Table des matières

Chapitre 1

Introduction au Codage de Canal

1.1 Introduction

Le problème du codage de source consistait à représenter les symboles de l'alphabet Q par des mots de code binaires (le plus souvent des codes à longueur variable) de façon à avoir un code source aussi compressé que possible. La séquence binaire obtenue est normalement destinée à être transmise sur un canal de transmission. Cependant, celle-ci ne peut pas être transmise directement car le canal introduit du bruit qui peut corrompre l'information. Le codage de canal consiste à faire un transcodage de façon à protéger le code source contre les erreurs. Dans ce chapitre, nous allons étudier les codes dits : codes linéaire en blocs.

1.2 Canaux Discrets Sans Mémoire

Pour étudier le codage de canal nous avons besoin de caractériser d'abord un canal de transmission. Soient un canal de transmission, un alphabet d'entrée au canal $\mathcal{X}$ et un alphabet de sortie du canal $\mathcal{Y}$. La représentation d'un canal de transmission peut se faire comme le montre la figure 1.1, où X est une source d'entrée au canal, Y est la sortie du canal et $p(y_j|x_i)$ représentent les probabilités de transition du canal.

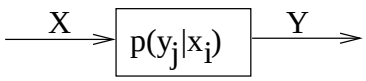

Figure 1.1 : *Arbre de décomposition pour la construction de codes sans préfixe*

Supposons que les alphabets $\mathcal{X}$ et $\mathcal{Y}$ disposent de m et n symboles, respectivement. Le canal est complètement décrit si toutes les probabilités de transition $p(y_j|x_i)$; $i \in [1 \dots m]$ et $j \in [1 \dots n]$, où $p(y_j|x_i)$ est la probabilité que y_j soit reçu sachant que x_i a été transmis, sont connues.

Nous pouvons organiser toutes les probabilités $p(y_j|x_i)$ sous forme d'une matrice pour obtenir la *matrice de transition* du canal.

$$[P(Y|X)] = \begin{bmatrix} p(y_1|x_1) & \cdots & p(y_n|x_1) \\ \vdots & \ddots & \vdots \\ p(y_1|x_m) & \cdots & p(y_n|x_m) \end{bmatrix} \tag{1.1}$$

Un canal de transmission est donc complètement décrit lorsqu'on connaît sa matrice de transition.

Définition 1 *Un canal discret est dit sans mémoire si le dernier symbole reçu $y \in \mathcal{Y}$ ne dépend que du dernier symbole émis $x \in \mathcal{X}$.*

Remarque 1

$$\sum_{j=1}^{n} p(y_j|x_i) = 1 \text{ pour tout } i. \tag{1.2}$$

Pour voir ça, notons que :

$$\begin{aligned} \sum_{j=1}^{n} p(y_j|x_i) &= \sum_{j=1}^{n} \frac{p(y_j, x_i)}{p(x_i)} \\ &= \frac{1}{p(x_i)} \sum_{j=1}^{n} p(y_j, x_i) \\ &= \frac{p(x_i)}{p(x_i)} = 1 \end{aligned}$$

Ceci est aussi intuitivement correct, car pour chaque symbole d'entrée x_i au moins un des symboles de sortie y_j doit apparaître.

Prenons :

$$[P(X)] = [p(x_1)p(x_2) \dots p(x_m)]$$
$$[P(Y)] = [p(y_1)p(y_2) \dots p(y_n)]$$

on a alors la relation matricielle

$$[P(Y)] = [P(X)][P(Y|X)] \tag{1.3}$$

Cette relation matricielle est similaire à la relation scalaire $p(y) = p(x)p(y|x)$ qui nous est déjà familière.

1.2.1 Canaux Remarquables

Quelques types de canaux de transmission sont remarquables compte tenu de leurs caractéristiques.

Canal sans perte Un canal sans perte est un canal dont la matrice de transition ne possède qu'un élément non nul par colonne.

***Exemple* 1**

$$[P(Y|X)] = \begin{bmatrix} \frac{3}{4} & \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{2}{3} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Dans ce type de canal, la simple observation de y_j renseigne directement sur le symbole x_i qui a été transmis, d'où le nom de canal sans perte.

Canal déterministe La matrice de transition d'un canal déterministe ne comporte qu'un élément non nul par ligne. Cet élément doit donc nécessairement être 1.

***Exemple* 2**

$$[P(Y/X)] = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Dans ce type de canal, un symbole x_i transmis, détermine à l'avance quel symbole y_j doit être reçu ; d'où le nom de canal déterministe.

Canal sans bruit Un canal qui est à la fois sans perte et déterministe est un canal sans bruit.

***Exemple* 3**

$$[P(Y/X)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

On voit clairement que la matrice de transition d'un canal sans bruit ne peut être que la matrice identité.

Canal binaire symétrique

$$[P(Y/X)] = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}$$

Le canal binaire symétrique est un modèle intéressant pour décrire le comportement de la plupart des canaux réels. L'entrée est et la sortie sont binaires et p désigne la probabilité d'erreur sur un bit.

1.3 Information mutuelle

Si l'on regarde les sources d'entrée $\mathcal{X}$ et de sortie $\mathcal{Y}$ d'un canal comme étant des sources discrètes de symboles, alors, en plus des entropies d'entrée et de sortie, nous pouvons définir les entropies conditionnelles et conjointes.

$$H(X) = -\sum_{i=1}^{m} p(x_i) \log p(x_i) \tag{1.4}$$

$$H(Y) = -\sum_{j=1}^{n} p(y_j) \log p(y_j) \tag{1.5}$$

$$H(X|Y) = -\sum_{j=1}^{n} \sum_{i=1}^{m} p(x_i, y_j) \log p(x_i|y_j) \tag{1.6}$$

$$H(Y|X) = -\sum_{j=1}^{n} \sum_{i=1}^{m} p(x_i, y_j) \log p(y_j|x_i) \tag{1.7}$$

$$H(X,Y) = -\sum_{j=1}^{n} \sum_{i=1}^{m} p(x_i, y_j) \log p(x_i, y_j) \tag{1.8}$$

Question Montrer que :

$$H(X,Y) = H(X|Y) + H(Y) \tag{1.9}$$

$$H(X,Y) = H(Y|X) + H(X) \tag{1.10}$$

On définit l'information mutuelle comme étant :

$$I(X;Y) = H(X) - H(X|Y) \tag{1.11}$$

Question Vérifier les propriétés suivantes :

1. $I(X;Y) = I(Y;X)$
2. $I(X;Y) \geq 0$
3. $I(X;Y) = H(Y) - H(Y|X)$
4. $I(X;Y) = H(X) + H(Y) - H(X,Y)$

1.4 Capacité d'un canal

Définition 2 *La capacité par symbole d'un canal discret sans mémoire est définie par :*

$$C_s = \max_{\{p(x_i)\}} I(X;Y) \tag{1.12}$$

où le maximum de C_s est recherché sur l'ensemble de toutes les répartitions $\{p(x_i)\}$ sur les symboles de la source X.

***Exemple* 4** *Pour un canal sans perte nous avons $H(X|Y) = 0$ (**montrer ce résultat**) d'où $I(X;Y) = H(X)$. Nous savions déjà que l'entropie $H(X)$ est maximale lorsque les symboles sont équiprobables et alors $H_{max} = \log m$.*

1.4.1 Théorème du codage de canal (deuxième théorème de Shannon)

Théorème 1 *Etant donné une source discrète sans mémoire X d'entropie $H(X)$ (**bit/symbole**) et un canal discret sans mémoire de capacité C_s.*

1. *Si $H(X) \leq C_s$ nous pouvons trouver un codage de canal pour transmettre l'information avec une probabilité d'erreur aussi petite que l'on désire.*
2. *Si $H(X) > C_s$ un tel codage n'existe pas.*

Dans la section suivante, nous supposons que la limite de Shannon ci-dessus est respectée et nous étudierons les codes linéaires par blocs.

1.5 Codes linéaires par blocs

Considérons un code binaire en bloc $\mathcal{C}(n,k)$ qui, à tout bloc de k bits source associe un mot de code de n bits (c'est ce qui est appelé le codage de canal).

$$\underbrace{c_1 c_2 \ldots c_k}_{k \text{ bits}} \underbrace{c_{k+1} c_{k+2} \ldots c_n}_{(n-k)\text{bits}} \tag{1.13}$$

Avec k bits, nous pouvons représenter 2^k mots de codes de données et les données ne différent l'une de l'autre que par un bit. L'idée de base du codage de canal est de rajouter $(n-k)$ bits de parité de façon à avoir des mots de codes qui différent l'un de l'autre par un aussi grand nombre de bits que possible. Ainsi, le bruit aura un effet minime sur les données.

Une mesure qui convient pour estimer la fiabilité d'un code est la distance de Hamming.

1.5.1 Distance de Hamming

Définition 3 *La distance de Hamming entre un mot codé c et un mot r est notée $d_H(c,r)$ et est le nombre de bits qui sont différents entre ces deux mots de code.*

Pour avoir un code efficace, il faut donc maximiser la distance minimale de Hamming d_{min}. Celle-ci est la distance de Hamming entre les deux mots de codes les proches du code $\mathcal{C}$.

Remarque 2 *Si un code est de distance d_{min} on pourra détecter toutes les erreurs de poids inférieur à $d_{min}-1$ et on pourra corriger t erreurs si $t < d_{min}/2$.*

1.5.2 Matrice génératrice d'un code linéaire

Les composantes du mot de code c sont obtenues par combinaison linéaire des composantes du mot de données original d. Si les k premiers bits d'un mot codé sont des bits de données, on dit que le code est systématique.
Considérons un code systématique. Soit $d = [d_1 d_2 \dots d_k]$ le mot de données et soit $c = [c_1 c_2 \dots c_n]$ le mot codé qui lui est associé. Nous pouvons écrire :

$$\begin{aligned}
c_1 &= d_1 \\
c_2 &= d_2 \\
&\vdots \\
c_k &= d_k \\
c_{k+1} &= p_{11}d_1 \oplus p_{12}d_2 \oplus \cdots p_{1k}d_k \\
&\vdots \\
c_n = d_1 &= p_{m1}d_1 \oplus p_{m1}d_2 \oplus \cdots p_{mk}d_k
\end{aligned}$$

où $m = n - k$
Sous forme matricielle, cette relation s'écrit :

$$[c] = [d]G \tag{1.14}$$

où

$$G = \begin{bmatrix} 1 & \cdots & 0 & p_{11} & \cdots & p_{m1} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 1 & p_{1k} & \cdots & p_{mk} \end{bmatrix} \tag{1.15}$$

est appelée la *matrice génératrice* du code.
Notons que G est sous forme :

$$G = \left[I_k P^k\right] \tag{1.16}$$

où I_k est la matrice identité de dimension $k \times k$ et

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1k} \\ \vdots & & \vdots \\ p_{m1} & \cdots & p_{mk} \end{bmatrix} \tag{1.17}$$

P est de dimension $(n-k) \times k$ et P^T est de dimension $k \times (n-k)$ comme attendu.

***Exemple* 5** *Un code $(5, 3)$ est défini par sa matrice génératrice G qui est donnée par*

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} = [I_3 P^T]$$

Trouver tous les mots de code.

1.5.3 Matrice de contrôle

Soit H la matrice définie par

$$H = [P I_m] = [P I_{n-k}] \tag{1.18}$$

H est de dimension $(n-k) \times n$. On a alors

$$H^T = \begin{bmatrix} P^T \\ I_m \end{bmatrix} \tag{1.19}$$

d'où l'on déduit que

$$\begin{aligned} GH^T &= [I_k P^T] \begin{bmatrix} P^T \\ I_{n-k} \end{bmatrix} \\ &= I_k P^T \oplus P^T I_{n-k} \\ &= P^T \oplus P^T \\ &= \mathbf{0} \end{aligned} \tag{1.20}$$

On obtient donc la matrice nulle de dimension $k \times (n-k)$. Maintenant, à partir de la relation $[c] = [d]G$, on peut trouver

$$[c]H^T = [d]GH^T = \mathbf{0} \tag{1.21}$$

La matrice H ainsi définie, est appelée la *matrice de contrôle de parité*.

Remarque 3 *De la relation $[c]H^T = \mathbf{0}$ on voit clairement que tout mot de code valide vérifie cette relation. Si, pour un mot de code c_1 on calcule $[c_1]H^T \neq 0$, on sait alors que c_1 est un code erroné.*

***Exemple* 6** *Pour le code $(5, 3)$ donné à l'exemple ci-dessus et ayant pour matrice génératrice*

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

Trouver H et vérifier que pour tout mot de code valide, on retrouve $[c]H^T = \mathbf{0}$. Vérifier que pour le mot invalide $r = [\ 1 \quad 1 \quad 1 \quad 0 \quad 1\ ]$, on retrouve $rH^T = [\ 0 \quad 1\ ]$.

1.5.4 Syndrome

Considérons le cas où l'émetteur transmet le mot de code c et le récepteur reçoit le mot r. On appelle syndrome, le vecteur $s = [r]H^T$. Si r n'a pas été affecté par le bruit, c'est-à-dire que r reste un mot de code valide, on doit retrouver un syndrome nul. Sinon, r est alors erroné. Soit e le vecteur d'erreur. On a alors :

$$r = c \oplus e \tag{1.22}$$

$$\begin{aligned} s = rH^T &= (c \oplus e)H^T \\ &= cH^T \oplus eH^T \\ &= \mathbf{0} \oplus eH^T \\ &= eH^T \end{aligned} \tag{1.23}$$

Supposons qu'on transmette c et qu'une erreur se produise sur le i^{eme} bit de c. Le vecteur d'erreur est donc donné par :

$$e = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \end{bmatrix} \tag{1.24}$$

Le produit eH^T va résulter en une matrice de dimension $1 \times (n-k)$ identique à la i^{eme} ligne de H^T ou, bien entendu, la i^{eme} colonne de H. Le décodage se fait alors de manière très simple ; on compare le syndrome avec les lignes de H^T et la position de la ligne avec laquelle il coïncide correspond à la position du bit nécessitant une correction dans le code reçu r.

Chapitre 2

Introduction aux corps finis de Galois

2.1 Introduction

Les corps finis de Galois sont l'une des structures algébriques les plus importantes sur lesquelles se base la conception et la compréhension des codes linéaires en bloc. Les corps finis de Galois sont essentiellement des structures de corps, telles que $\mathbb{R}$ ou $\mathbb{C}$ (les corps des nombres réels et des nombres complexes respectivement), avec un nombre fini d'éléments.

2.2 Structure de groupe

Définition 4 *Un groupe est un ensemble d'éléments* $G = \{a, b, c, \ldots\}$ *muni d'une opération* $\oplus$ *vérifiant les axiomes suivants*

1. $\oplus$ *est une loi interne equation :* $\forall a, b \in G, a \oplus b \in G$
2. *Associativité :* $\forall a, b, c \in G, (a \oplus b) \oplus c = a \oplus (b \oplus c)$
3. *Identité :* $\exists$ *dans* G *un élément neutre (identité) noté* 0 *pour lequel* $\forall a \in G, a \oplus 0 = 0 \oplus a = a$
4. *Inverse :* $\forall a \in G$, $\exists$ *dans* G *un élément inverse de* a*, noté* $(-a)$ *tel que* $a \oplus (-a) = 0$

***Exemple* 7** *Les ensembles* $\mathbb{Z}$*,* $\mathbb{R}$*,* $\mathbb{C}$*, et* $\mathbb{F}_2$ *(corps des nombres binaires) munis de l'opération d'addition " + " sont des exemples connus de structures de groupe.*

***Exemple* 8** $\mathbb{F}_2 = \{0, 1\}$ *est un exemple de groupe fini.*

Question Vérifier l'ensemble des axiomes ci-dessus pour $\mathbb{F}_2 = \{0, 1\}$ en interprétant $\oplus$ comme addition *mod* 2.

***Exemple* 9** *L'ensemble des éléments non-nuls de* $\mathbb{R}$ *muni de l'opération de multiplication est une structure de groupe avec* 1 *comme élément neutre, et l'inverse de* a *étant* $(1/a)$

Exemple* 10** *On considère l'ensemble* $G = \{0, 1, \ldots, m-1\}$ *muni de l'opération* $\oplus$ *interprétée comme addition* mod m. *G est une structure de groupe.* ***Quel est l'inverse de $a \in G$ ***?***

Exemple* 11** *L'ensemble* $G = \{1, 2, \ldots, p-1\}$ *muni de l'opération de multiplication* mod p *est une structure de groupe si* p *est un nombre premier.* ***Si p ***n'est pas premier*** G ***est-il une structure de groupe ? lequel des axiomes est violé dans ce cas ?***

2.2.1 Groupe Abélien

Définition 5 *Si* G *est un groupe sous l'opération* $\oplus$ *et si* $\oplus$ *est commutative alors* G *est appelé groupe commutatif ou groupe Abélien.*

2.2.2 Sous-groupe

Définition 6 *Étant donné un groupe* G *et soit* S *un sous-ensemble de* G. S *est un sous-groupe de* G *si* S *est lui-même une structure de groupe*

Théorème 2 *(Théorème de lagrange)*
Pour un groupe fini G *et tout sous-groupe* S *de* G*, nous avons :*
$|S|$ *divise* $|G|$
où $|X|$ *désigne l'ordre de* X *(nombre d'éléments dans* X*).*

Démonstration G est l'union de un ou plusieurs sous-groupes disjoints. Soit C le nombre de sous-groupes. Il s'ensuit que
$|G| = C|S|$

***Exemple* 12** *(sous- groupe cyclique)*
Soit G *un groupe fini et* $a \in G$. *On considère* $S = \{a, 2a, 3a, \ldots\}$
Si $ia = ja$ *pour deux entiers* i *et* j *avec* $j > i$ *alors* $(j-i)a = 0$ *et il existe donc un plus petit entier* $m > 0$ *tel que* $ma = 0$ *du fait que G est fini. L'ensemble* $S = \{0, a, 2a, \ldots, (m-1)a\}$ *est alors un sous-groupe cyclique de* G.
m *est appelé l'ordre de* a*, noté* ordre(a)

Remarque 4 *D'après le théorème de Lagrange, ordre(*a*) divise* $|G|$.

2.3 Structure de corps

Définition 7 *Un corps est un ensemble $\mathbb{F}$ muni d'au moins deux éléments et de deux opérations $\oplus$ et $*$ satisfaisant aux axiomes suivants :*

- *$\mathbb{F}$ muni de $\oplus$ est une structure de groupe Abélien,*
- *$\mathbb{F} - \{0\}$ muni de $*$ est une structure de groupe Abélien,*
- *distributivité : $\forall a, b, c \in \mathbb{F}, (a \oplus b) * c = (a * c) \oplus (b * c)$.*

***Exemple* 13** *Un exemple fondamental de corps fini (appelé corps de Galois) est l'ensemble $\mathbb{Z}_p$ des entiers* mod p *où p est un nombre premier.*
$\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$, p étant premier. " $\oplus$ " est interprétée comme addition mod p*; " $*$ " interprétée comme multiplication* mod p.

Question Vérifier la distributivité.

Théorème 3 *Pour tout nombre premier p, l'ensemble des éléments $\{0, 1, \ldots, p-1\}$ est une structure de corps fini $\mathbb{F}_p$ sous les opérations $\oplus$ et $*$ interprétées comme addition et multiplication* mod p*, respectivement.*

2.4 Polynômes

Les polynômes sur un corps fini $\mathbb{F}_p$ sont définis de manière similaire que sur le corps des nombres réels. Un polynôme $f(x)$ sur un corp fini $\mathbb{F}_p$ est un ensemble d'éléments de la forme

$$f(x) = f_0 + f_1 x + f_2 x^2 + \cdots + f_m x^m \tag{2.1}$$

où $f_i \in \mathbb{F}_p$, $0 \leq i \leq m$; $f_m \neq 0$; $m = deg f(x)$

Toutes les opérations sur les polynômes qui se font sur le corps des nombres réels se font de manière similaire sur le corps $\mathbb{F}_p$ sauf que les opérations sur ce dernier se font *mod p*.

Remarque 5

1. *Un polynôme sur $\mathbb{F}_p$ est irréductible s'il n'a pas de facteur.*
2. *On dit que le polynôme $f(x) = f_0 + f_1 x + f_2 x^2 + \cdots + f_m x^m$ est unitaire si $f_m = 1$*

2.5 Corps des polynômes sur $\mathbb{F}_p$

Soit p un nombre premier et considérons le corps $\mathbb{F}_p$. Supposons que $f(x) \in \mathbb{F}_p[x]$ ($\mathbb{F}_q[x]$ est la notation utilisée pour désigner l'ensemble des polynômes sur un corps $\mathbb{F}_q$) est un polynôme irréductible de degré m. Considérons l'ensemble $\{g(x)\}$ des polynômes de degrés inférieurs ou égals à $m-1$ et définissons l'opération de multiplication de deux polynômes $g(x)$ et $h(x)$ comme étant

$$g(x)h(x) = [g(x)h(x)] \text{mod } f(x); \;\; g(x), h(x) \in \{g(x)\} \tag{2.2}$$

On note $\mathbb{F}_{p^m}$ l'ensemble de ces polynômes.

Théorème 4 *L'ensemble $\mathbb{F}_{p^m}$ muni des opérations d'addition et de multiplication de polynômes modulo un polynôme irréductible $f(x)$ sur $\mathbb{F}_p$ est une structure de corps fini.*

***Exemple* 14** *Considérons le polynôme $f(x) = 1+x+x^2$. $f(x)$ est irréductible sur $\mathbb{F}_2$ et nous avons $deg\,(f(x)) = m = 2$. Nous pouvons donc construire le corps des polynômes de degrés ≤ 1. L'ensemble de ces polynômes est $\mathbb{F}_{2^2} = \{0, 1, x, x+1\}$.*

Question *Obtenir les tables d'addition et de multiplication des éléments de ce corps.*

2.6 Sous-corps

Définition 8 *Un sous-ensemble $\mathbb{G}$ d'éléments d'un corps fini $\mathbb{F}$, muni des opérations dans $\mathbb{F}$, est un sous-corps de $\mathbb{F}$ si $\mathbb{G}$ est une structure de corps.*

Le plus petit sous-corps d'un corps fini $\mathbb{F}_{p^m}$ est le corps $\mathbb{F}_p$ des entiers *mod* p, où p est un nombre premier. $\mathbb{F}_p$ est alors appelé le sous-corps premier ou corps de base du corps fini $\mathbb{F}_{p^m}$. p est appelé la caractéristique du corps.

Dans un corps fini de caractéristique p nous pouvons montrer le résultat suivant :

$$(\alpha+\beta)^p = \alpha^p + \beta^p; \;\; \alpha, \beta \in \mathbb{F}_{p^m} \tag{2.3}$$

Ce résultat peut s'étendre par récurrence à :

$$(\alpha+\beta)^{p^n} = \alpha^{p^n} + \beta^{p^n}; \;\; \alpha, \beta \in \mathbb{F}_q; \;\; q = p^m \tag{2.4}$$

Question Montrer le premier résultat ci- dessus.

2.7 Propriétés fondamentales des corps finis

Nous avons vu dans la section précédente que pourvu que l'on dispose d'un polynôme $f(x)$ de degré m, irréductible sur un corps de base $\mathbb{F}_p$, nous pouvons construire un corps fini $\mathbb{F}_{p^m}$ de tous les polynômes de degrés inférieurs ou égal à m. Il reste maintenant à s'assurer de l'existence d'un polynôme irréductible pour tout entier $m \geq 1$.

Notons qu'un polynôme $f(x) \in \mathbb{F}_{p^m}[x]$ est irréductible sur $\mathbb{F}_p$ s'il n'a pas de racines dans $\mathbb{F}_p$. Puisque $\mathbb{F}_{p^m}$ est fini nous allons d'abord chercher un polynôme sur $\mathbb{F}_{p^m}[x]$ dont les racines sont les éléments de $\mathbb{F}_{p^m}$ puis nous étudierons sa factorisation sur $\mathbb{F}_p$.

2.7.1 Factorisation sur $\mathbb{F}_{p^m}$

Soit $q = p^m$ et β un élément quelconque de $\mathbb{F}_q$. Nous savons que l'ensemble des éléments non-nuls de $\mathbb{F}_q$ forme une structure de groupe pour la multiplication. Considérons le sous-ensemble $\{\beta, \beta^2, \beta^3, \ldots\}$. Puisque le groupe est fini, il existe un plus petit entier n tel que $\beta^n = 1$. Ce sous-ensemble est une structure de groupe et donc d'après le théorème de lagrange, n divise $q-1$. Puisque $\beta^n = 1$ nous déduisons que β est une racine du polynôme $x^n - 1 \in \mathbb{F}_q[x]$. Aussi puisque n divise $q-1$, β est aussi racine du polynôme $x^{q-1} - 1 \in \mathbb{F}_q[x]$. Chaque élément $\beta \in \mathbb{F}_q$ a un certain ordre et est donc dans un sous-group cyclique de $\mathbb{F}_q$. Tous les éléments de $\mathbb{F}_q$ vérifient l'équation $x^{q-1} - 1 = 0$. Nous résumons donc ces résultats par :

Théorème 5 *Dans un corps fini $\mathbb{F}_q$, tout élément non-nul $\beta \in \mathbb{F}_q$ satisfait à $\beta^{q-1} = 1$ et il a donc un ordre multiplicatif qui divise $q-1$. Les éléments non-nuls de $\mathbb{F}_q$ sont les $q-1$ racines distinctes du polynôme $x^{q-1} - 1 \in \mathbb{F}_q[x]$;*

$$x^{q-1} - 1 = \prod_{\beta \in \mathbb{F}_q;\ \beta \neq 0} (x - \beta) \tag{2.5}$$

Avec une légère modification pour inclure l'élément nul du corps $\mathbb{F}_q$, nous avons :

$$x^{q-1} - x = \prod_{\beta \in \mathbb{F}_q} (x - \beta) \tag{2.6}$$

L'équation $x^q - x = 0$ est appelée l'équation caractéristique du corps $\mathbb{F}_q$.

2.7.2 Élément primitif

Définition 9 *Un élément primitif β d'un corps $\mathbb{F}_q$ est un élément dont l'ordre multiplicatif est égal à $q-1$.*

Les différentes puissances de β décrivent alors tous les éléments de $\mathbb{F}_q$ et nous pouvons écrire :

$$x^{q-1} - 1 = \prod_{i=1}^{q-1}(x - \beta^i) \tag{2.7}$$

où β est un élément primitif.
Nous pouvons montrer que tout corps fini $\mathbb{F}_q$ a au moins un élément primitif.

2.8 Factorisation de $x^{q-1} - 1$ sur le corps de base $\mathbb{F}_p$

On énonce le résultat fondamental suivant :

Théorème 6 *Pour un corps $\mathbb{F}_q$ avec $q = p^m$ éléments, un sous-corps existe avec p^n éléments si et seulement si n devise m. $x^{q-1} - 1$ se factorise sur $\mathbb{F}_p$ en produit de polynômes unitaires irréductibles de degrés m et de chaque degré n divisant m.*

Ainsi, ce théorème montre clairement que des corps finis $\mathbb{F}_q$ avec $q = p^m$ éléments existent pour tout nombre premier p et tout entier positif m. Il montre aussi que des polynômes irréductibles sur $\mathbb{F}_p$ existent et indique aussi la factorisation de $x^{p^m-1} - 1$ en polynômes irréductibles sur $\mathbb{F}_p$.

Chapitre 3

Introduction aux codes linéaires en bloc

3.1 Introduction

Dans le premier chapitre nous avions déjà introduit les codes linéaires en bloc mais nous nous sommes restreints aux codes binaires ; c'est-à-dire que les codes décrits sont définis sur $\mathbb{F}_2$. Dans ce chapitre, nous allons nous intéresser aux codes définis sur un corps fini quelconque $\mathbb{F}_q$. Les codes les plus intéressants dans cette classe de codes sont les codes de Reed-Solomon (RS).

3.2 Codes linéaires sur un corps fini

Définition 10 *Un code linéaire en bloc $\mathcal{C}(n,k)$ sur un corps fini $\mathbb{F}_q$ est un sous-espace de dimension k de l'espace vectoriel $(\mathbb{F}_q)^n$ de tous les n - uplets sur $\mathbb{F}_q$.*

Formellement, supposons que l'on dispose d'une base de k vecteurs $\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_k$, de dimensions n, linéairement indépendants. Un code $\mathcal{C}(n,k)$ est généré en formant toutes les combinaisons linéaires possibles des vecteur de cette base.

$$\mathcal{C} = \left\{ \sum_{j=1}^{k} a_j \mathbf{g}_j, \;\; a_j \in \mathbb{F}_q; \;\; 1 \leq j \leq k \right\} \tag{3.1}$$

Ceci nous montre donc que $\mathcal{C}$ contient q^k mots de codes distincts.

***Exemple* 15** *On considère le corps $\mathbb{F}_3 = \{0,1,2\}$ avec 2 vecteurs générateurs $\mathbf{g}_1 = (1110)$ et $\mathbf{g}_2 = (0121)$. On peut alors générer le code $\mathcal{C}$ contenant les 9 quadruplets ci-dessous. Notons*

que $\mathcal{C}$ est un code $(4, 2)$.

$$\mathcal{C} = \{0000, 1110, 2220, 0121, 1201, 2011, 0212, 1022, 2102\}$$

Soit $c \in \mathcal{C}$ un mot de code quelconque. Puisque $\mathbb{F}_q$ est une structure de corps, c'est aussi une structure de groupe. On déduit que $\mathcal{C} - c = \mathcal{C}$. Or $\mathcal{C} - c$ constitue l'ensemble des distances de hamming entre les différents mots de code de $\mathcal{C}$. La distance minimale de Hamming d est donc égale au plus petit poids d'un mot de code non nul de $\mathcal{C}$.

Remarque 6 *On notera par la suite par $\mathcal{C}(n, k, d)$ un code linéaire en bloc $\mathcal{C}(n, k)$ ayant une distance de Hamming minimale d.*

Un code linéaire $\mathcal{C}(n, k, d)$ peut corriger t erreurs sur les symboles si $2t < d$. Pour rechercher un bon code, il semble donc naturel de considérer comme tel celui qui a le meilleur pouvoir de correction. Cependant, la complexité du décodage est aussi importante.

3.2.1 Borne du singleton

Théorème 7 *Tout code linéaire en bloc $\mathcal{C}(n, k, d)$ satisfait à la borne du singleton*

$$k + d \leq n + 1 \tag{3.2}$$

Question Montrer ce résultat

Un code $\mathcal{C}(n, k, d)$ satisfaisant à la borne du singleton avec égalité est appelé *code optimal*

$$k + d = n + 1; \text{ code optimal} \tag{3.3}$$

3.3 Codes RS

Les codes RS sont des codes optimaux. Pour tous paramètres de code $n, k, d = n - k + 1$, il existe un code $RS(n, k, d)$ sur le corps fini $\mathbb{F}_q$ pour $n \leq q + 1$. Dans cette section, nous allons donc illustrer les codes RS pour $n = q$, puis nous considérerons les codes RS avec $n = q - 1$ pour lesquelles nous allons montrer quelques propriétés intéressantes.

3.3.1 Construction de codes RS

Soit $(f_0, f_1, \ldots, f_{k-1}),\ \ f_j \in \mathbb{F}_q$ et $0 \leq j \leq k-1$ le vecteur des symboles d'information. Pour chaque vecteur de symboles d'information nous pouvons associer le polynôme $f(z) \in \mathbb{F}_q[z]$

$$f(z) = f_0 + f_1 z + \cdots + f_{k-1} z^{k-1} \tag{3.4}$$

Soient $\beta_1, \beta_2, \ldots, \beta_q$ les q éléments distincts de $\mathbb{F}_q$ et α une racine primitive de $\mathbb{F}_q$. Nous pouvons arranger les éléments $\beta_i,\ 1 \leq i \leq q$ dans l'ordre suivant :

$$\beta_1 = 0, \beta_2 = \alpha, \ldots, \beta_j = \alpha^{j-2}, \ldots, \beta_q = \alpha^{q-2} = \alpha^{-1}$$

Pour construire un code $RS(n = q, k, d = n - k + 1)$ on associe au vecteur d'information le vecteur $(f(\beta_1), f(\beta_2), \ldots, f(\beta_n))$ sur $\mathbb{F}_q$ dont les composantes $f(\beta_i)$ sont les évaluations du polynôme $f(z)$ pour chaque $\beta_i \in \mathbb{F}_q$. Donc :

$$f(\beta_i) = \sum_{j=0}^{k-1} f_j \beta_i^j \in \mathbb{F}_q,\ \ 1 \leq i \leq q \tag{3.5}$$

Pour générer ce code, on voit donc qu'on peut prendre comme polynômes générateurs $g_j(z),\ \ 0 \leq j \leq k-1$ ou la base des k vecteurs générateurs suivants :

$$\begin{aligned}
\mathbf{g}_0 &= (1, 1, 1, \ldots, 1) \\
\mathbf{g}_1 &= (0, 1, \alpha, \ldots, \alpha^{-1}) \\
\mathbf{g}_2 &= (0, 1, \alpha^2, \ldots, \alpha^{-2}) \\
&\vdots \\
\mathbf{g}_{k-1} &= \left(0, 1, \alpha^{k-1}, \ldots, \alpha^{-(k-1)}\right)
\end{aligned}$$

Notons qu'on peut former q^k polynômes d'information distincts.

Théorème 8 *Les q^k q - uplets générés par l'association $f(z) \mapsto \{f(\beta_i),\ \ \beta_i \in \mathbb{F}_q\}$ lorsque $f(z)$ décrit tous les q^k polynômes sur $\mathbb{F}_q$ de degrés inférieurs à k forment un code $\mathcal{C}(n, k, d = n - k + 1)$ linéaire optimal sur $\mathbb{F}_q$*

Démonstration La linéarité est évidente. En effet, si $f_1(z)$ et $f_2(z)$ sont des polynômes d'information alors

$$(f_1(z) + f_2(z)) \mapsto \{f_1(\beta_i),\ \ \beta_i \in \mathbb{F}_q\} + \{f_2(\beta_i),\ \ \beta_i \in \mathbb{F}_q\} \tag{3.6}$$

De même,

$$(\beta f(z)) \mapsto \beta\{f(\beta_i),\ \ \beta_i \in \mathbb{F}_q\} \tag{3.7}$$

Pour montrer l'optimalité, il faut noter qu'un mot de code ne peut avoir plus de $k-1$ composantes égales à zero. Donc le poids minimal des mots de code est $n-k+1$ (**pourquoi un mot de code ne peut avoir au plus que $k-1$ composantes nulles ?**). Nous déduisons que la distance minimale du code vérifie la relation $d \geq n-k+1$. D'après la borne du singleton, $d \leq n-k+1$. D'où $d = n-k+1$ et le code est donc optimal.

***Exemple* 16** *Reconsidérons le corps fini que nous avons construit auparavant en utilisant le polynôme irréductible x^2+x+1 sur $\mathbb{F}_2$. $\mathbb{F}_4 = \{0, 1, x, x^2 = x+1\}$. Posons $\beta_1 = 0, \beta_2 = 1, \beta_3 = x, \beta_4 = x+1$. Notons que $\beta_1 = 0$ est une racine primitive. Un code RS avec $n = q = 4$ et $k = 2$ est généré par l'association*

$$f(z) = f_0 + f_1 z \mapsto (f(\beta_1), f(\beta_2), f(\beta_3), f(\beta_4))$$

On peut voir que le code peut être généré par les deux polynômes $g(z) = 1$ et $g(z) = z$ ou par les deux vecteurs générateurs $(1\ 1\ 1\ 1)$ et $(0\ 1\ x\ x^2)$

Question Énumérer les 16 différents mots de code.

3.3.2 Codes RS poinçonnés

Avant d'aborder les codes RS poinçonnés examinons d'abord le théorème suivant qui s'applique sur tout corps (fini ou pas) $\mathbb{F}$.

Théorème 9 *Soit $\mathbb{F}$ un corps avec 1 comme élément neutre pour la multiplication et ω une racine n^{eme} de 1 dans $\mathbb{F}$ autre que 1 ; c'est-à-dire $\omega^n = 1$ et $\omega \neq 1$. Nous avons alors*

$$\sum_{j=0}^{n-1} \omega^j = 0 \tag{3.8}$$

Démonstration

$$\omega \sum_{j=0}^{n-1} \omega^j = \sum_{j=1}^{n} \omega^j = \sum_{j=0}^{n-1} \omega^j \tag{3.9}$$

Ceci se traduit par $\omega S = S$ avec $S = \sum_{j=0}^{n-1} \omega^j$. Donc $S = 0$ si $\omega \neq 1$. Nous pouvons donc noter en passant que $(\omega^i)^n = \omega^{in} = (\omega^n)^i = 1$. Donc si ω est une racine n^{eme} de 1 alors ω^i l'est aussi pour tout i. Ce résultat nous amène alors au corollaire suivant

Corollaire 1 *Soit $\mathbb{F}$ un corps et $\omega^n = 1$ et $\omega^i \neq 1$ pour l'entier $1 \leq i < n$. alors :*

$$\sum_{j=1}^{n} \omega^{ij} = \begin{cases} n; & \textit{si} \quad i = 0\ mod\ n \\ 0; & \textit{sinon} \end{cases} \tag{3.10}$$

Question Montrer ce corollaire

Considérons maintenant le code $RS(n = q-1, k, d = n-k+1)$ obtenu en poinçonnant le code $RS(n = q, k, d = n-k+1)$ de la section précédente dans sa première composante correspondant à $\beta_1 = 0$. Un mot de code $F = (f(1), f(\alpha), \ldots, f(\alpha^{-1})$ peut être obtenu à partir d'un vecteur d'information $(f_0, f_1, \ldots, f_{k-1})$ par la relation

$$F_i = \sum_{j=0}^{k-1} f_j \alpha^{ij}; \;\; 1 \leq i \leq q-1 \tag{3.11}$$

Si on note que si on travaille sur un corps fini $\mathbb{F}_q$ alors $n = q-1 = -1$ puisque $q = 0$ dans $\mathbb{F}_q$ (structure de groupe pour l'addition). On peut retrouver le vecteur d'information par la relation

$$f_j = -\sum_{i=0}^{n-1} F_i \alpha^{-ij}; \;\; 1 \leq i \leq q-1 \tag{3.12}$$

Question Vérifier ce résultat.

Remarque 7 *Puisque $f_j = 0$ pour $k \leq j \leq n-1$ nous avons :*

$$\sum_{i=0}^{n-1} F_i \alpha^{-ij} = 0; \;\; k \leq j \leq n-1 \tag{3.13}$$

3.3.3 Propriétés des codes RS poinçonnés

Soit $F(z) \in \mathbb{F}_q[z]$ où $F(z) = F_0 + F_1 z + \cdots + F_{n-1} z^{n-1}$ est le polynôme de degré $n-1$ correspondant à un mot de code $(F_0, F_1, \ldots, F_{n-1})$. De l'équation de la remarque ci-dessus on voit que α^{-j} est une racine de $F(z)$ pour $k \leq j \leq n-1$. On peut écrire $\alpha^{-j} = \alpha^{(q-1)-j}$. Ceci veut dire alors que $\alpha^j \in \mathbb{F}_q$ est une racine de $F(z)$ pour $1 \leq j \leq n-k$. Donc $F(z)$ a $n-k$ facteurs de la forme $z - \alpha^j$ $(1 \leq j \leq n-k)$ ce qui veut dire que $F(z)$ est divisible par le polynôme générateur

$$g(z) = \prod_{j=1}^{n-k} (z - \alpha^j) \tag{3.14}$$

On conclue donc que le code RS poinçonné est caractérisé par l'ensemble des polynômes $F(z)$ divisibles par le polynôme générateur $g(z)$. On peut voir qu'on peut former ainsi q^k polynômes distincts. Nous résumons par le théorème suivant :

Théorème 10 *Le code RS poinçonné $RS(n = q-1, k, d = n-k+1)$ sur $\mathbb{F}_q$ correspond à l'ensemble des q^k polynômes de degrés inférieurs à k*

$$F(z) = \{g(z)h(z); \;\; \textit{deg } h(z) < k\} \tag{3.15}$$

où $g(z)$ est le polynôme générateur $g(z) = \prod_{j=1}^{n-k}(z - \alpha^j)$

3.3.4 Cyclicité des codes RS poinçonnés

La cyclicité veut dire que si $F = (F_0, F_1, \ldots, F_{n-1})$ est un mot de code alors $F' = (F_{n-1}F_0, F_1, \ldots, F_{n-2})$ est aussi un mot de code.

Question Montrer que $F'(z) = zF(z) - F_{n-1}(z^n - 1)$. Donc $F'(z)$ est divisible par $g(z)$ et est un mot de code.

Chapitre 4

Introduction aux Codes Convolutionnels

4.1 Introduction

Les codes linéaires sont principalement divisés en deux classes : les codes linéaires en bloc et les codes convolutionnels. Les codes RS que nous avons étudié dans le chapitre 3 sont l'un des plus importants représentants de la classes des codes en bloc. Dans ce chapitre, nous allons considérer la seconde classe qui est celle des codes convolutionnels. Ceux-ci sont des codes à mémoire, ce qui veut dire que la sortie du codeur dépend non seulement de l'entrée présente mais aussi de m (mémoire) entrées précédentes. Chaque sortie s'obtient par convolution d'une suite d'entrée et d'un vecteur générateur, caractéristique du codeur.

4.2 Principes

Pour illustrer le principe d'un code convolutionnel, nous allons prendre un code de rendement $1/n$, où n est un entier. Un code de rendement $1/n$ est un code qui, pour chaque bit d'entrée fait correspondre n bits de sortie. Chaque bit de sortie s'obtient par convolution entre la séquence d'entrée et un vecteur générateur. Pour un code de rendement $1/n$, si les n sorties du codeur dépendent de l'entrée présente et de m entrées précédentes alors une mémoire (registre à décalage) de $K = m + 1$ cellules est nécessaire pour réaliser les opérations de convolutions. Les n vecteurs générateurs sont donc de longueur K. m est appelé la *mémoire du code* et K sa *longueur de contrainte*.

***Exemple* 17** *Un exemple de code convlutionnel de rendement* $1/3$ *et de longueur de*

contrainte $K = 3$ est illustré par la figure 4.1

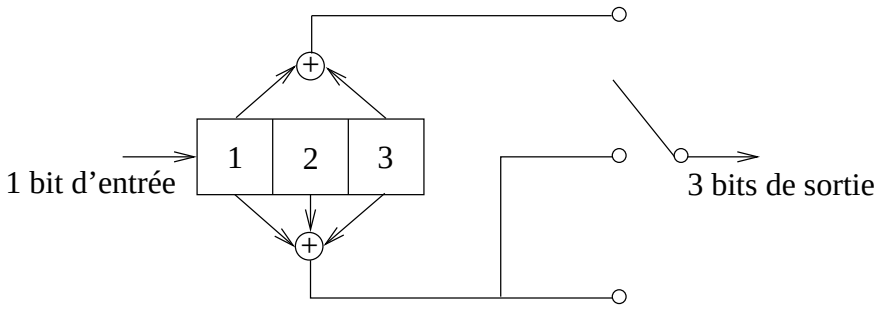

Figure 4.1 : *Exemple de codeur convolutionnel*

Les 3 sorties du codeur sont multiplexées, ce qui veut dire donc que la période d'un bit d'entrée est 3 fois celle d'un bit de sortie.

4.2.1 Formulation polynomiale

Un code convolutionnel de rendement $1/n$ est complètement décrit par ses n polynômes générateurs. Appelons $\mathbf{g}_i, i \in [1 \dots n]$ les n polynômes générateurs. Les expressions des n sorties du codeur s'écrivent

$$y_i(t) = \sum_{j=0}^{K-1} \mathbf{g}_i(j) x(t-j); i \in [1 \dots n] \tag{4.1}$$

où $x(t)$ est la séquence d'entrée. Pour l'exemple ci-dessus, les 3 sorties du codeur s'écrivent :

$$y_1(t) = x(t) \oplus x(t-2) \tag{4.2}$$

$$y_2(t) = x(t) \oplus x(t-1) \oplus x(t-2) \tag{4.3}$$

$$y_3(t) = x(t) \oplus x(t-1) \oplus x(t-2) \tag{4.4}$$

Pour le codeur de cet exemple, 3 vecteurs générateurs sont nécessaires pour définir le code. Ceux-ci sont :

$$\mathbf{g}_1 = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \qquad \mathbf{g}_2 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \qquad \mathbf{g}_3 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{4.5}$$

Notons que la longueur des vecteurs générateurs est égale à la longueur de contrainte.

Le plus souvent, dans la littérature, les vecteurs générateurs sont notés en octal pour une représentation plus compacte. Les trois vecteurs générateurs ci-dessus seront ainsi notés respectivement par $5, 7, 7$. Les vecteurs générateurs peuvent aussi être donnés sous forme de fonctions génératrices ; chaque vecteur correspond alors à un polynôme de degré $K - 1$

***Exemple* 18** $g_1(x) = 1 + x^2$; $g_2(x) = 1 + x + x^2$; $g_3(x) = 1 + x + x^2$

Un codeur convolutionnel peut aussi être vu comme une série de n filtres. Les polynômes générateurs sont alors vus comme les réponses impulsionnelles des n filtres du codeur. Par conséquent, le codeur peut être décrit par un vecteur de n fonctions de transfert en utilisant la transformée en Z. Ainsi, la séquence d'entrée est

$$X(z) = \sum_{i=0}^{\infty} x(i) z^{-i} \tag{4.6}$$

Les n fonctions de transfert sont

$$G_i(z) = \sum_{j=0}^{K-1} g_i(j) z^{-j}; \;\; i \in [1 \dots n] \tag{4.7}$$

et les n sorties du codeur sont

$$Y_i(z) = \sum_{i=0}^{\infty} y_i(j) z^{-i}; \;\; i \in [1 \dots n] \tag{4.8}$$

ou

$$Y_i(z) = X(z) G_i(z) \tag{4.9}$$

sous forme matricielle, on écrit

$$[Y_i(z)] = X(z)[G_i(z)] \tag{4.10}$$

4.2.2 Formulation matricielle

Considérons les n vecteurs générateurs du code qu'on organise sous forme de matrice avec n lignes et K colonnes comme suit

$$G = \begin{bmatrix} g_{11} & \cdots & g_{1K} \\ \vdots & \ddots & \vdots \\ g_{n1} & \cdots & g_{nK} \end{bmatrix} \tag{4.11}$$

Considérons un bloc de l bits d'entrée. La relation matricielle entre le bloc de $L = n(l + K - 1)$ bits de sortie s'écrit comme

$$[Y] = [x]G \tag{4.12}$$

G est la matrice génératrice du code est s'obtient comme

$$G = \begin{bmatrix} C_1 & C_2 & \cdots & C_K & & \\ & C_1 & C_2 & \cdots & C_K & \\ & & & \ddots & & \\ & & & C_1 & C_2 & \cdots & C_K \end{bmatrix} \tag{4.13}$$

La taille de la matrice génératrice G est $l \times [n(l + K - 1)]$

***Exemple* 19** *Reprenons notre exemple et considérons pour simplifier seulement les deux sorties y_1 et y_2. Les polynômes générateurs pour ce code sont*

$$\mathbf{g}_1 = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}; \ \textit{et} \ \ \mathbf{g}_2 = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \tag{4.14}$$

Réécrivons-les sous forme de matrice

$$M = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{4.15}$$

On extrait ainsi les 3 vecteurs $C_1 = [\ 1 \quad 1\ ]$, $C_2 = [\ 0 \quad 1\ ]$ et $C_3 = [\ 1 \quad 1\ ]$ qui nous serviront pour construire la matrice génératrice.
Si on considère maintenant un bloc d'entrée de 4 bits $\begin{bmatrix} x(t) & x(t-1) & x(t-2) & x(t-3) \end{bmatrix}$, la matrice génératrice sera

$$G = \begin{bmatrix} C_1 & C_2 & C_3 & & \\ & C_1 & C_2 & C_3 & \\ & & C_1 & C_2 & C_3 \end{bmatrix} \tag{4.16}$$

qui est égale à

$$G = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & & & \\ & & 1 & 1 & 0 & 1 & 1 & 1 & \\ & & & & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{4.17}$$

4.3 Distance libre d'un code convolutionnel

La distance libre d'un code convolutionnel est la plus petite distance non nulle entre deux séquences codées. La capacité de correction d'erreurs d'un code dépend donc de sa distance libre. Remarquons qu'il n'existe pas de méthode algébrique pour rechercher les bons codes. Par conséquent, des travaux de simulation ont été conduit par des chercheurs qui ont catalogué les meilleurs codes qu'ils ont découvert. Les résultats sont largement publiés dans la littérature.

4.4 Représentations des codes convolutionnels

Nous pouvons distinguer trois représentations intéressantes pour un code convolutionnel : la représentation par diagramme d'états, la représentation par diagramme en arbre et la représentation par treillis. La représentation en treillis est probablement la plus intéressante du fait qu'elle soit utilisée pour expliciter le décodage par le célèbre algorithme de Viterbi.

4.4.1 Diagramme d'états

Le codeur peut être vu comme une machine séquentielle qui est, à un instant donné, dans un état parmi un nombre fini M d'états possibles. Les états possibles sont représentés graphiquement par des points ou des cercles. Les transitions d'un état à un autre sont représentées par des flèches qui portent l'indication sur l'entrée qui provoque la transition et la sortie produite par le codeur. Les M états possibles reflètent les contenus possibles de la mémoire du code. Nous représentons ci-dessous (figure 4.2) le diagramme d'états pour le codeur de l'exemple de la section précédente.

4.4.2 Diagramme en arbre

On représente à travers un tel diagramme l'évolution au cours du temps des états du codeur et on représente en conséquence sur les branches de l'arbre les sorties possibles correspondantes. On suppose que le codeur part de l'état initial nul. Cette représentation est illustrée par la figure 4.3.

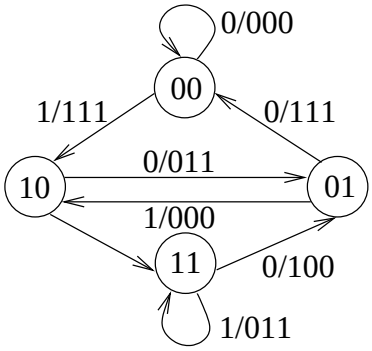

Figure 4.2 : *Diagramme d'états d'un code convolutionnel*

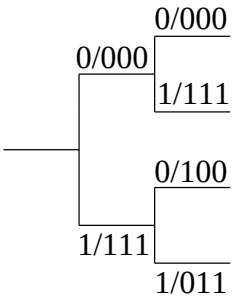

Figure 4.3 : *Représentation en arbre d'un code convolutionnel*

4.4.3 Diagramme en treillis

Cette représentation est équivalente à la représentation par diagramme d'états étalés dans le temps. Pour l'exemple de la section précédente, le treillis se présente comme l'indique la figure 4.4.

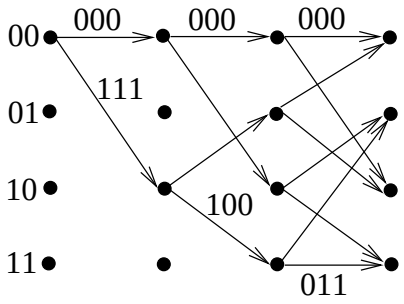

Figure 4.4 : *Diagramme en trellis d'un code convolutionnel*

4.5 Décodage des codes convolutionnels : Algorithme de Viterbi