primitif de $\mathbb{F}_{q^m}$, $n = q^m - 1$.

On défini le code BCH primitif au sens strict sur $\mathbb{F}_q$ de distance construite δ comme l'ensemble des polynômes de $\mathbb{F}_q[z]/(z^n - 1)$ s'annulant en $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$. Ce code a pour matrice de parité :

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \ldots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \ldots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{\delta-1} & \alpha^{2(\delta-1)} & \ldots & \alpha^{(n-1)(\delta-1)} \end{pmatrix}$$

Le théorème de la borne BCH nous assure que ce code a une distance minimale supérieure ou égale à δ, ce code est donc au moins t-correcteur où :

$$t = \lfloor \frac{\delta - 1}{2} \rfloor$$

pour $t > 2$, il est difficile d'effectuer un calcul direct de l'erreur.

Par exemple dans le cas binaire pour $t = 3$, il faudrait résoudre le système :

$$\begin{cases} X_1 + X_2 + X_3 = \xi_1 \\ X_1^3 + X_2^3 + X_3^3 = \xi_3 \\ X_1^5 + X_2^5 + X_3^5 = \xi_5 \end{cases}$$

où ξ_1, ξ_3 et ξ_5 seraient connus, et X_1, X_2 et X_3 inconnus. Ce système semble à première vue peu pratique.

En fait on sait décoder les codes BCH à l'aide de l'algorithme d'Euclide étendu. Dans la section suivante nous présenterons les définitions utiles à l'exposé de l'algorithme proprement dit.

d - polynômes localisateur et évaluateur

Soit α une racine n-ième de l'unité, $\mathbb{F}_{q^m}$ la plus petite extension de $\mathbb{F}_q$ contenant α.

Soit $\mathbf{a} = (a_0, a_1, \ldots, a_{n-1}) \in \mathbb{F}_q^n$, de poids w, les composantes non nulles de ce vecteur sont exactement :

$$a_{i_1}, a_{i_2}, \ldots, a_{i_w}$$

on associe à $\mathbf{a}$ les éléments suivants de $\mathbb{F}_{q^m}$:

$$X_1 = \alpha^{i_1}, X_2 = \alpha^{i_2}, \ldots, X_w = \alpha^{i_w}$$

appelés les *localisateurs* de $\mathbf{a}$, ainsi que les éléments suivants de $\mathbb{F}_q$:

$$Y_1 = a_{i_1}, Y_2 = a_{i_2}, \ldots, Y_w = a_{i_w}$$

qui sont les valeurs des coordonnées non nulles de $\mathbf{a}$.

Définition 5.17 *Le* polynôme localisateur *de* **a** *est le polynôme :*

$$\sigma(z) = \prod_{i=1}^{w}(1 - X_i z) = \sum_{i=0}^{w} \sigma_i z^i \in \mathbb{F}_{q^m}[z].$$

Les racines de $\sigma(z)$ *sont les inverses des localisateurs de* **a**.

Les coefficients σ_i de $\sigma(z)$ sont les fonctions symétriques élémentaires des X_i :

$$\begin{cases} \sigma_1 &= -(X_1 + \ldots + X_w), \\ \sigma_2 &= X_1X_2 + X_1X_3 + \ldots + X_{w-1}X_w, \\ &\vdots \\ \sigma_w &= (-1)^w X_1 \ldots X_w. \end{cases}$$

Définition 5.18 *Le* polynôme évaluateur *de* **a** *est le polynôme :*

$$\omega(z) = \sum_{i=1}^{w} X_i Y_i \prod_{\substack{j=1 \\ j\neq i}}^{w}(1 - X_j z)$$

on a pour tout i, $1 \leq i \leq w$ *:*

$$Y_i = \frac{X_i^{-1}\omega(X_i^{-1})}{\prod_{j\neq i}(1 - X_j X_i^{-1})}.$$

La connaissance de $\sigma(z)$ *et* $\omega(z)$ *permet de déterminer entièrement le vecteur* **a**.

On a $\deg\omega(z) < \deg\sigma(z) = w$, et $\sigma(z)$ et $\omega(z)$ sont premiers entre eux.

Définition 5.19 *Le* polynôme de Mattson-Solomon *associé à* **a** *est le polynôme suivant de* $\mathbb{F}_{q^m}[z]$ *:*

$$A(z) = \sum_{j=1}^{n} A_j z^{n-j}, \text{ où } A_j = \sum_{i=0}^{n-1} a_i \alpha^{ij}.$$

A_j *est la valeur en* α^j *du polynôme* $a(x) = a_0 + a_1 x + \ldots + a_{n-1}x^{n-1}$, *notons que* A_j *est défini pour tout entier* j.

on a donc :

$$\begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \alpha & \alpha^2 & \ldots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \ldots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{n-1} & \alpha^{2(n-1)} & \ldots & \alpha^{(n-1)^2} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

Théorème 5.4 *Soit* $\mathbf{a} = (a_0, \ldots, a_{n-1}) \in \mathbb{F}_q^n$, $\sigma(z)$ *le polynôme localisateur de* $\mathbf{a}$, $\omega(z)$ *son polynôme évaluateur. Pour tout* $j = 1, \ldots, \infty$, $A_j = a(\alpha^j)$ *où* $a(x) = a_0 + a_1 x + \ldots + a_{n-1}x^{n-1}$. *On a :*

$$\omega(z) = S(z)\sigma(z)$$

où

$$S(z) = \sum_{j=1}^{\infty} A_j z^{j-1}.$$

preuve :

$$\begin{aligned}
\frac{\omega(z)}{\sigma(z)} &= \sum_{i=1}^{w} \frac{X_i Y_i}{1 - zX_i}, \\
&= \sum_{i=1}^{w} Y_i X_i \sum_{j=1}^{\infty} (zX_i)^{j-1}, \\
&= \sum_{j=1}^{\infty} z^{j-1} \sum_{i=1}^{w} Y_i X_i^j, \\
&= \sum_{j=1}^{\infty} z^{j-1} \sum_{i=0}^{n-1} a_i \alpha^{ij}, \\
&= \sum_{j=1}^{\infty} A_j z^{j-1} = S(z).
\end{aligned}$$

□

e - l'algorithme d'Euclide étendu

Quel problème doit on résoudre ? Soit $\mathbf{a} \in C \longrightarrow \mathbf{b} = \mathbf{a} + \mathbf{e} \in \mathbb{F}_q^n$, connaissant $\mathbf{b}$ on veut déterminer $\mathbf{e}$, en supposant que le poids de $\mathbf{e}$ est suffisemment petit.

Le décodeur a accès à $H\mathbf{b}^t = H\mathbf{e}^t$, ce qui revient, pour un code BCH primitif au sens strict, à connaitre les valeurs en $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$ du polynôme $e(x) = e_0 + e_1 x + \ldots + e_{n-1}x^{n-1}$ associé au vecteur $\mathbf{e}$. Si l'on se réfère à la section précédente, il suffit de calculer les valeurs des polynômes localisateur et évaluateur de $\mathbf{e}$.

Proposition 5.12 *Soient* $e(x) = e_0 + e_1 x + \ldots + e_{n-1}x^{n-1} \in \mathbb{F}_q[x]/(x^n - 1)$ *de poids* $w_H(e) \leq t = \lfloor \frac{\delta-1}{2} \rfloor$, *et* $S(z) = \sum_{i=1}^{\infty} e(\alpha^i) z^{i-1}$. *Les polynômes localisateur et évaluateur de* $e(x)$ *sont les seuls polynômes* $\sigma(z)$ *et* $\omega(z)$ *vérifiant :*

$$\begin{cases}
\omega(z) \equiv \sigma(z)S(z) \bmod z^{\delta-1} \\
\omega(z) \text{ et } \sigma(z) \text{ premiers entre eux} \\
\sigma(0) = 1 \\
\deg \omega(z) < t \\
\deg \sigma(z) \leq t
\end{cases} \tag{5.1}$$

preuve : D'après la section précédente, les polynômes localisateur et évaluateur de e vérifient (5.1). Réciproquement soient $\tilde{\sigma}(z)$ et $\tilde{\omega}(z)$ vérifiant (5.1), on a :

$$\tilde{\omega}(z)\sigma(z) \equiv \tilde{\sigma}(z)S(z)\sigma(z) \equiv \tilde{\sigma}(z)\omega(z) \bmod z^{\delta-1},$$

$$\tilde{\omega}(z)\sigma(z) - \tilde{\sigma}(z)\omega(z) \equiv 0 \bmod z^{\delta-1},$$

or $\tilde{\omega}(z)\sigma(z) - \tilde{\sigma}(z)\omega(z)$ est de degré $< \delta - 1$, donc $\tilde{\omega}(z)\sigma(z) = \tilde{\sigma}(z)\omega(z)$, ce qui suffit à montrer le résultat. □

Description de l'algorithme

Théorème 5.5 (Algorithme d'Euclide) *Soient $r_{-1}(z)$ et $r_0(z)$, tels que $\deg r_0 \leq \deg r_{-1}$, on construit deux suites $(r_i(z))_{i\geq 1}$ et $(q_i(z))_{i\geq 1}$ telles que :*

$$\begin{array}{rcll}
r_{-1}(z) & = & q_1(z)r_0(z) + r_1(z), & \deg r_1 < \deg r_0 \\
r_0(z) & = & q_2(z)r_1(z) + r_2(z), & \deg r_2 < \deg r_1 \\
r_1(z) & = & q_3(z)r_2(z) + r_3(z), & \deg r_3 < \deg r_2 \\
 & \vdots & & \vdots \\
r_{j-2}(z) & = & q_j(z)r_{j-1}(z) + r_j(z), & \deg r_j < \deg r_{j-1} \\
r_{j-1}(z) & = & q_{j+1}(z)r_j(z). &
\end{array}$$

alors $r_j(z)$, le dernier reste non nul de ces divisions est le pgcd *de $r_{-1}(z)$ et $r_0(z)$.*

Soient les polynômes $U_i(z)$ et $V_i(z)$ définis par :

$$\begin{array}{ll} U_{-1}(z) = 0, & U_0(z) = 1, \\ V_{-1}(z) = 1, & V_0(z) = 0, \end{array}$$

$$\begin{array}{l} U_i(z) = U_{i-2}(z) - q_i(z)U_{i-1}(z), \\ V_i(z) = V_{i-2}(z) - q_i(z)V_{i-1}(z). \end{array}$$

alors pour tout $i \geq 1$ on a :

$$\left\{ \begin{array}{l} r_i(z) = V_i(z)r_{-1}(z) + U_i(z)r_0(z) \\ \deg U_i = \deg r_{-1} - \deg r_{i-1} \\ U_i(z) \text{ et } V_i(z) \text{ premiers entre eux} \end{array} \right. \tag{5.2}$$

c'est cette relation que nous utiliserons pour l'algorithme de décodage.

Algorithme de décodage

On veut résoudre (5.1) :

$$\left\{ \begin{array}{l} \omega(z) \equiv \sigma(z)S(z) \bmod z^{\delta-1} \\ \omega(z) \text{ et } \sigma(z) \text{ premiers entre eux} \\ \sigma(0) = 1 \\ \deg \omega(z) < t \\ \deg \sigma(z) \leq t \end{array} \right.$$

$S(z) \bmod z^{\delta-1}$ étant connu.

Si on applique l'algorithme d'Euclide à $r_{-1}(z) = z^{\delta-1}$ et $r_0(z) = (S(z) \bmod z^{\delta-1})$, on peut alors construire à l'aide de (5.2), des polynômes $r_i(z)$ et des $U_i(z)$ vérifiant :

$$r_i(z) \equiv U_i(z) r_0(z) \bmod z^{\delta-1},$$

il existe un rang k pour lequel $\deg r_{k-1} \geq t$ et $\deg r_k < t$, on aura alors :

$$\begin{cases} \deg r_k < t, \\ \deg U_k = \deg r_{-1} - \deg r_{k-1} \leq t \end{cases}$$

à un coefficient multiplicatif près, on a donc résolu le problème. Soit λ tel que $\lambda U_k(0) = 1$, alors :

$$\begin{cases} \sigma(z) = \lambda U_k(z) \\ \omega(z) = \lambda r_k(z). \end{cases}$$

est la solution du système (5.1). □

interprétation pour le décodage Soit $a(x) \in \mathbb{F}_2[x]/(x^n - 1)$, tel que

$$a(\alpha) = a(\alpha^2) = \ldots = a(\alpha^{\delta-1}) = 0,$$

on considère le polynôme $b(x) = a(x) + e(x)$, où $e(x)$ est de poids $w < \delta/2$.

Connaissant $b(x)$, nous voulons déterminer $a(x)$ et $e(x)$. Puisque pour $i = 1, \ldots, \delta - 1$, on a $a(\alpha^i) = 0$, pour ces même valeurs de i, $e(\alpha^i) = b(\alpha^i)$ est connu. Le syndrome de $e(x)$, $S(z) = \sum_{i=1}^{\delta-1} e(\alpha^i) z^{i-1}$, nous permet à l'aide de l'algorithme d'Euclide étendu de déterminer la valeur de $e(x)$ par l'intermédiaire de ses polynômes localisateur et évaluateur.

Chapitre 6

Codes convolutifs

6.1 Définition

Définition 6.1 *Un codeur convolutif binaire (k, n, m) est un circuit linéaire sans feedback, d'ordre m, à k entrées et n sorties sur $\mathbb{F}_2$, et invariant dans le temps.*

- $\vec{X}(D) = (X_1(D), \ldots, X_k(D))$ le vecteur d'entrée,
  $X_i(D) = \sum_{j=t_0}^{\infty} x_{ij} D^j$ la transformée de Huffman de la i-ème entrée,
  $\vec{Y}(D) = (Y_1(D), \ldots, Y_n(D))$ le vecteur de sortie
- On a la relation
$$\vec{Y}(D) = \vec{X}(D) \cdot G(D)$$
où $G(D)$ est la matrice de transfert, $G(D) = (g_{ij}(D))_{1 \le i \le k, 1 \le j \le n}$ avec $g_{ij}(D) \in \mathbb{F}_2[D]$.
- Par définition l'ordre du codeur est égal à $m = \max_{1 \le i \le k, 1 \le j \le n} \deg g_{ij}(D)$
- On peut considerer $G(D)$ comme un polynôme sur l'anneau des matrices $k \times n$ sur $\mathbb{F}_2$:
$$G(D) = G_0 + G_1\, D + \ldots + G_m\, D^m$$
avec $G_l = \big([D^l] g_{ij}(D)\big)_{1 \le i \le k, 1 \le j \le n}$ pour tout l, $0 \le l \le m$.

Définition 6.2 *Un* code convolutif *est l'ensemble des séquences produites par un codeur convolutif.*

Exemple : Le codeur binaire $(2, 1, 3)$ de matrice de transfert
$$G(D) = \begin{pmatrix} 1 + D^2 + D^3 & 1 + D + D^2 + D^3 \end{pmatrix}$$
peut être représenté par le circuit donné en figure 6.1.

Exemple : Le codeur binaire $(3, 2, 1)$ de matrice de transfert
$$G(D) = \begin{pmatrix} 1+D & D & 1+D \\ D & 1 & 1 \end{pmatrix}$$
peut être représenté par le circuit donné en figure 6.2.

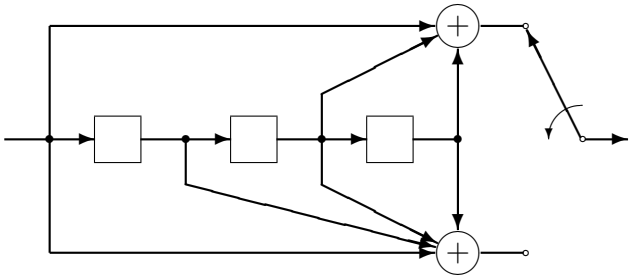


FIG. 6.1 – Codeur convolutif $(2, 1, 3)$

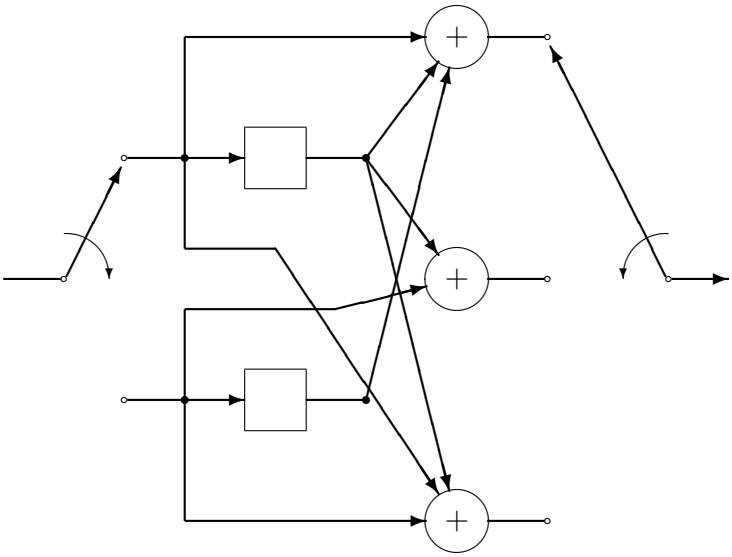


FIG. 6.2 – Codeur convolutif $(3, 2, 1)$

6.2 Diagramme d'état – Treillis de codage

Définition 6.3 *La* mémoire *d'un codeur convolutif, notée K, est définie comme le nombre minimal de registres nécessaire à la réalisation du codeur.*

Pour le codeur de l'exemple 6.1 on a $K = 3$, et pour le codeur de l'exemple 6.1 on a $K = 2$.

L'état d'un codeur convolutif est défini comme la valeur de ses registres. Le nombre d'états possible pour un codeur binaire est donc 2^K.

6.2.1 Diagramme d'état

Le diagramme d'état d'un codeur est un graphe orienté ayant comme nœuds les états du codeur et comme branches les transitions d'un état vers un autre. Une séquence codée correspond à un chemin dans ce graphe.

Dans la figure 6.3 nous représentons le diagramme d'état du code de l'exemple 6.1. Ce codeur admet huit états, et à partir de chaque état deux transitions sont possibles.

Sur chacune des branches du graphe nous donnons également la valeur de l'entrée et la valeur de la sortie, ici l'entrée consiste en un seul bit, et la sortie en 2 bits.

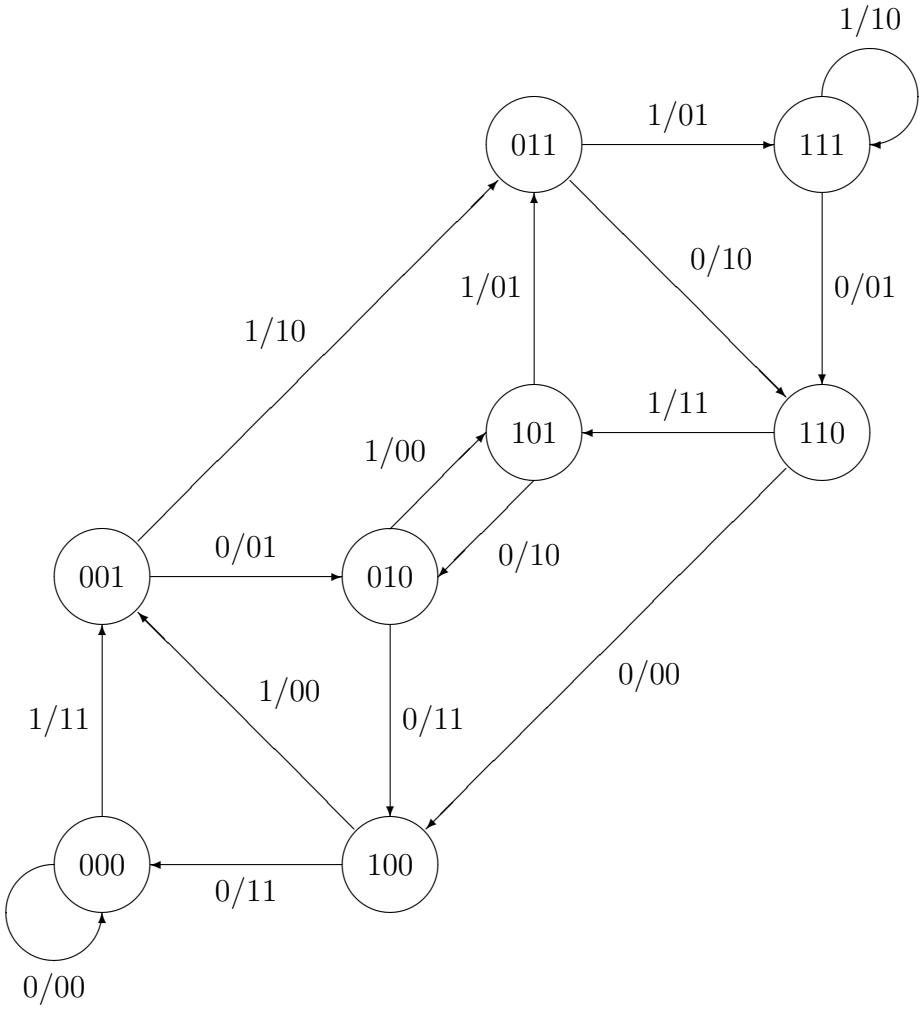


FIG. 6.3 – Diagramme d'état d'un code $(2, 1, 3)$

6.2.2 Treillis de codage

Le treillis de codage d'un codeur convolutif est son diagramme d'état étendu dans le temps, c'est-à-dire que chaque unité de temps posséde son propre diagramme d'état.

Exemple : Prenons un codeur $(3, 1, 2)$ de matrice de transfert

$$G(D) = \begin{pmatrix} 1+D & 1+D^2 & 1+D+D^2 \end{pmatrix}.$$

Pour un tel code le treillis de codage est donné par la figure 6.4. La valeur inscrite sur chaque branche est la valeur de sortie au temps considéré.

Dans la figure 6.4, nous partons de l'état 00, et à chaque étape nous ajoutons les branches correspondant aux transitions possibles.

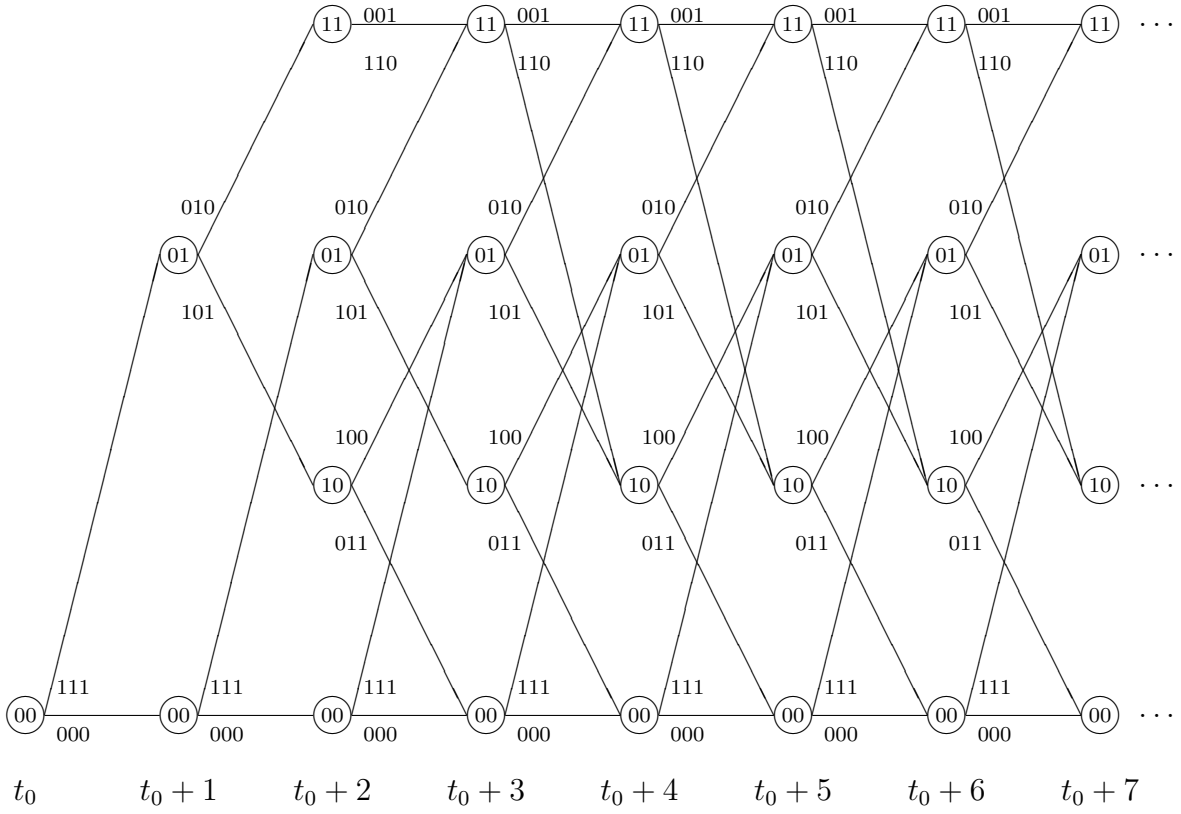


FIG. 6.4 – Treillis de codage d'un code $(3, 1, 2)$

6.3 Décodage

Le décodage d'un code convolutif va consister à trouver parmi les différents chemins du treillis de codage le chemin le plus probable pour aller d'un état donné au temps t_0 à un autre état donné au temps t.

6.3.1 Algorithme de Viterbi

L'algorithme de Viterbi suppose que le nombre d'états du codeur est faible.

On se place dans le cas d'un canal binaire symétrique, la métrique de Hamming permet alors de choisir la séquence la plus probable.

On considère la séquence X_t correspondant à un chemin quelconque s'arrêtant au temps t dans le treillis de codage, on pose $X_t = (x_{t_0}, \ldots, x_t)$, avec $x_i \in \mathbb{F}_2^n$.

Si $Y_t = (y_{t_0}, \ldots, y_t)$, avec $y_i \in \mathbb{F}_2^n$, est la séquence reçue jusqu'au temps t, on peut calculer facilement $d_H(X_t, Y_t)$ à partir de $d_H(X_{t-1}, Y_{t-1})$:

$$d_H(X_t, Y_t) = d_H(X_{t-1}, Y_{t-1}) + d_H(x_t, y_t).$$

(X_{t-1} et Y_{t-1} sont les séquences X_t et Y_t tronquées au temps $t-1$)

L'algorithme de Viterbi :
On suppose que pour chaque état du codeur au temps t, le meilleur chemin est connu ainsi que la distance de Hamming entre la séquence codée correspondant à ce chemin et la séquence recue.

1. On calcule pour chaque état au temps t, tous ses successeur (il y en a 2^k).
2. Chaque état à l'instant $t+1$ peut être atteint de 2^k façons, on conserve le chemin tel que la distance de Hamming à la séquence recue soit minimale.

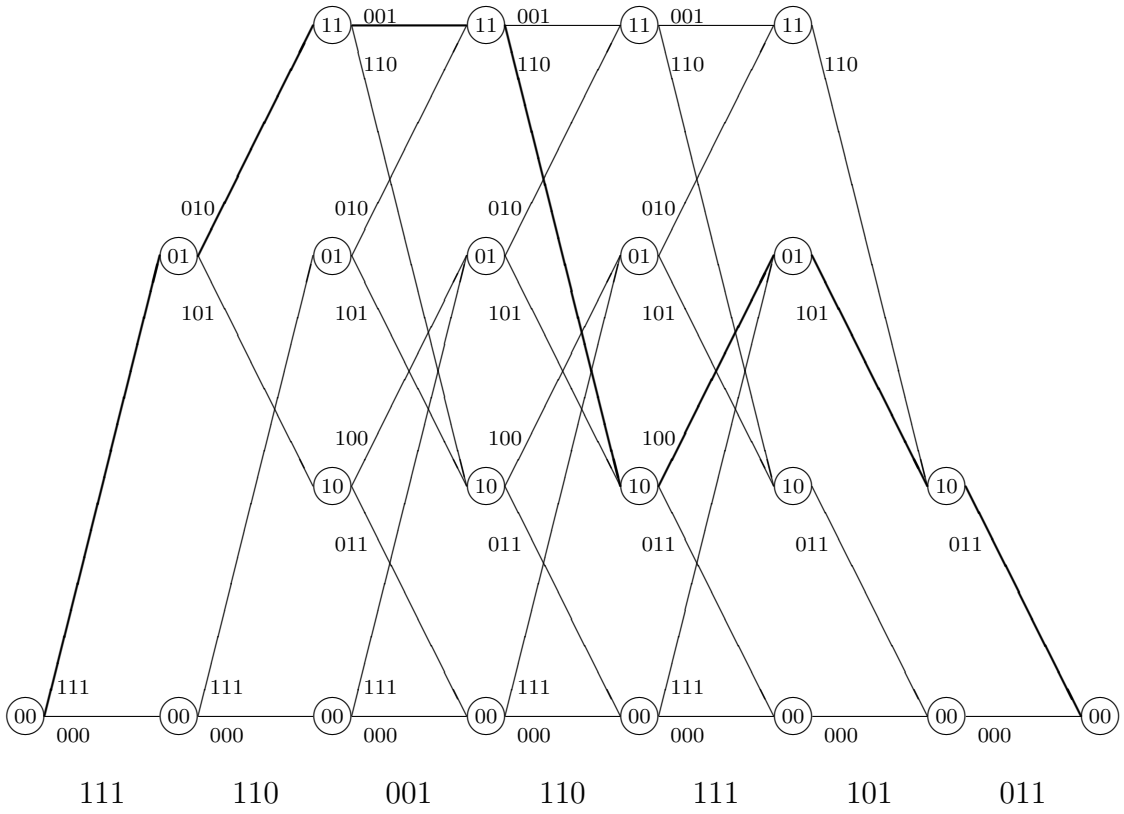


FIG. 6.5 – Exemple de décodage

Exemple : Nous reprenons le codeur de l'exemple 6.2.2. Supposons que la séquence d'information soit $(1, 1, 1, 0, 1, 0, 0)$, avec une valeur des registres initialement nulle, la séquence émise est alors :

$$111 \quad 010 \quad 001 \quad 110 \quad 100 \quad 101 \quad 011 \ .$$

On convient généralement qu'une séquence codée doit ramener les registres à des valeurs nulles, donc la fin de la séquence d'information est constituée de 0. Après passage dans le canal nous recevons

$$111 \quad \underline{1}10 \quad 001 \quad 110 \quad 1\underline{11} \quad 101 \quad 011 \ .$$

La figure 6.5 nous donne l'ensemble des chemins possibles, il suffit alors de déterminer de proche en proche le plus probable.

6.3.2 Décodage souple

Dans le cas ou l'aphabet est binaire, on peut choisir de représenter le "0" par "-1", le "1" restant inchangé. L'ensemble des mots pouvant être reçu à travers le canal est dans l'intervalle réel fermé $[-1, 1]$. On suppose que le canal est sans mémoire, et est tel que la distance euclidienne corresponde au maximum de vraisemblance (c'est-à-dire que plus le mot x est proche du mot y pour la distance euclidienne, plus la probabilité que x ait été émis sachant que y a été reçu est importante).

Dans un tel canal, il est possible d'adapter facilement l'algorithme de Viterbi. En effet, si l'on utilise la métrique euclidienne au lieu de la métrique de Hamming, l'algorithme nous donnera le chemin le plus probable. Le seul point à vérifier est que la distance euclidienne entre deux séquences, à l'instar de la distance de Hamming, peut se calculer de façon incrémentale.

Soient deux séquences $X = (x_0, \dots, x_t, \dots)$ et $Y = (y_0, \dots, y_t, \dots)$, $x_i \in [-1, 1]^n$ et $y_i \in [-1, 1]^n$. Soient X_t et Y_t les même séquences tronquées au rang t, d_E la distance euclidienne, on a

$$d_E^2(X_t, Y_t) = \sum_{i=0}^{t} d_E^2(x_i, y_i).$$

Table des matières