Académie de Montpellier
Université Montpellier II
Sciences et Techniques du Languedoc

MÉMOIRE DE STAGE DE MASTER M2

effectué au Laboratoire d'Informatique de Robotique
et de Micro-électronique de Montpellier

Spécialité : **IFPRU Informatique à Finalité Professionnelle et Recherche Unifiées**

LA STEGANOGRAPHIE MODERNE : L'ART DE LA COMMUNICATION SECRETE

par **Hugo ALATRISTA SALAS**

Date de soutenance : **21 juin 2010**

Sous la direction de :
Marc CHAUMONT
William PUECH

Table des matières

Table des figures

Liste d'algorithmes

Résume

Ces dernières années, la stéganographie, l'art de la communication secrète, a beaucoup changé. Certains événements mondiaux comme l'attentat du *World Trade Center* le 11 de septembre 2001 ont réouvert l'intérêt pour cet art.
De nouvelles techniques et méthodes apparaissent en stéganographie, chacune d'elles plus performante que la précédente. Ce stage a pour but l'étude et la compression d'une de ces techniques : la stéganographie basée sur le codage par syndrome BCH.
L'insertion de données cachées en utilisant le codage BCH a été introduit par Schönfeld et Winkler [14] ensuit de nouveaux articles sont apparus. Deux de ces articles [1] et [2] sont utilisés comme référence dans ce travail.
Ce rapport de stage contient une étude détaillée du fonctionnement de la technique d'insertion de données cachées en utilisant la technique de codage par syndrome ainsi que des exemples et une implémentation.

Keywords : Stéganographie, BCH, codage par syndrome, codage matriciel, *matrix embedding*, *look-up-tables*.

Introduction

≪Un effet essentiel de l'élégance est de dissimuler leurs moyens≫

Honoré de Balzac.

La stéganographie ou *l'art de l'occulte* est une science ancienne, mais des applications dans le monde digital nous amènent quelques années en arrière.

La stéganographie digitale a pour but de dissimuler un message secret dans un médium insoupçonné appelé *médium de couverture*, qui peut être par exemple une image, un son, une vidéo. Cette dissimulation doit se faire en modifiant le médium de couverture, mais avec la contrainte de minimiser le nombre de coefficients modifiés pour la rendre imperceptible.

Une solution à cette contrainte a été introduite par Richard E. Crandall avec la technique *Matrix Embedding* ou *codage matriciel* [4] et qui a été utilisée en stéganographie en 1998. Quelques années plus tard (2001), cette technique a été implémentée par Andreas Westerfield [1] avec l'algorithme *F5*.

Ce stage vise à étudier la stéganographie basée sur les codes BCH [2]. Les codes BCH sont un type de codes correcteurs d'erreurs qui non seulement permettent de vérifier l'intégrité de l'information qui traverse un canal de communication, mais peuvent aussi corriger les bits erronés. L'objectif de ce stage est la compréhension de cette technique stéganographique. Ce projet de stage prend comme référence deux études précédentes décrites dans les articles [1] et [2] publiées l'année 2009.

Je m'emploierai dans un premier temps à préciser les principales caractéristi-ques de la stéganographie. L'histoire de la stéganographie, les différences entre stéganographie et tatouage ainsi que quelques techniques stéganographi-ques sont décrites dans ce premier chapitre. À la fin de

1. http ://www1.inf.tu-dresden.de/ aw4/publications.html
2. Raj Chandra Bose ; Dwijendra Kumar Ray-Chaudhuri ; Alexis Hocquenghem (1960)

cette première partie, je décris quelques propriétés qui vont nous permettre d'évaluer certaines caractéristiques du processus stéganographique ainsi que les applications stégranographiques les plus fréquentes dans le monde actuel.

Dans un deuxième temps, je rappellerai les notions fondamentales qui nous aideront à comprendre le fonctionnement des codes correcteurs d'erreurs. Cette partie présente des structures algébriques et des notions de base des polynômes.

La troisième partie exposera les codes correcteurs d'erreurs. Cette partie est composée de notions de base des codes correcteurs d'erreurs et plus spécialement de la technique BCH. Cette troisième partie montre pas à pas comment construire un corps de Galois, structure sur laquelle reposent les codes BCH et leur processus de codage. Je donnerai également des exemples de codage et décodage en utilisant cette technique.

Le dernier chapitre décrit la technique stéganographique basée sur le codage par syndrome *fastBCH* utilisé dans [1] et [2]. L'utilisation de tables de correspondance (*look-up tables* en anglais) permet de réduire le temps de calcul, c'est pour cette raison que ce chapitre est également consacré à leur description.

Enfin la dernière partie de ce rapport contient des résultats, des conclusions et une partie d'annexes dédiée à la factorisation de polynômes et la construction de classes cyclotomiques, deux éléments importants dans la création de codes BCH.

Ce dossier permettra donc de découvrir d'une façon globale le fonctionnement des codes correcteurs d'erreurs en mettant l'accent sur les codes *BCH* et leur utilisation dans la stéganographie.

Chapitre 1

La stéganographie

1.1 Brève histoire de la stéganographie

A travers l'histoire, de multiples méthodes ont été utilisées pour cacher l'information. Quelques-uns des plus anciens témoignages à propos de la dissimulation de l'écriture nous ramènent à Hérodote, qui a fait une chronique du conflit entre la Grèce et la Perse au V^e siècle avant Jésus-Christ.
Hérodote nous raconte dans sa chronique que Damarato, un grec exilé de la cité Perse de Susa, connaissait les préparatifs de Xerxès pour attaquer la Grèce et a décidé d'alerter les Spartiates avec un message caché dans des planches de bois. La méthode a été d'enlever la cire des tables en bois, d'écrire le message, puis de les recouvrir avec de la cire.
Dans les 2000 ans qui se sont écoulés depuis Hérodote, différentes formes de messages cachés ont été utilisées de par le monde. Par exemple, dans la Chine ancienne, les messages étaient écris sur de la soie fine, froissée de façon à former une petite boule puis recouverte de cire, avalée par le messager.
Au XV^e siècle, le scientifique italien Giovanni Porta a découvert une technique permettant de masquer des messages dans un œuf cuit grâce à une encre spéciale. Il a mélangé une once d'alun et une pinte de vinaigre et a écrit avec sur la coquille d'œuf. La solution pénètre la coquille poreuse et laisse un message sur la surface de l'albumine de l'œuf cuit, ce qui ne peut être lu que si on épluche l'œuf.
Ces exemples de stéganographie classique montrent que les méthodes pour dissimuler l'information reposait sur la notion de canal secret pour envoyer l'information.
Le stéganographie a pris de l'importance avec l'avènement des réseaux informatiques et des canaux numériques. Le domaine de la cryptographie a beaucoup avancé après la deuxième guerre mondiale, par contre, la stéganographie

moderne n'existe que depuis 1998. Depuis cette année d'importants travaux sont apparus en stéganographie.
Voici une définition de la stéganographie moderne (numérique) :

Définition 1.1.1 La stéganographie est une technique qui permet d'insérer des messages (cryptés ou non cryptés) dans des fichiers apparemment inoffensifs. Ces fichiers sont appelés *médium de couverture* et sont des médias dans lesquels seront dissimulés les informations que l'on souhaite cacher. Il peut s'agir d'un texte, d'une image, d'un son ou d'une vidéo. Dans la stéganographie, il existe une notion de *détectabilité / indétectabilité* ; les données cachées doivent être invisibles par le système visuel humain. L'opération d'insertion ne doit pas détériorer le médium de couverture d'une façon perceptible. Il existe également la notion d'*invisibilité statistique* : un observateur peut comparer les propriétés statistiques de la communication qu'il soupçonne et les comparer avec celles d'une communication ne contenant pas de messages cachés. Des grandes différences peuvent être l'indice d'une communication secrète.

1.2 Schéma stéganographique

Un exemple classique décrivant une communication secrète a été proposé par J Simmons [1] comme le problème des prisonniers. Imaginons une prison, deux prisonniers Alice et Bob et une gardienne Eve. Comme dans toute prison, Alice et Bob sont enfermés dans deux cellules séparées. Alice et Bob sont autorisés à communiquer par messages surveillés, c'est-à-dire que le message doit passer nécessairement par la gardienne Eve, comme montre la figure fig1. Avant d'être enfermés, Alice et Bob ont partagé une *technique secrète* pour partager des informations sans provoquer le moindre signe de soupçon chez Eve. Alice et Bob ont l'idée de s'évader de la prison, grâce à leur *technique secrète*, Alice et Bob pourront communiquer leur plan d'évasion sans que Eve soit suspicieuse. La règle du jeu est donc le suivant : si Eve conclut qu'il y a une communication secrète, elle stoppe la communication ou alors laisse la communication si il n'y a pas de communication secrète.
D'après cette histoire, on peut définir un schéma sténographique, en utilisant les notations suivantes : soit M l'ensemble des messages possibles à insérer et C l'ensemble de tous les supports possibles.
Un schéma stéganographique est défini par deux fonctions :

$$Emb : C \times M \to C$$

1. J. Simmons, *The prisioner's problem and the subliminal chanel*, Plenum Press, 1983

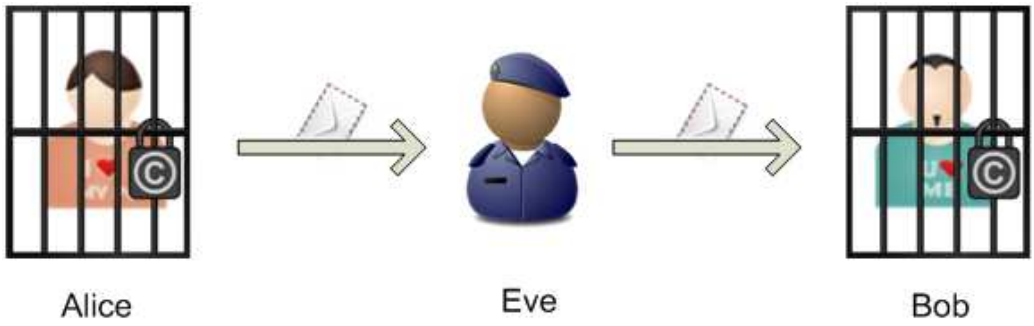


FIGURE 1.1 – Problème classique de communication secrète

qui est la fonction d'insertion (*embedding*), elle prend en paramètre un support ou médium de couverture et un message qu'on veut dissimuler, puis retourne un nouveau médium de couverture appelé *Stego- médium* ou *Stego-objet*. Le stego-médium peut être intercepté par quelqu'un qui peut faire une modification ou non. Finalement, le stego-objet est traité par une fonction d'extraction (fonction inverse à la fonction d'insertion)

$$Ext : C \rightarrow M$$

qui retourne le message original.

1.3 Stéganographie et Tatouage

La stéganographie et le tatouage sont deux techniques d'insertion de données cachées *(data hiding)* mais leurs objectifs sont différents, ainsi que les notions d'attaques.
Pour la science de la stéganographie, le but est de dissimuler un message secret dans un médium de couverture n'ayant rien à voir, de façon à ce qu'un attaquant ne puisse pas savoir si des informations sont dissimulées dans le médium (canal secret).
Pour la science du tatouage ou *watermarking*, le but est différent. Il s'agit ici d'insérer de l'information ayant un rapport direct ou indirect avec le médium de couverture de sorte que le message résiste à des modifications (attaques) du médium (canal fiable).
Finalement, la stéganographie est liée à une communication secrète et le tatouage est lié à une communication fiable.
Le tableau 1.1 donne un résumé des différences entre stéganographie et tatouage.

On cherche à avoir	Stéganographie	Tatouage
Robustesse	Non	Oui
Indétectabilité statistique	Oui	Non
Rapport entre média et message	Aucun	Souvent, il existe un rapport
Communication	Secrète	Fiable
Imperceptibilité	Pas nécessaire	Oui

TABLE 1.1 – Différences entre Stéganographie et Tatouage

1.4 Quelques techniques de stéganographie

1.4.1 Technique LSB

La technique de base, dite LSB pour *Least Significant Bit*, est très simple. Dans le cas d'une image, elle consiste à modifier le bit de poids faible des pixels codant l'image. Une image numérique est une suite de points, que l'on appelle pixel, et dont on code la couleur, le plus souvent, à l'aide d'un triplet d'octets (ex : RGB sur 24 bits). Chaque octet du triplet $\in [0, 255]$ peut être modifié de $+/-1$ sans que la teinte du pixel ne soit visuellement altérée. C'est ce que l'on fait en modifiant le bit de poids faible de l'octet.

Exemple 1.1 Imaginons que les trois pixels suivantes :

$$\{10110101, 11101010, 10010101\},$$
$$\{11101010, 10110101, 00100100\},$$
$$\{10110101, 11010101, 10101010\}$$

On va cacher le caractère $'x'$ représenté par 88 dans le système ASCII. Le caractère $'x'$ a la suite 01011000 comme représentation binaire. Alors, les trois pixels précédents seront modifiés par substitution du LSB

$$\{1011010\mathbf{0}, 1110101\mathbf{1}, 1001010\mathbf{0}\},$$
$$\{1110101\mathbf{1}, 1011010\mathbf{1}, 0010010\mathbf{0}\},$$
$$\{1011010\mathbf{0}, 1101010\mathbf{0}, 10101010\}$$

pour insérer le message 01011000.

1.4.2 Matrix Embedding

La technique de *matrix embedding* est une méthode de codage par syndrome, utilisant la théorie des codes correcteurs, en particulier les codes

linéaires comme le code de Hamming et les codes cycliques comme BCH.
Dans la théorie des codes correcteurs, deux entités souhaitent communiquer à travers un canal bruité. Pour cela, l'émetteur qui désire envoyer un message M transforme celui-ci en un *mot de code* grâce à une matrice génératrice G ou un polynôme générateur $g(x)$.
Le récepteur reçoit un mot résultant d'un mot de code possiblement perturbé (modification de certains bits) pendant la communication, et calcule alors ce que l'on appelle le syndrome du code à l'aide de la *matrice dite de contrôle de parité* H.
La figure 1.2 montre le schéma d'une communication en utilisant des codes correcteur d'erreurs.

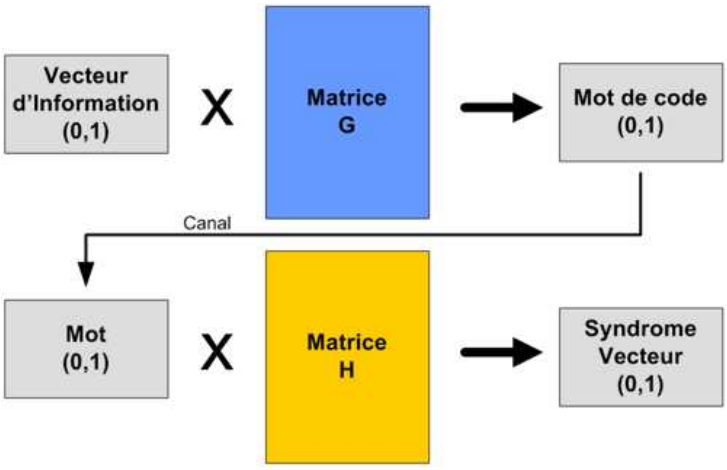


FIGURE 1.2 – Schéma d'une communication en utilisant des codes correcteur d'erreurs

La stéganographie a reprit les notions de codes correcteurs d'erreurs et de syndrome mais l'utilisation a été détournée. Nous étudierons la technique de *Matrix Embedding* en détail dans ce rapport.

1.4.3 Algorithme F5

Dans [5] A. Westfeld parle de divers algorithmes stéganographiques en expliquant leurs insuffisances. De plus, la nouvelle méthode qu'il a développée, en plus d'offrir une résistance aux attaques visuelles et statistiques, a une plus grande capacité stéganographique.
L'un des problèmes mentionnés est que les algorithmes précédents à F5 (comme LSB) intègrent des données en continu. Lorsque nous insérons des données dans les images nous avons une capacité limitée pour cacher le message. Dans de nombreux cas, le message que nous voulons insérer n'occupe

pas toute l'image et une partie de la photo n'est pas utilisée.
Il en résulte des changements qui se concentrent au début de l'image et le reste de l'image reste inaltéré. Au final l'image est vulnérable à une attaque statistique. Pour prévoir des attaques, la fonction d'incrustation devrait utiliser l'image de manière régulière.
Ce que fait F5 pour combattre ce problème, c'est d'étaler les bits du message pseudo - aléatoirement dans l'image.

1.4.4 Technique du papier mouillé

Les endroits où réaliser les modifications (sélection du canal) dans un médium de couverture est un problème évident. Le destinataire ne peut pas déterminer le même canal de choix parce qu'il n'a pas accès au médium de couverture ou à aucune autre information. La technique du papier mouillé (*wet paper* en anglais) propose une solution à ce scénario. Cette méthode permet la construction des arrangements stéganographiques avec des canaux arbitraires. Nous supposons que le canal de choix n'est pas partagé avec le destinataire. Les pixels variables peuvent être modifiés indépendamment entre eux pour communiquer des données secrètes au destinataire, alors que les pixels restants ne sont pas modifiés pendant le codage. Pour décoder les messages secrets, le destinataire ne connaît pas quels pixels ont été utilisées par l'expéditeur pour dissimuler l'information.
Pour expliquer cette méthode, on peut utiliser la métaphore suivante : imaginons que le médium de couverture X est une image qui a été exposée à la pluie et l'expéditeur peut seulement modifier légèrement les taches sèches de X mais pas les taches humides. Pendant la transmission, l'image stéganographié Y se dessèche et le destinataire n'a aucune information sur les pixels secs (le destinataire ne connaît pas les pixels qui ont été employés par l'expéditeur pour dissimuler l'information secrète). Le technique *wet paper* permet aux deux parties de communiquer des messages secrets sous le scénario précédent.

1.5 Propriétés stéganographiques

La stéganographie consiste en l'insertion d'information dans un support hôte sans que celle-ci ne puisse être détectée. La difficulté de la stéganographie est de savoir comment modifier l'image hôte telle que l'image stego soit identique à l'image hôte (voir figure 1.3).

Le stéganalyse ou analyse stéganographique a pour objectif de détecter l'éventuelle présence d'un message caché, si c'est le cas, la communication

FIGURE 1.3 – Objectif de la stéganographie

est stoppée. Il existe des propriétés qui vont nous permettre d'évaluer certaines caractéristiques du processus stéganographique. Ces propriétés sont :

- La capacité d'insertion : c'est le nombre maximal de bits qui peuvent être cachés dans un médium de couverture. Par exemple, si on utilise la technique LSB (*least significant bit*) sur des images à niveaux de gris, on peut cacher 1 bit pour chaque pixel (les bits de poids faible) alors la capacité d'insertion (en bits) est la taille de l'image. Pour une image PGM (*portable grey map*) à niveaux de gris de taille 512 * 512, on peut cacher au maximum 262144 bits.
- La capacité stéganographique : c'est le nombre maximal de bits qui peuvent être modifiés dans un médium de couverture de façon que la probabilité de détection soit insignifiante. La capacité stéganographique est souvent plus petite que la capacité d'insertion. Calculer la capacité stéganographique est une tâche difficile.
- L'efficacité d'insertion : c'est le nombre de bits du message secret insérés par unité de distorsion. Si l'impact de toutes les modifications réalisées pour le processus stéganographique est le même, on peut mesurer l'efficacité d'insertion comme le nombre de bits du message insérés par un changement d'insertion.

1.6 Les applications de la stéganographie

Après la définition de la stéganographie, des questions apparaissent de façon automatique : à quoi peut bien servir la stéganographie ? Pourquoi dissimuler un message si l'on a rien à se reprocher ? Cette science a toujours été utilisée à des fins d'espionnage pourtant, nous allons voir que la stéganographie n'est pas toujours synonyme d'insécurité et peut, au contraire, servir à protéger le droit.

1. Les utilisations malveillantes :
   Internet est une source inépuisable de ressources, la quantité innombrable d'images qui circulent sur le web ainsi que les nombreux fi-

chiers audio qui s'échangent via les logiciels de P2P rendent difficile la détection de messages cachés. La stéganographie peut, en effet, être utilisée pour des usages illicites différents comme par exemple, le camouflage de codes fragmentés à travers du stego-médium et procéder au réassemblage du code malveillant directement sur l'ordinateur de la victime.

La stéganographie a fait récemment la une de la presse en relation avec des réseaux terroristes qui, selon certaines sources, l'auraient utilisée pour communiquer secrètement en cachant des messages dans des photos sur la Web.

La stéganographie intéresse également les pédophiles ainsi que toutes les personnes qui souhaitent cacher des données interdites. Il peut s'agir de dissimuler des données interdites sur des images anodines et des échanges de ce type de données sur internet sans provoquer le moindre signe de soupçon.

Finalement, la stéganographie peut être utilisée pour l'espionnage industriel. En effet, l'utilisation de la stéganographie paraît bien adaptée au vol d'informations confidentielles, car les messages cachés sont difficilement détectables pour les non-avertis.

2. Les utilisations légitimes :
   Il existe quelques pays où il n'existe pas de liberté d'expression. Dans ces pays non démocratiques, la stéganographie apparaît comme un moyen de communiquer plus librement. Dans ce type de pays, l'utilisation de la stéganographie est illicite (à différence des pays démocratiques) mais son usage paraît à nos yeux plus légitime.

   La stéganographie peut aussi être utilisée dans le cas d'une guerre entre deux nations. Dans ce type de situations, les messages ne doivent pas tomber dans des mains ennemies et si la communication est interceptée, le message caché ne doit jamais rester à découvert.

Chapitre 2

Rappels et résultats fondamentaux

2.1 Structures Algébriques

Une structure algébrique est un ensemble d'éléments muni d'une ou plusieurs opérations, par exemple, l'ensemble des entiers $\mathbb{Z}$ muni de l'addition. Dans ce cas, l'opération est dite binaire car elle agit sur deux éléments de $\mathbb{Z}$. Il existe plusieurs structures algébriques, par exemple, l'ensemble $\mathbb{N}$ des entiers naturels, l'ensemble $\mathbb{Q}$ des rationnels, et caetera.

2.1.1 Groupes

Un groupe G [1] est un ensemble d'éléments sur lesquels est définie une opération binaire $\star$ par exemple, pour laquelle on vérifie les propriétés suivantes :

1. L'opération $\star$ doit être associative.
2. G admet un élément identité (existence d'un élément neutre) $e : \forall a \in G, a \star e = e \star a = a$
3. G admet un inverse $\forall a \in G, \exists a^{-1} \in G$ tel que $a \star a^{-1} = a^{-1} \star a = e$.
4. G est *commutatif* ou *abélien* si pour deux éléments a et $b \in G, a \star b = b \star a$

On note souvent un groupe comme un triplet $(G, \star, e)$

1. Introduction à l'algèbre pour les Codes cycliques, A. Bonnecaze, 2006 - 2007. Disponible sur `http://pages-perso.esil.univmed.fr/ bonnecaze//Math/AlgCodeCycl.pdf`

Exemple 2.1 La structure $(\mathbb{Z}_4, +, 0)$ (i. e. la structure $\mathbb{Z}_4$ munie de l'opération addition) peut être définie par sa table d'addition (voir table 2.1).

+	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

TABLE 2.1 – Le groupe additif $(\mathbb{Z}_4, +, 0)$

On a les définitions suivantes :

Définition 2.1.1 Un groupe fini multiplicatif G est dit cyclique s'il admet un élément a tel que pour tout $b \in G$, il existe un entier i tel que $b = a^i$. L'élément a est alors appelé un générateur du groupe cyclique. On note $G = \langle a \rangle$.
Par exemple, le groupe $(\mathbb{Z}_7, +, 0)$ qui a comme élément générateur $a = 3$ est cyclique parce que :

$$3^1 \bmod 7 = 3$$
$$3^3 \bmod 7 = 2$$
$$3^3 \bmod 7 = 6$$
$$3^4 \bmod 7 = 4$$
$$\ldots \text{ et caetera.}$$

Définition 2.1.2 Le nombre d'éléments d'un groupe fini G est appelé l'ordre du groupe et est noté par $|G|$.

2.1.2 Anneaux

Définition 2.1.3 Un anneau A est un ensemble d'éléments défini par deux opérations binaires appelées addition et multiplication et peut être représenté par le triplet $(A, +, *)$. Un anneau A est un groupe *abélien* selon l'opérateur addition. La multiplication doit respecter les contraintes suivantes :

1. La multiplication doit être associative, $a * (b * c) = (a * b) * c$ pour tout a, b, c compris dans A.
2. La multiplication doit être distributive par rapport à l'addition, $a * (b + c) = a * b + a * c$ et $(a + b) * c = a * c + b * c$ pour tout a, b, c compris dans R.

Un anneau est dit commutatif si sa multiplication est commutative, $a*b = b*a$ pour tout a, b compris dans A.

Définition 2.1.4 Un anneau est dit unitaire si l'opération $*$ dispose d'un élément neutre, noté 1 tel que $a * 1 = 1 * a = a$ pour tout a compris dans A.

Exemple 2.2 L'anneau des entiers modulo q avec q premier est :

$$\mathbb{Z}_q = \{0, 1, ..., q-1\}.$$

0 est l'élément neutre additif et 1 l'élément neutre multiplicatif. L'ordre $|\mathbb{Z}_q|$de $\mathbb{Z}_q$ est q puisque $\mathbb{Z}_q$ contient q éléments.

Définition 2.1.5 On se place dans le cas d'un anneau commutatif $(A, +, *)$ et I une partie de A. I est un idéal[2] de A si et seulement si :

1. $(I, +)$ est un sous-groupe de $(A, +)$
2. $I \neq \Phi$
3. $\forall x \in I, \forall a \in A, a * x \in I$

Définition 2.1.6 Dans un anneau A, un idéal I est maximal si et seulement si il y a deux idéaux qui contiennent I : l'anneau A et lui-même (l'idéal I).
Dans un anneau commutatif unitaire, un idéal maximal I est nécessairement premier.
L'idéal I est un idéal maximal de l'anneau A si et seulement si A/I es un *corps*.

2.1.3 Corps

Définition 2.1.7 [3] Considérons l'ensemble des entiers relatifs $\mathbb{Z}$ et un nombre q appartenant à $\mathbb{N}$: l'ensemble des entiers relatifs modulo q est un corps (représenté par F_q) et il contient q éléments. q est appelé *caractéristique* de F_q.

Dans un corps F_q, il est possible de faire des additions, soustractions, multiplications et divisions. Tous les éléments non nuls de F_q sont inversibles : pour un élément a non nul dans F_q, il existe un élément a^{-1} dans F_q tel que $a * a^{-1} = 1$. Un corps possède toutes les propriétés définies aux sections 2.1.2 et 2.1.1.

Exemple 2.3 Le corps formé pour les éléments $0, 1$ est désigné comme F_2. Les tables de addition et multiplication pour ce corps sont :

+	0	1
0	0	1
1	1	0

*	0	1
0	0	0
1	0	1

Un corps fini F contient un nombre fini d'éléments, ce nombre est son ordre et est noté par $|F|$. Les corps finis qui satisfont à la loi de la commutativité sont souvent appelés *Corps de Galois*.

Corps de Galois

Les *Corps de Galois* font partie d'une branche particulière des mathématiques qui modélise les fonctions du monde numérique. Ils sont très utilisés dans la cryptographie ainsi que pour la reconstruction des données comme on le verra dans le chapitre 3.
La dénomination Corps de Galois (*Galois field* en anglais et désigné par GF) provient du mathématicien français E. Galois qui en a découvert leurs propriétés fondamentales.
Un *Corps de Galois* consiste en un ensemble fini de nombres, ces nombres sont constitués à l'aide de l'élément base α comme suit :

$$0, 1, \alpha, \alpha^2, \alpha^3, ...\alpha^{n-2}$$

avec $n \in \mathbb{N}$.
Pour tout élément $\alpha \in GF$, il doit exister des éléments 0 (l'élément neutre additif) et 1 (l'élément neutre multiplicatif) tel que :

$$0 + \alpha = \alpha$$
$$1 * \alpha = \alpha$$
$$(-\alpha) + \alpha = 0$$
$$0 * \alpha = 0$$
$$\text{si } \alpha \neq 0,\ \alpha^{-1} * \alpha = 1$$

Exemple 2.4 On peut construire un corps binaire (ie. les éléments de ce corps seront composés de valeurs 0 ou 1) fini de taille 3. Ce corps fini noté par F_2^3, a comme éléments :

$$F_2^3 = \{000, 001, 010, 100, 011, 101, 110, 111\}$$

L'ordre du corps fini F_2^3 noté par $|F_2^3|$ est 8.

2. http ://www.dma.ens.fr/culturemath/maths/pdf/logique/ideaux.pdf
3. http ://www.math93.com/galois-corpsdegalois.pdf

2.2 Polynômes

Définition 2.2.1 Un polynôme à une inconnue sur un anneau unitaire (cf. définition 2.1.4) est une fonction $f : \mathbb{R} \to \mathbb{R}$ définie par une expression du type :

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0$$

où x est un symbole appelé *indéterminée* du polynôme supposé être distinct de tout élément de l'anneau A et les nombres $a_0, ..., a_n \in A$ sont appelés les coefficients de f.
Si $a_n \neq 0$, n est appelé le degré de f et est noté par $deg(f)$.

Définition 2.2.2 Un polynôme $a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0$ est dit unitaire si et seulement si $a_n = 1$, c'est-à-dire que le coefficient de la variable de degré majeur est 1.

Définition 2.2.3 Un corps de Galois défini comme un ensemble de polynômes à coefficients dans F_q : considérons les polynômes en x dont les coefficients sont dans F_q (cf. définition 2.1.7). Soit $f(x)$ un tel polynôme, il s'écrit :

$$f(x) = \alpha_{n-1} x^{n-1} + \alpha_{n-2} x^{n-2} + \ldots + \alpha_1 x + \alpha_0$$

Les coefficients $\alpha_0, \alpha_1, \ldots, \alpha_{n-1}$ sont des éléments de F_q.
L'ensemble de ces polynômes est noté $F_q[x]$.

Exemple 2.5 Par exemple, les éléments $\alpha^3 = 011$ et $\alpha^5 = 101 \in F_2^3$ sont représentés par les polynômes :

$$f_3(x) = 1x + 1$$
$$f_5(x) = 1x^2 + 1$$

Définition 2.2.4 Soit g(x) un polynôme non nul dans $F_q[x]$, alors pour tout $f(x) \in F_q[x]$, il existe deux polynômes $q(x)$ et $r(x)$ de $F_q[x]$ tels que :

$$f(x) = q(x)g(x) + r(x), \text{ où } deg(r(x)) < deg(q(x)).$$

Soient $f(x)$, $g(x)$ et $d(x)$ des polynômes dans $F_q[x]$, si $d(x)$ divise[4] $f(x)$ et $g(x)$, et si tout polynôme divisant $f(x)$ et $g(x)$ divise aussi $d(x)$, alors $d(x)$ est le plus grand diviseur commun de $f(x)$ et $g(x)$. On note $d(x) = pgcd(f(x), g(x))$. Si $pgcd(f(x), g(x)) = 1$, on dit que $f(x)$ et $g(x)$ sont premiers entre eux.

4. La division euclidienne, ressemble formellement à celle des nombres entiers. Si B et C sont deux polynômes de l'anneau A, il existe un unique couple (Q, R) de polynômes de A tel que : $B = C * Q + R$ avec deg $R <$ deg C :
Source : `http://fr.wikipedia.org/wiki/Division_d'un_polynôme`

Définition 2.2.5 Un polynôme constant est un polynôme constitué d'un unique monôme de degré 0, il s'identifie avec un élément de l'anneau A. On écrit souvent $p(x) = a_0$. Les autres polynômes sont dits non constants.

Définition 2.2.6 Un polynôme non constant (cf. définition 2.2.5) $f(x) \in F_q[x]$ est dit irréductible sur F_q si les seuls polynômes différents de $f(x)$ qui le divisent sont constants. Sinon, le polynôme $f(x)$ est réductible.

Définition 2.2.7 Tout polynôme $f(x) \in F_q[x]$ peut s'écrire :

$$f(x) = f_1(x)^{e_1} + f_2(x)^{e_2} + ... + f_k(x)^{e_k}$$

où les f_i sont des polynômes irréductibles unitaires de $F_q[x]$ et les exposants e_i des entiers positifs. Cette factorisation est unique.
Rappelons qu'un polynôme unitaire a son coefficient de plus haut degré égal à 1.

Définition 2.2.8 Un élément a est une racine (ou un zéro) du polynôme $f(x)$ si $f(a) = 0$.

Chapitre 3

Codes correcteurs d'erreurs

Un des domaines actuels d'application de l'algèbre est la théorie des codes. Dans les années 1940, Richard Hamming, l'un des initiateurs de la théorie des codes, a rapporté l'anecdote suivante : lorsqu'il travaillait pour Bell Company, il avait accès aux ordinateurs uniquement les week-ends. Il avait l'habitude de laisser les ordinateurs exécutant leurs programmes et quand il revenait le week-end suivant, il constatait que certains des programmes les plus importants n'avaient pas été exécutés (lorsque les ordinateurs détectaient une erreur dans un programme, ils s'arrêtaient). Ce retard causait de sérieux problèmes dans son travail et l'a amené à considérer la possibilité d'utiliser certaines techniques de façon à ce qu'il puisse corriger les erreurs.
L'émetteur envoie un message sous la forme d'une chaîne de symboles, mais durant la transmission, il est commun que le message subisse des erreurs et que le récepteur ne reçoive pas le message correctement.

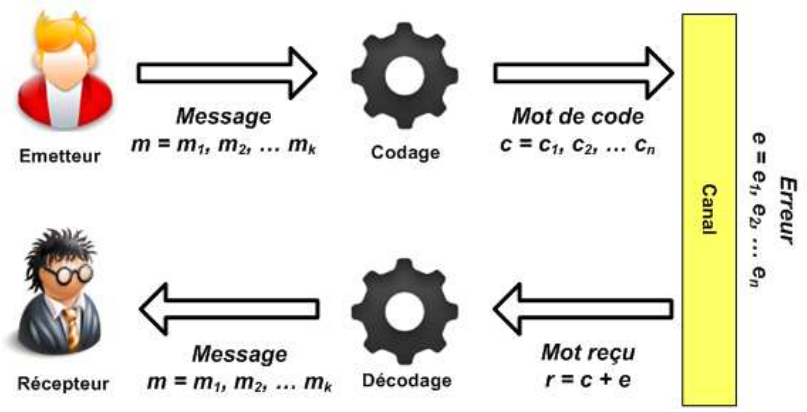


FIGURE 3.1 – Schéma général d'un canal de communication

L'idée basique de la théorie des codes correcteurs d'erreurs est d'envoyer le message que l'on souhaite transmettre avec certaines informations redondantes, c'est-à-dire, étendre la succession de symboles du message à une succession plus grande de façon à permettre au récepteur de détecter et si possible de corriger les éventuelles erreurs de transmission. La figure 3.1 montre le schéma général d'un canal de communication.
Le but des codes correcteurs d'erreurs est la construction de codes qui corrigent la plus grande quantité d'erreurs possible, en minimisant la quantité d'information redondante, en réduisant la probabilité de mauvaise correction et enfin en ayant une complexité de calcul faible.

3.1 Concepts de base

Pour tout code, il faut considérer un alphabet A. On appelle A l'ensemble fini de symboles que l'on utilise pour la construction d'un alphabet. Cet ensemble s'appelle binaire quand l'alphabet a 2 éléments. Par exemple, en informatique, on utilise l'alphabet $A = \{0, 1\}$.
Un code C sur un alphabet A est un sous-ensemble non vide de A^n, avec $n \in \mathbb{N}$. Les éléments de C sont appelés *mots*.
Les mots ont une *longueur* n et sont de la forme $x_1, x_2, ..., x_n$ avec $x_i \in A$.

Définition 3.1.1 Soit m un mot sur le corps binaire représenté par F_2, le poids de Hamming $\omega(m)$ de ce mot m est la quantité de *un* dans m.
Par exemple, si $m = 10010111$, alors $\omega(m) = 5$

Dans la théorie des codes correcteurs d'erreurs, il faut construire des codes qui servent à détecter les erreurs qui pourraient se produire dans la transmission de messages. Il est entendu que, pour être véritablement utile, il faut que les mots de code ne se ressemblent pas trop. Par exemple, si un code contient deux mots qui ne diffèrent que par un symbole, une seule erreur de transmission peut convertir un mot en un autre et passer inaperçu. Pour quantifier le degré de ressemblance entre les mots de code, on utilisera le *distance de Hamming* (cf. définition 3.1.2)

Définition 3.1.2 On se place sur un corps binaire de longueur n. Soient x et y deux mots sur ce corps F_2^n. On appelle *distance de Hamming* d_H entre x et y le nombre de composantes de x et y qui sont différentes et si $x = (x_1, ..., x_n)$ et $y = (y_1, ..., y_n)$, on pose :

$$d_H(x, y) = card(\{i \in 1, ..., n; x_i \neq y_i\})$$

Exemple 3.1 Soient $x = 100\mathbf{0}1010$ et $y = 100\mathbf{1}1010$ deux mots sur F_2^8, la distance de Hamming $d_H(x, y) = 1$

Exemple 3.2 On peut calculer la distance de Hamming dans des corps différents a F_2, par exemple, soient $x = 221\mathbf{1}1\mathbf{2}2$ et $y = 221\mathbf{0}1\mathbf{1}2$ deux mots $\in F_3^7$, la distance de Hamming $d_H(x, y) = 2$

Définition 3.1.3 Soit F_q un corps fini de caractéristique q et longueur n. Si nous définissons les opérations dans F_q^n

$$(x_1, \ldots, x_n) + (y_1, \ldots, y_n) = (x_1 + y_1, \ldots, x_n + y_n)$$

$$a * (x_1, \ldots, x_n) = (a * x_1, \ldots, a * x_n)$$

on obtient un espace vectoriel sur le corps F_q^n. Un code $C \subset F_q^n$ est dit linéaire si C est un sous-espace vectoriel de F_q^n.
Soit k un entier positif, si C a comme dimension k, alors, on dira que C c'est un $[n, k]_q$-code linéaire.
Le nombre d'éléments d'un $[n, k]_q$-code linéaire est q^k avec $k \leqslant n$

Définition 3.1.4 Si C est un code linéaire, et soit $x, y \in C$, on appelle *distance minimale* d la plus petite distance de Hamming entre x et y :

$$d = min\{d(x, y)|x, y \in C; x \neq y\}$$

ou

$$d = min\{\omega(x)|x \in C; x \neq 0\}$$

On dit que C est un $[n, k, d]_q$-code linéaire. Dans ce cas, le code détecte $(d-1)$ erreurs et corrige correctement $\lfloor \frac{d_{min}-1}{2} \rfloor$ erreurs où $\lfloor x \rfloor$ est le plus grand entier inférieur ou égal à x. Plus la distance minimale est grande, meilleure sera la quantité d'erreurs que le code pourra corriger.

Exemple 3.3 Soient a et b les deux mots de F_2^3 avec : $a = 000$ et $b = 111$
La distance minimale entre a et b notée par $d_{min}(a, b)$ est 3. Alors, on pourra corriger :

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor = \left\lfloor \frac{3 - 1}{2} \right\rfloor = 1 \text{ erreur}$$

Définition 3.1.5 Une *matrice génératrice* G d'un $[n, k]_q$-code linéaire C sur le corps F_q^n est une matrice à k lignes et n colonnes dont les lignes représentent les mots de code et sont linéairement indépendants.

Définition 3.1.6 Une *matrice de contrôle de parité* ou simplement *matrice du contrôle* H d'un $[n,k]_q$-code linéaire C sur le corps F_q^n est une matrice à $(n-k)$ lignes et n colonnes qui satisfait :

$$GH^t = HG^t = 0$$

où M^t est la matrice transposée de M.

3.1.1 Le code de Hamming

Soit k le nombre de bits du message à transmettre. Le code de Hamming permet, pour un message de taille k, de faire un code correcteur de longueur n qui détecte et corrige de façon sûre 1 erreur, et en détecte la plupart du temps 2 erreurs. Cette correction est permise grâce à l'ajout de $n-k$ bits de contrôle de parité.
Dans un code de Hamming, les bits de contrôle de parité c_i sont en position 2^i pour $i = 0, 1, 2, \ldots$ et les bits du message d_j occupe le reste du message.
Un code de Hamming $[7,4]_2$ a la structure suivante :

d_3	d_2	d_1	c_2	d_0	c_1	c_0
7	6	5	4	3	2	1

Règle : Le bit de parité de la position 2^k vérifie les bits dans les positions qui contiennent le bit k dans leur représentation binaire. Par exemple, le bit en position 13 est vérifié pour les bits en positions 8, 4 et 1 parce que $13 = \mathbf{1}1\mathbf{0}\mathbf{1}_{(2)}$; $8 = \mathbf{1}000_{(2)}$; $5 = 0\mathbf{1}0\mathbf{1}_{(2)}$; $1 = 000\mathbf{1}_{(2)}$.

Exemple 3.4 On souhaite envoyer le message 0111 :

1	1	1	c_2	0	c_1	c_0
7	6	5	4	3	2	1

Complétons le mot de Hamming correspondant :
c_0 doit produire le bit de parité pair dans les positions 1, 3, 5, 7, c'est-à dire, $(c_0, 0, 1, 1)$, alors c_0 vaut 0 ;
c_1 doit produire le bit de parité pair dans les positions 2, 3, 6, 7, c'est-à dire, $(c_1, 0, 1, 1)$, alors c_1 vaut 0 ;
c_2 doit produire le bit de parité pair dans les positions 4, 5, 6, 7, c'est-à dire, $(c_2, 1, 1, 1)$, alors c_2 vaut 1 ;
Finalement, le mot de Hamming est :

1	1	1	1	0	0	0
7	6	5	4	3	2	1

3.2 Les codes cycliques

Les codes cycliques sont les plus utilisés dans la pratique. Leur mise en oeuvre est simple, la richesse de leurs propriétés algébriques permet d'en faire une étude approfondie, et de bonnes techniques de décodage sont connues pour plusieurs classes de codes cycliques comme les codes Reed-Solomon et BCH. Ce dernier est l'objet de ce stage. On travaille sur un corps fini de caractéristique q et longueur n représenté par F_q^n.

Définition 3.2.1 Un code linéaire C de longueur k sur F_q^n est cyclique si l'ensemble de ses mots $\{c_0, c_1, ..., c_{n-1}\}$ est invariant par décalage circulaire, c'est-à-dire :

$$\{c_0, c_1, ..., c_{n-1}\} \in C \text{ et } \{c_{n-1}, c_0, c_1, ..., c_n\} \in C$$

Exemple 3.5 Le code binaire $C = \{000, 101, 011, 110\}$ est cyclique.
Par contre, la distance minimale $d_{min}(C) = 2$, d'où notre code peut corriger correctement :

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor = \left\lfloor \frac{2 - 1}{2} \right\rfloor = 0 \text{ erreurs}$$

Définition 3.2.2 Tout mot $c = (c_0, c_1, ..., c_{n-1})$ d'un code C sur le corps F_q^n peut être identifié par un polynôme (voir definition 2.2.3)

$$c(x) = c_{n-1}x^{n-1} + \ldots + c_1x + c_0$$

de $F[x]$. Le polynôme $c(x)$ est appelé *représentation polynomiale du mot c*

Exemple 3.6 Soit le mot $c = 10111$ sur F_2^5. Ce mot peut être représenté par le polynôme :

$$c(x) = 1x^4 + 1x^3 + 1x^2 + 1$$

Soit C un code cyclique de longueur n sur F_q, alors on a les définitions suivantes :

Définition 3.2.3 Il existe un seul polynôme unitaire (cf. définition 2.2.7) $g(x)$ en C. Ce polynôme est un générateur de C, c'est-à-dire $C = \langle g(x) \rangle$ et est appelé *polynôme générateur du code C*. Ce polynôme générateur est unique.

Définition 3.2.4 $g(x)$ divise $(x^n - 1)$ dans $F_q[x]$

Définition 3.2.5 Soit r le degré de $g(x)$, tout mot de code $c(x) \in C$ peut être représenté en forme unique par $c(x) = f(x)g(x)$ en $F_q[x]$. La dimension de C est $(k = n - r)$ et n'importe quel message $m = f(x)$ peut être codé par $f(x)g(x)$.

Définition 3.2.6 Si $g(x) = g_0 + g_1x + ... + g_rx^r$ alors

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \dots & g_r & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & \dots & g_{r-1} & g_r & 0 & \dots & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & \dots & 0 & 0 & g_0 & g_1 & \dots & g_{r-1} & g_r \end{pmatrix}$$

est une matrice génératrice de C.

Définition 3.2.7 Soit $g(x)$ le polynôme générateur du code cyclique C. Comme $g(x)$ divise $x^n - 1$ alors

$$h(x) = (x^n - 1)/g(x)$$

est un polynôme de contrôle de parité du code cyclique C.
Si $c(x) = f(x)g(x)$ est un mot code C alors

$$\begin{aligned} c(x)h(x) &= f(x)g(x)h(x) \\ &= f(x)(x^n - 1) \\ &= 0 \end{aligned}$$

On considère la matrice de taille $(n - k) \times (n)$ associée à ce polynôme de contrôle

$$H = \begin{pmatrix} h_k & h_{k-1} & h_{k-2} & \dots & h_0 & 0 & 0 & \dots & 0 \\ 0 & h_k & h_{k-1} & \dots & h_1 & h_0 & 0 & \dots & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & \dots & 0 & 0 & h_k & h_{k-1} & \dots & h_1 & h_0 \end{pmatrix}$$

avec $h(x) = h_0 + h_1x + ... + h_nx^n$.

Exemple 3.7 Soit le code C généré pour le polynôme $g(x) = x^3 + x + 1$ (voir annexe), alors, en considérant la définition 3.2.6, la matrice génératrice pour le code C est :

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

Pour avoir la matrice de contrôle, d'après la définition 3.2.7, considérons le polynôme de vérification défini par :

$$\begin{aligned} h(x) &= (x^n - 1)/g(x) \\ &= (x^7 - 1)/(x^3 + x + 1) \\ &= x^4 + x^2 + x + 1 \end{aligned}$$

Alors, la matrice de contrôle pour C est :

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

3.3 Construction d'un corps de Galois à partir d'un élément primitif

Proposition 3.3.1 Soit F_q^n un corps fini de caractéristique q et longueur n et $p(x)$ sa représentation polynomiale, alors $p(x)$ est un idéal maximal sur F_q^n (cf. définition 2.1.6).

Supposons que $p(x)$ est unitaire et irréductible en F_q^n de degré d et α est une racine de $p(x)$, on peut définir les éléments du corps F_q^n comme expressions polynomiales en α de degré strictement inférieur à d. Noter que $p(x)$ est le seul polynôme irréductible en F_q^n qui a α comme racine et est appelé *polynôme irréductible* de α sur F_q^n
Ce polynôme permet de construire le corps de Galois souhaité. Tous les éléments *non nuls* du corps peuvent être construits en utilisant l'élément α comme racine du polynôme primitif. La table 3.1 [1] montre les polynômes primitifs pour les principaux corps de Galois. Pour chaque valeur de m, il peut y avoir plusieurs polynômes primitifs $p(x)$, mais dans cette table, on mentionne seulement les polynômes ayant le moins d'éléments.

Exemple 3.8 On veut construire un corps binaire F_2 à 8 éléments. Pour représenter vectoriellement les 8 valeur binaires distinctes du corps il faut $log_2(8) = 3$ bits. On peut donc raisonner sur F_2^3.
En consultant la table 3.1 pour $m = 3$, on obtient le mot 1101 dont la représentation polynomiale est $p(x) = x^3+x+1$. Ce polynôme est le polynôme primitif du corps F_2^3.
Soit α une racine du polynôme $p(x) = x^3 + x + 1$. On a alors :

$$p(\alpha) = \alpha^3 + \alpha + 1$$

1. Source : http ://www.univ-tln.fr/ langevin/CDE/primitif.data

m	Représentation binaire de p(x)
2	1 1 1
3	1 1 0 1
4	1 1 0 0 1
5	1 0 1 0 0 1
6	1 1 0 0 0 0 1
7	1 1 0 0 0 0 0 1
8	1 1 1 0 0 0 0 1 1
9	1 0 0 0 1 0 0 0 0 1
10	1 0 0 1 0 0 0 0 0 0 1
11	1 0 1 0 0 0 0 0 0 0 0 1
12	1 1 1 0 0 0 0 0 1 0 0 0 1
13	1 1 1 0 0 1 0 0 0 0 0 0 0 1
14	1 1 1 0 0 0 0 0 0 0 0 0 1 0 1
15	1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
16	1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 1
17	1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
18	1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1
19	1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1
20	1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

TABLE 3.1 – Polynômes primitifs pour les corps F_q^m avec q = 2

Comme $p(\alpha) = 0$, alors :

$$0 = \alpha^3 + \alpha + 1$$
$$\alpha^3 = \alpha + 1$$

Avec α^3, on peut construire tout les autres éléments du corps F_2^3, comme par exemple :

$$\alpha^4 = \alpha * \alpha^3$$
$$\alpha^4 = \alpha * (\alpha + 1)$$
$$\alpha^4 = \alpha^2 + \alpha$$

$$\alpha^5 = \alpha * \alpha^4$$
$$\alpha^5 = \alpha * (\alpha^2 + \alpha)$$
$$\alpha^5 = \alpha^3 + \alpha^2$$
$$\alpha^5 = (\alpha + 1) + \alpha^2$$
$$\alpha^5 = \alpha^2 + \alpha + 1$$

Et ainsi de suite, on construit le corps F_2^3. La table 3.2 montre le corps sur F_2^3 avec $p(x) = x^3 + x + 1$. Soit r le nombre de coefficients d'un

Puissance de α	Élément du GF	Polynôme	α^2	α	1
0	1	1	0	0	1
1	α	α	0	1	0
2	α^2	α^2	1	0	0
3	α^3	$\alpha + 1$	0	1	1
4	α^4	$\alpha^2 + \alpha$	1	1	0
5	α^5	$\alpha^2 + \alpha + 1$	1	1	1
6	α^6	$\alpha^2 + 1$	1	0	1

TABLE 3.2 – Corps sur F_2^3 qui a comme polynôme irréductible $x^3 + x + 1$

polynôme $p(x)$. Tout élément non nul du corps F_2^3 peut être représenté par un polynôme avec $r \leqslant m$.
Les trois premiers éléments α^0, α^1 et α^2 du corps F_2^3 sont appelés *éléments de base*[2].

Note : La création d'un corps de Galois à partir d'un élément primitif ne permet pas de représenter l'élément neutre additif *0* parce que la première valeur du corps est $\alpha^0 = 1$.
Avec cette contrainte, cette méthode permet de générer des corps F_q^m de taille $n = 2^m - 1$.

Dans ce rapport, je propose un algorithme (voir Algorithme 1) qui permet la création d'une table de correspondance pour un corps de Galois $GF(2^m)$ (telle comme est illustrée dans la table 3.2). Cet algorithme est divisé en trois parties. D'abord, la création des éléments de base, après la création de l'élément correspondant à l'élément primitif, et finalement, on génère le reste des éléments du corps F_2^m. Cette table de correspondance sera utilisée très souvent dans ce projet, spécialement dans le chapitre suivant (voir *Look-up tables*).

2. Les éléments de base sont des éléments du corps de Galois qui peuvent être linéairement combinés pour produire tous les autres éléments du corps [13]

Algorithme 1 Construction du $GF(2^m)$

Données :
n : longueur du code
p : Polynôme primitif
s : taille du polynôme primitif
v : vecteur de polynômes du $GF(2^m)$

Début de l'algorithme :
// Pour les éléments de base
for all $i = 0$ to *(s - 1)* **do**
    créer un polynôme p
    i-ème coefficient de $p \leftarrow 1$
    ajouter p à v
end for
// Pour le polynôme primitif
ajouter *pp* à v
// Pour les autres n - s éléments du corps
for all $i = s$ to n **do**
    créer un polynôme *aux*
    $aux \leftarrow i$-ème élément de v
    $aux \leftarrow aux * x^1$
    if *(s-1)*-ème coefficient de $aux = 1$ **then**
        $aux \leftarrow aux + s$-ème élément de v
        (s - 1)-ème coefficient de $aux \leftarrow 0$
    end if
    if degré$(aux) > (s - 2)$ **then**
        créer un polynôme *aux1*
        for all $j = 0$ to *s-1* **do**
            j-ème coefficient de $aux1 \leftarrow j$-ème coefficient de aux
        end for
        ajouter $aux1$ à v
    else
        ajouter aux à v
    end if
end for

Il existe une bibliothèque qui nous permet de créer facilement des corps de Galois de la forme $GF(2^m)$. Elle a été développée en C++ par Arash Partow et est disponible sur `http://www.partow.net/projects/galois/`.
La bibliothèque est divisée en trois classes : corps de Galois, éléments du

corps du Galois et polynômes du corps de Galois. Les opérations telles que l'addition, la soustraction, la multiplication, la division, le module et la puissance peuvent se produire au-dessus des éléments du corps ou au-dessus des polynômes du corps. Le décalage gauche (ie. le produit d'un polynôme par x^n) et le décalage droit (ie. la division d'un polynôme par x^n) peuvent se produire sur des polynômes du corps.
Cette bibliothèque a été utilisée pour la création de l'application de ce stage. Pour générer un corps de Galois, cette bibliothèque utilise un polynôme irréductible (voir table 3.1) .

Exemple 3.9 Le code suivant permettre la création d'un corps de Galois $GF(2^4)$ qu'a comme polynôme irréductible $x^4 + x + 1$.

```
unsigned int polyPrim[6] = {1,1,0,0,1};
galois::GaloisField gf(4,polyPrim);
```

Exemple 3.10 Pour notre code d'exemple, la création des polynômes sur $GF(2^4)$ est faite avec le code suivant :
Pour le polynôme $m_1 = x^4 + x + 1$

```
galois::GaloisFieldElement gf1[5] = {
  galois::GaloisFieldElement(&gf, 1),
  galois::GaloisFieldElement(&gf, 1),
  galois::GaloisFieldElement(&gf, 0),
  galois::GaloisFieldElement(&gf, 0),
  galois::GaloisFieldElement(&gf, 1),
} ;
galois::GaloisFieldPolynomial m1(&gf,4,gf1);
```

Pour le polynôme $m_2 = x^4 + x^3 + x^2 + x + 1$

```
galois::GaloisFieldElement gf2[5] = {
  galois::GaloisFieldElement(&gf, 1),
  galois::GaloisFieldElement(&gf, 1),
  galois::GaloisFieldElement(&gf, 1),
  galois::GaloisFieldElement(&gf, 1),
  galois::GaloisFieldElement(&gf, 1),
};
galois::GaloisFieldPolynomial m2(&gf,4,gf2);
```

Et on peut effectuer des opérations comme :

```
galois::GaloisFieldPolynomial m3(&gf,0);
m3 = m1 + m2; \\somme des polynomes m1 et m2
m3 = m2 % m1; \\reste de la division m2 par m1
```

3.4 Codes BCH

Ils ont été découverts par A. Hocquenghem (pour le cas binaire) en 1959, R. C. Bose et Ray-Chaudhuri en 1960 (d'où le nom de BCH). Ils représentent une famille très importante de codes. D'un point de vue pratique, ils s'encodent et se décodent facilement.
Pour n'importe quel code, il est important de calculer la distance minimale pour connaître sa capacité de correction.

Définition 3.4.1 Soit $\Sigma = \bigcup_i C_i$ l'union des C_i classes cyclotomiques (voir annexe). On appelle Σ l'ensemble de définition de C et les racines primitives de l'unité (voir annexe) $Z(C) = \{\alpha^i | i \in \Sigma\}$ sont appelées *zéros du code ou racines du code C*.

On note le degré de l'ensemble Σ comme $|\Sigma| = deg(g(x))$
De la définition antérieure, on peut conclure qu'un mot de code en représentation polynomiale $c(x) \in C$ si et seulement si $c(\alpha^i) = 0$ pour chaque $i \in \Sigma$ et que la dimension de C est $n - |\Sigma| = n - deg(g(x))$.

Exemple 3.11 Considérons les classes cyclotomiques pour un corps F_2^3 (voir annexe) :

$$C_0 = \{0\}$$
$$C_1 = C_2 = C_4 = \{1, 2, 4\}$$
$$C_3 = C_5 = \{3, 5, 6\}$$

On a $\Sigma = C_0 \bigcup C_1 = \{0, 1, 2, 4\}$ avec $Z(C) = \{\alpha^0, \alpha^1, \alpha^2, \alpha^4\}$ et $|\Sigma| = 4$.

Le théorème suivant présente une borne inférieure pour la distance minimale des codes cycliques.

Théorème 3.4.1 (de la borne BCH) Soit $\delta \in \mathbb{Z}$ et C un code cyclique de longueur n sur F_q^m : si $Z(C)$ contient $(\delta - 1)$ puissances successives ou éléments consécutifs de α, i.e. s'il existe b tel que $\{b, b+1, ..., b+s-1\} \subset Z(C)$, alors $d(C) \geq \delta$.

On souhaite construire des codes cycliques de longueur n avec distance minimale (cf. définition 3.1.4) et dimension aussi grande que possible. Comme la dimension du code C est $n - deg(g(x))$ alors on cherche $deg(g(x))$ le plus petit possible. Si on souhaite que le code ait une distance minimale d'au moins δ (avec δ entier positif), on doit choisir $deg(g(x))$ le plus petit possible et contenant $(\delta - 1)$ éléments consécutifs.

Définition 3.4.2 Un code BCH sur F_q^m de longueur n et distance construite [3] δ est le plus grand code possible ayant comme zéros

$$\alpha^b, \alpha^{b+1}, ..., \alpha^{b+\delta-2}$$

où $\alpha \in F_q^m$ est une racine primitive de l'unité (cf. section A.1) et b est un entier positif.

Il existe deux cas importants :

1. Si $b = 1$, on aura des codes BCH en sens restreint (*narrow-sense BCH codes*, en anglais).
2. Si $n = q^m - 1$, on parle de code BCH primitif.

Définition 3.4.3 Un code BCH avec une capacité de correction de t erreurs sur F_q^m et de distance construite δ est un code cyclique de longueur n engendré par le polynôme :

$$g(x) = ppcm(P_{\alpha^{b+i}} | 0 \leq i \leq \delta - 2)$$

où P_{α^j} est le polynôme minimal de α^j sur F_q^m.

Soit C un code BCH. La distance construite δ est une borne inférieure de la distance minimale d c'est-à-dire, $d(C) > \delta$.
Supposons que l'on désire un code BCH primitif (i.e. n premier avec q) de longueur n qui corrige t erreurs sur F_q^m. Alors, on peut construire un code BCH de paramètres $[n, k, \delta]$ ou $[n, k, 2t + 1]$.

Exemple 3.12 Le tableau 3.3 montre les codes BCH pour $n = 7$ avec les données de l'exemple A.2 de l'annexe : les polynômes irréductibles pour $g(x) = x^7 - 1$ sont :

$$m_1(x) = x^3 + x + 1$$
$$m_1(x)m_3(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$

3. Parfois appelée *distance assignée* ou *design distance* en anglais

t	Σ	$g(x)$	$k = n - \lvert\Sigma\rvert$	$\delta = 2t + 1$	d
1	$\{1, 2, 4\}$	$m_1(x)$	4	3	3
2 à 3	$\{1, 2, 3, 4, 5, 6\}$	$m_1(x)m_3(x)$	1	5 à 7	7

TABLE 3.3 – Codes BCH binaires de longueur 7

Définition 3.4.4 Un code BCH de longueur n et distance construite δ sur F_q^m est définie comme le code dont la matrice de contrôle est :

$$H = \begin{pmatrix} 1 & (\alpha^b) & (\alpha^b)^2 & \dots & (\alpha^b)^{n-1} \\ 1 & (\alpha^{b+1}) & (\alpha^{b+1})^2 & \dots & (\alpha^{b+1})^{n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (\alpha^{b+\delta-1}) & (\alpha^{b+\delta-1})^2 & \dots & (\alpha^{b+\delta-1})^{n-1} \end{pmatrix}$$

Chaque élément de cette matrice appartient à F_q^m et est représenté par un m-tuple d'éléments de F_q^m.

3.4.1 Construction d'un code BCH

On est intéressé par les codes BCH binaires, primitifs et stricts (i.e. $b = 1$ et $n = q^m - 1$). Soient $m \in \mathbb{Z}$, $n = 2^m - 1$ et $\alpha \in F_2^m$ une racine primitive n-ième de l'unité (c'est-a-dire, un élément primitif de l'unité). On va construire le code BCH [15, 7, 5]. Comme $n = 15 = 2^4 - 1$, on construira un code BCH sur F_2^4
D'abord, on cherche les classes cyclotomiques C_i en suivant la procédure décrite dans le annexe. Pour $2^4 - 1$ sur F_2^4, on a les classes cyclotomiques suivantes :

$$\begin{aligned} C_0 &= \{0\} \\ C_1 &= \{1, 2, 4, 8\} \\ C_3 &= \{3, 6, 9, 12\} \\ C_5 &= \{5, 10\} \\ C_7 &= \{7, 11, 13, 14\} \end{aligned}$$

On peut construire les polynômes irréductibles sur la base des classes cyclotomiques, alors :

$$m_0 = x + 1$$

$$m_1 = (x - \alpha)(x - \alpha^2)(x - \alpha^4)(x - \alpha^8)$$
$$m_1 = x^4 + x + 1$$

$$m_3 = (x - \alpha^3)(x - \alpha^6)(x - \alpha^9)(x - \alpha^{12})$$
$$m_3 = x^4 + x^3 + x^2 + x + 1$$

$$m_5 = (x - \alpha^5)(x - \alpha^{10})$$
$$m_5 = x^2 + x + 1$$

$$m_7 = (x - \alpha^7)(x - \alpha^{11})(x - \alpha^{13})(x - \alpha^{14})$$
$$m_7 = x^4 + x^3 + 1$$

Comme $\delta \leq deg(C)$ alors les mots correspondent avec des polynômes qui ont des racines $\alpha, \alpha^2, \alpha^3, \alpha^4$, parce que $\delta = 5$. Notez que $\alpha, \alpha^2, \alpha^4$ appartiennent à la même classe cyclotomique.
Pour un code BCH binaire en sens restreint (i.e. $b = 1$), si $C_k = \{k, 2k, ...2^{\delta-1}k\}$ est une classe cyclotomique, alors :

$$i \in C_k \Rightarrow 2i \in C_k$$

par conséquent :

$$m_i(x) = m_{2i}(x)$$

Comme on a $\delta = 5 = 2t+1$ alors $t = 2$, c'est-à-dire que ce code a la possibilité de corriger 2 erreurs.
On aura besoin des polynômes minimaux des deux premières racines impaires de α (parce que $t = 2$) :

$$\alpha : m_1(x) = x^4 + x + 1$$
$$\alpha^3 : m_3(x) = x^4 + x^3 + x^2 + x + 1$$

D'après la définition 3.4.4, la matrice de contrôle de parité aura la forme :

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

On construit le polynôme générateur

$$\begin{aligned} g(x) &= m_1(x)m_3(x) \\ &= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) \\ &= x^8 + x^7 + x^6 + x^4 + 1 \end{aligned}$$

La dimension de notre code BCH est $n - deg(g(x)) = 15 - 8 = 7$.

La matrice de contrôle H peut s'obtenir si on calcule le corps sur F_2^4 pour le polynôme irréductible de plus petit degré. Pour notre exemple $m_1(x) = x^4 + x + 1$.

Comme $m_1(x) = x^4 + x + 1$ est un facteur irréductible de $x^{15} - 1$, alors on peut construire le corps F_2^4 en suivant l'algorithme 1 decrit dans le section 3.3.
La table 3.4 montre les 15 éléments du corps F_2^4 généré par le polynôme primitif $x^4 + x + 1$.

Puissance de α	Élément du GF	Polynôme	α^3	α^2	α	1
0	1	1	0	0	0	1
1	α	α	0	0	1	0
2	α^2	α^2	0	1	0	0
3	α^3	α^3	1	0	0	0
4	α^4	$\alpha + 1$	0	0	1	1
5	α^5	$\alpha^2 + \alpha$	0	1	1	0
6	α^6	$\alpha^3 + \alpha^2$	1	1	0	0
7	α^7	$\alpha^3 + \alpha + 1$	1	0	1	1
8	α^8	$\alpha^2 + 1$	0	1	0	1
9	α^9	$\alpha^3 + \alpha$	1	0	1	0
10	α^{10}	$\alpha^2 + \alpha + 1$	0	1	1	1
11	α^{11}	$\alpha^3 + \alpha^2 + \alpha$	1	1	1	0
12	α^{12}	$\alpha^3 + \alpha^2 + \alpha + 1$	1	1	1	1
13	α^{13}	$\alpha^3 + \alpha^2 + 1$	1	1	0	1
14	α^{14}	$\alpha^3 + 1$	1	0	0	1

TABLE 3.4 – Corps sur F_2^4 pour $x^4 + x + 1$

La matrice de contrôle pour le code BCH[15,7,5] est (voir définition 3.4.4) :

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} & \alpha^{12} & \alpha^{13} & \alpha^{14} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} & 1 & \alpha^3 & \alpha^6 & \alpha^9 & \alpha^{12} \end{pmatrix}$$

En remplaçant les valeurs de α par les valeurs binaires de la table 3.4 on a :

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

On peur noter que les colonnes 4^e, 6^e, 7^e, 8^e et 15^e sont linéairement indépendantes, alors, on peut en déduire que la distance minimale de ce code est 5.

3.4.2 Algorithme de codage pour un code BCH [n, k, δ]

Dans les codes BCH [n, k, δ] ou BCH [n, k, t] (où $\delta = 2t + 1$), il existe 2 façons de transformer un mot c dans un *mot de code* mc en utilisant le polynôme générateur $g(x)$:

- le codage non systématique : c'est le plus simple et le plus rapide, il faut tout simplement multiplier le mot c (représenté par son polynôme $c(x)$) par le polynôme générateur $g(x)$.

$$mc(x) = c(x) * g(x)$$

- le codage systématique [4] : il est souvent utilisé pour coder l'information qui transite par les réseaux informatiques. L'algorithme 2 montre les pas à suivre pour réaliser ce type de codage.

Exemple 3.13 En utilisant l'algorithme de codage systématique, on va simuler l'envoi du caractère $'h'$ qui a comme représentation binaire 1101000. La représentation polynomiale du caractère $'h'$ est $x^6 + x^5 + x^3$.
Les données du code BCH[15, 7, 5] sont :

$n = 15$, la longueur du code BCH,
$k = 7$, la dimension du code,
$t = 2$

4. Hank Wallace, *Error Detection and Correction Using the BCH Code*, 2001. Disponible sur `www.aqdi.com/bch.pdf`

Algorithme 2 Codage systématique d'un code BCH [n, k, δ]

Données :
$m(x)$ le message à envoyer de taille k ;
$g(x)$ le polynôme générateur du code BCH ;
n la taille du code ;
$n-k$ le nombre de bits de parité ;
La distance minimale de C, $d(C) \geq \delta = 2t+1$;

Début de l'algorithme :
Multiplier $m(x)$ par x^{n-k}.
Diviser $x^{n-k}m(x)$ par $g(x)$.
Avoir le reste $r(x)$ de la division antérieure.
Sommer $r(x)$ et $x^{n-k}m(x)$.
Envoyer la somme d'avant.

$m(x) = x^6 + x^5 + x^3$
$g(x) = x^8 + x^7 + x^6 + x^4 + 1$

1er. pas :
$x^{n-k} * m(x) = (x^6 + x^5 + x^3) * (x^8) = x^{14} + x^{13} + x^{11}$
2ème pas :
$x^{n-k} * m(x)/g(x) = (x^{14} + x^{13} + x^{11})/(x^8 + x^7 + x^6 + x^4 + 1) = x^6 + x^4 + 1$
avec reste $r(x) = x^7 + 1$
3ème pas :
$x^{n-k} * m(x) + r(x) = x^{14} + x^{13} + x^{11} + x^7 + 1$
4ème pas :
Envoyer le mot de code représenté par le polynôme $x^{14} + x^{13} + x^{11} + x^7 + 1$ = 110100010000001.

On peut réaliser une vérification de la façon suivant : si on divise le *mot de code* par le polynôme générateur $g(x)$, le reste de cette division doit être 0.

$$(x^{14} + x^{13} + x^{11} + x^7 + 1)/(x^8 + x^7 + x^6 + x^4 + 1) = x^6 + x^4 + 1$$

avec reste $r(x) = 0$.

3.4.3 Décodage d'un code BCH par syndrome

L'intérêt d'utiliser les codes BCH réside, pour une partie, dans le fait que l'on peut, a priori, choisir la capacité de correction souhaitée (déterminée par δ) et dans une autre, dans l'existence d'un algorithme de décodage efficace.

Soit C le code BCH en sens restreint (i.e. $b = 1$) sur F_q^m (ie. de longueur $q^m - 1$) et distance construite $\delta = 2t + 1$ (i.e. on peut corriger au maximum t erreurs). Soit α une racine primitive n-ième de l'unité. On va décoder le code déterminé par les racines $\alpha, \alpha^2, ..., \alpha^{\delta-1}$ avec :

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

On suppose qu'on a envoyé un mot de code $c(x) \in C$ et le récepteur a reçu un vecteur r représenté par son polynôme $r(x) = c(x) + e(x)$ avec $\omega(e(x)) = d \leq t$. Soient $0 \leq i_1 < ... < i_d \leq (n-1)$ les positions où l'erreur est apparu et $e_{i_1}, e_{i_2}, ..., e_{i_d}$ les positions des erreurs $e(x)$. On peut exprimer le $e(x)$ par le polynôme :

$$e(x) = e_{i_1} x^{i_1} + e_{i_2} x^{i_2} + ... + e_{i_d} x^{i_d}$$

Définition 3.4.5 Soient r le vecteur reçu, e le vecteur d'erreur, H la matrice de contrôle de parité du code C : le syndrome du vecteur reçu est défini par :

$$S(r) = S(e) = rH^t = (s_1, s_2, ..., s_{2t})$$

où H^t est la transposée de la matrice de contrôle.
On peut utiliser une notation polynomiale du syndrome :

$$S(x) = s_1 + s_2 x + ... + s_{2t} x^{2t-1}$$

Si $j = 1, 2, ..., 2t$, on définit des syndromes comme :

$$\begin{aligned} s_j &= r(\alpha^{j+1}) \\ &= c(\alpha^{j+1}) + e(\alpha^{j+1}) \\ &= e(\alpha^{j+1}) \end{aligned}$$

Comme $e(x) = e_{i_1} x^{i_1} + e_{i_2} x^{i_2} + ... + e_{i_d} x^{i_d}$ et α est une racine primitive n-ième de l'unité, on peut représenter le syndrome de la façon suivante :

$$s_j = \sum_{k=1}^{d} e_{i_k} (\alpha^{i_k})^{j+1}$$

Si le polynôme erreur $e(x)$ est nul, alors les syndromes sont tous *zéro*. Pour simplifier la notation, on considère les égalités suivantes : $\eta_l = e_{i_l}$, $\beta_l =$

α^{i_l} pour $l = 1, 2, ...d$ où i_l est la position du i-èsime erreur, alors on a les syndromes :

$$\begin{aligned} s_1 &= \eta_1\beta_1 + \eta_2\beta_2 + ... + \eta_d\beta_d \\ s_2 &= \eta_1\beta_1^3 + \eta_2\beta_2^3 + ... + \eta_d\beta_d^3 \\ &\vdots \\ s_{2t} &= \eta_1\beta_1^{2t-1} + \eta_2\beta_2^{2t-1} + ... + \eta_d\beta_d^{2t-1} \end{aligned}$$

Les η_l sont les valeurs de l'erreur (en relation au vecteur reçu $r(x)$) et les β^l sont les positions où les erreurs se trouvent. A ce moment là, seuls les premiers $2t$ symboles du syndrome sont connus, β^l et le nombre d'erreurs d sont inconnues.

Exemple 3.14 Dans l'exemple 3.13, le mot de code envoyé était 1101000100000001.
Imaginons que d'après la transmission, le mot de code ait été modifié dans les positions représentées par le polynôme $x^{11} + x^5$
On va calculer le syndrome avec les données suivantes :

$$\begin{aligned} c(x) &= x^{14} + x^{13} + x^{11} + x^7 + 1 \\ e(x) &= x^{11} + x^5 \\ r(x) &= x^{14} + x^{13} + x^7 + x^5 + 1 \end{aligned}$$

Une façon de calculer les syndromes en utilisant moins de calculs est : [5]

$$s_i = r(x) \bmod m_i(x)$$

où $m_i(x)$ est le polynôme minimal pour α^i, c'est-à-dire les polynômes irréductibles trouvés dans la sous-section 3.4.1.
Dans notre cas, les $2t$ syndromes sont :

$$\begin{aligned} s_1 &= r(x) \bmod m_1(x) \\ s_2 &= r(x) \bmod m_2(x) \\ s_3 &= r(x) \bmod m_3(x) \\ s_4 &= r(x) \bmod m_4(x) \end{aligned}$$

5. Hank Wallace, *Error Detection and Correction Using the BCH Code*, 2001. Source : www.aqdi.com/bch.pdf

En remplaçant les valeurs de $r(x)$ et $m_i(x)$ pour chaque syndrome S_i :

$$\begin{aligned}
s_1 &= x^{14} + x^{13} + x^7 + x^5 + 1 \bmod x^4 + x + 1 \\
s_2 &= x^{14} + x^{13} + x^7 + x^5 + 1 \bmod x^4 + x + 1 \\
s_3 &= x^{14} + x^{13} + x^7 + x^5 + 1 \bmod x^4 + x^3 + x^2 + x + 1 \\
s_4 &= x^{14} + x^{13} + x^7 + x^5 + 1 \bmod x^4 + x + 1
\end{aligned}$$

Les valeurs de s_i sont :

$$\begin{aligned}
s_1 &= x^3 \\
s_2 &= x^3 \\
s_3 &= x + 1 \\
s_4 &= x^3
\end{aligned}$$

On transforme les syndromes exprimées par x en équations exprimés par α :

$$s_i(x) = (s_i(\alpha))^i$$

Dans notre exemple, on a :

$$\begin{aligned}
s_1 &= \alpha^3 \\
s_2 &= (\alpha^3)^2 \\
s_3 &= (\alpha^1)^3 + (\alpha^0)^3 \\
s_4 &= (\alpha^3)^4
\end{aligned}$$

Finalement, les syndromes sont :

$$\begin{aligned}
s_1 &= \alpha^3 \\
s_2 &= \alpha^6 \\
s_3 &= \alpha^3 + 1 = \alpha^{14} \\
s_4 &= \alpha^{12}
\end{aligned}$$

Les syndromes peuvent être représentés par le polynôme

$$S(x) = \alpha^{12}x^3 + \alpha^{14}x^2 + \alpha^6 x + \alpha^3$$

Définition 3.4.6 On considère le polynôme suivant en $F_q^m[x]$

$$\sigma(x) = \sigma_d x^d + \sigma_{d-1}x^{d-1} + \ldots + \sigma_1 x^1$$

appelé *polynôme localisateur d'erreurs* et est défini comme le polynôme qui a comme racines les inverses x_l^{-1} des positions d'erreur pour $l = 1, 2, ..., d$, c'est-à-dire

$$\sigma(x) = (\beta_1 + x)(\beta_2 + x)\ldots(\beta_n + x)$$

Si on connaît les coefficients de $\sigma(x)$, on pourrait calculer ses racines et réussir à avoir les localisations des erreurs. Il existe une relation entre les coefficients de $\sigma(x)$ et les syndromes connus. Cette relation peut être exprimée de façon matricielle :

$$\begin{bmatrix} s_1 & s_2 & \dots & s_{v-1} & s_v \\ s_2 & s_3 & \dots & s_v & s_{v+1} \\ s_3 & s_4 & \dots & s_{v+1} & s_{2v+2} \\ \vdots & & & & \\ s_v & s_{v+1} & \dots & s_{2v-2} & s_{2v-1} \end{bmatrix} \begin{bmatrix} \sigma_v \\ \sigma_{v-1} \\ \sigma_{v-2} \\ \vdots \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} -s_{v+1} \\ -s_{v+2} \\ -s_{v+3} \\ \vdots \\ -s_{2v} \end{bmatrix} \tag{3.1}$$

Le problème de trouver les coefficients du polynôme localisateur d'erreurs se réduit à résoudre ce système de d équations linéaires dans des d variables inconnues $\sigma_1, \sigma_2, \dots, \sigma_d$. Un autre problème est aussi la valeur de d qui est inconnue.
Si le code reçu $r(x)$ n'est pas affecté par des erreurs alors tous les coefficients du syndrome seront nuls $r(x) = c(x)$.
Il existe de nombreuses méthodes [6] pour résoudre le système d'équations 3.1 :

- l'algorithme d'Euclide, est un algorithme récursif qui permet de trouver le plus grand diviseur commun de deux polynômes dans un *corps de Galois* $GF(q^m)$;
- l'algorithme de Berlekamp - Massey qui, de manière générale, permet de résoudre les identités de Newton de façon itérative ;
- l'algorithme de Peterson - Gorenstein - Zierler qui permet de trouver les coefficients du polynôme localisateur d'erreurs à l'aide du calcul de la déterminante et l'inverse d'une matrice de Syndromes.

Ces algorithmes ne seront pas traités dans ce rapport. Leur étude et leur implémentation seront laissés *a posteriori* et comme projet personnel.
Les articles [1], [2] présentent une méthode différente pour calculer les coefficients du polynôme localisateur d'erreurs et sera décrite dans le prochain chapitre.

6. Rocío Meza Moreno, *El algoritmo de Berlekamp-Massey y decodificación de códigos BCH sobre el anillo* Z_p^s, México 2007. Source : http ://tesiuami.izt.uam.mx/uam/aspuam/presentatesis.phprecno=13706&docs=UAMI13706.pdf

Chapitre 4

Codage par syndrome fastBCH pour la stéganographie

L'article [1] présente une technique stéganograhique pour cacher des données basées sur le codage de syndrome de $BCH[n, k, t]$, où n est la longueur du mot de code, k est la dimension code et t est le nombre maximal d'erreurs que le code BCH pourra corriger. La technique proposée permet d'insérer des données en modifiant quelques coefficients dans un bloc afin que le syndrome représente le message (les bits insérés). La figure 4.1 montre un schéma de décodage par syndrome à l'aide de la matrice de contrôle de parité.

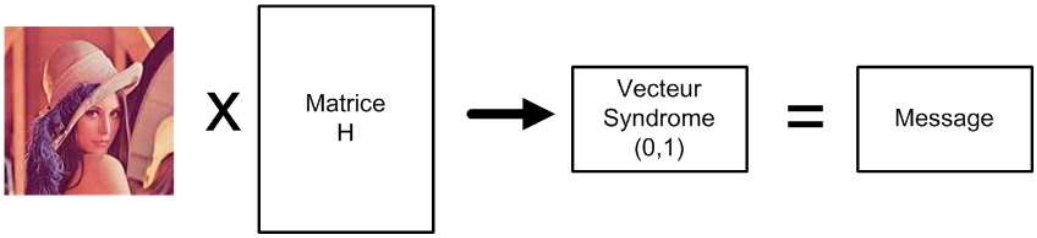


FIGURE 4.1 – Décodage par syndrome

Soit $r(x)$ la représentation polynomiale du mot reçu, $v(x)$ la représentation polynomiale du vecteur de couverture, $e(x)$ la représentation polynomiale du vecteur erreur et H^t est la transposé de la matrice de contrôle de parité H. Alors :

$$r(x) = v(x) + e(x)$$

L'insertion d'un message m sur le médium de couverture représenté par $v(x)$ produit le polynôme $r(x)$. A l'extraction, le message m est obtenue en calcu-

lant $r * H^t$ c'est-à-dire, en calculant le syndrome :

$$\begin{aligned} m &= r * H^t \\ r &= c + e \\ m &= (v + e)H^t \\ m - vH^t &= eH^t \end{aligned} \tag{4.1}$$

et l'on note :

$$s = m - vH^t \tag{4.2}$$

D'après les deux dernières équations, on a :

$$s = e * H^t \tag{4.3}$$

Pour la stéganographie, le but est de trouver un nombre minimal de coefficients *un* dans $e(x)$ afin de diminuer la dégradation du support. La solution à l'équation 4.3 montre les positions appropriées des éléments dans le vecteur $c(x)$ à modifier afin de cacher le message m pour diriger $c(x)$.
Le message caché peut être récupéré du vecteur $r(x) = c(x) + e(x)$ en utilisant l'égalité $m = r * H^t$.
L'article [1] assume que le polynôme $e(x)$ a ν *flips*[1] dans les positions $(j_1, j_2, \ldots, j_\nu)$ où les j_i sont les indices des coefficients à être modifiés pour camoufler le message m avec $0 \leqslant j_1 < j_2 < \ldots < j_\nu < n$.
Le polynôme $e(x)$ est alors :

$$e(x) = x^{j_1} + x^{j_2} + \ldots + x^{j_\nu}$$

D'après l'exemple 3.14 et [1] le syndrome peut être exprimé comme suit :

$$\begin{aligned} s_1 &= \alpha^{j_1} + \alpha^{j_2} + \ldots + \alpha^{j_\nu} \\ s_2 &= (\alpha^{j_1})^3 + (\alpha^{j_2})^3 + \ldots + (\alpha^{j_\nu})^3 \\ &\vdots \\ s_t &= (\alpha^{j_1})^{2t-1} + (\alpha^{j_2})^{2t-1} + \ldots + (\alpha^{j_\nu})^{2t-1} \end{aligned} \tag{4.4}$$

Avec, $\alpha^{j_1}, \alpha^{j_2} + \ldots + \alpha^{j_\nu}$ inconnues.
Les valeurs $(j_1, j_2, \ldots, j_\nu)$ sont les indices des coefficients qui seront modifiés dans $c(x)$. Trouver ces valeurs donne une solution à notre problème stéganographique.
L'équation 4.4 a $2k$ solutions possibles.

1. *flip* : mot anglais pour designer le basculement d'un bit d'information

[1] fait la transformation suivante :

$$\beta_l = \alpha^{j_l}$$

où $1 \leqslant l \leqslant \nu$. Après la transformation précédente, l'équation 4.4 est :

$$\begin{aligned} s_1 &= \beta_1 + \beta_2 + \ldots + \beta_\nu \\ s_2 &= (\beta_1)^3 + (\beta_2)^3 + \ldots + (\beta_\nu)^3 \\ &\vdots \\ s_t &= (\beta_1)^{2t-1} + (\beta_1)^{2t-1} + \ldots + (\beta_\nu)^{2t-1} \end{aligned} \tag{4.5}$$

D'après la définition 3.4.6 et [1], on a un polynôme appelé *polynôme localisateur d'erreurs* représentée par :

$$\sigma(x) = (\beta_1 + x)(\beta_2 + x)\ldots(\beta_\nu + x)$$

Les solutions (ie. racines) de $\sigma(x)$ sont $\beta = \beta_1, \beta_2, \ldots, \beta_\nu$.
La relation entre $\sigma(x)$ et β_l est représenté par :

$$\begin{aligned} \sigma_1 &= \beta_1 + \beta_2 + \ldots + \beta_\nu \\ \sigma_2 &= \beta_1\beta_2 + \beta_2\beta_3 + \ldots + \beta_{\nu-1}\beta_\nu \\ &\vdots \\ \sigma_\nu &= \beta_1\beta_2\ldots\beta_\nu \end{aligned} \tag{4.6}$$

A partir des racines $\sigma_1, \sigma_2, \ldots, \sigma_\nu$ du polynôme $\sigma(x)$ on obtient les valeurs $\beta_1, \beta_2, \ldots, \beta_\nu$, ce que nous permet déduire les valeurs $j_1, j_2, \ldots, j_\nu$ du vecteur $e(x)$. Les valeurs $j_1, j_2, \ldots, j_\nu$ sont les positions de bascule (*flip*) binaire dans le vecteur de couverture $v(c)$.
L'article [1] utilise un schéma BCH avec $t = 2$. L'algorithme proposé utilise des tables de correspondance (*look-up tables* en anglais) pour calculer les racines quadratiques et cubiques de $\sigma(x)$.

4.1 Look-up tables

Pour décoder un mot codé par BCH, après le calcul du syndrome et des coefficients du polynôme localisateur d'erreurs $\sigma(x)$ (de degré égal au nombre d'erreurs), le système doit chercher les racines du polynôme localisateur d'erreurs et chaque racine nous donne la position de l'erreur.
Un méthode pour calculer ces racines de degré 2 et degré 3 du polynôme localisateur d'erreurs a été décrite dans [13]. Le calcul de syndromes et le calcul des racines du polynôme localisateur d'erreurs sont des taches qui

consomment une grande quantité de temps. Grâce aux *look-up tables*, on peut réduire le temps de calcul des racines du polynôme correcteur d'erreurs $\sigma(x)$.
Une table de correspondance est une structure de données employée pour remplacer un calcul par une opération de consultation qui, souvent est plus simple. Le gain de vitesse peut être significatif, car rechercher une valeur en mémoire est souvent plus rapide qu'effectuer un calcul important.

4.1.1 Calcul des racines du polynôme de degré 2

Si on a un mot de code a deux erreurs, le polynôme localisateur d'erreurs est de la forme :

$$f(x) = x^2 + \sigma_1 x + \sigma_2 \tag{4.7}$$

Pour trouver les racines du polynôme$f(x)$, on doit résoudre l'équation :

$$x^2 + \sigma_1 x + \sigma_2 = 0$$

Si y_0 est une racine du polynôme

$$g(y) = y^2 + y + \frac{\sigma_2}{\sigma_1}$$

alors $\sigma_1 y_0$ et $\sigma_1 y_0 + \sigma_1$ sont racines de $f(x)$.
Soit $c = \dfrac{\sigma_2}{\sigma_1}$ avec $\sigma_1 \neq 0$ La table de correspondance doit stocker les racines de

$$y^2 + y + c = 0 \tag{4.8}$$

avec $c \in [0, 2^m - 1]$.
La méthode décrite par [13] propose une solution à l'équation 4.8. Cette solution est une combinaison linéaire des $2m-1$ *solutions de base*. Une solution de base est une solution à l'équation 4.8 avec c comme un des éléments base $c_1, c_2, \ldots c_{2m-1}$ du corps de Galois.
Les éléments de base sont des éléments du corps de Galois qui peuvent être linéairement combinés pour produire tous les autres éléments du corps.
La solution à l'équation 4.8 est une combinaison linéaire des solutions de base qui correspondent à des éléments de base dont

$$y^2 + y + c = 0$$

L'algorithme 3 montre le pseudo-code de la technique décrite par [13].

Algorithme 3 Calcul des racines de degré 2

Données :
b : nombre d'éléments de base et nombre de solutions de base ;
v_i : table d'éléments du corps F_2^m, $i = \{0, 1, \ldots n-1\}$;
e_i : éléments de base, $i = \{0, 1, \ldots b\}$;
s_i : solutions de base, $i = \{0, 1, \ldots b\}$;
y_i : racines de degré 2, $i = \{0, 1\}$;

Début de l'algorithme :
// calculer b
$b \leftarrow 2m - 1$
// calculer les b éléments de base
repeat
    $s \leftarrow (v_j)^2 - v_j$
    if $s = e_i$ **then**
        garder s
        $i \leftarrow i + 1$
    end if
    $j \leftarrow j + 1$
until avoir tous les éléments de base e_i
// construire le *look-up table*
for all chaque élément v_i du corps F_2^m **do**
    $y_0 \leftarrow v_i \mapsto s_i$ // representation de v_i par des solutions de base
    $y_1 \leftarrow y_0 + v_0$
end for

Exemple 4.1 Pour un corps F_2^4 qui a x^4+x+1 comme polynôme générateur, on aura 3 (parce que $2m-1 = 2*2-1 = 3$) éléments de base qui sont α^0, α^1 et α^2.
Maintenant, on cherche les solutions de base de la façon suivante : il faut trouver des valeurs de α qui satisfait l'équation

$$\alpha^2 + \alpha = e_i$$

où e_i sont des éléments base du corps F_2^4 et $i = 1, 2, \ldots 2m-1$
Les solutions de base pour les éléments de base α^0, α^1 et α^2 du corps F_2^4 sont $\alpha^5, \alpha^9, \alpha^3$, en effet :

$$\begin{aligned} (\alpha^5)^2 + \alpha^5 &= \alpha^2 + \alpha + 1 + \alpha^2 + \alpha = \alpha^0 \\ (\alpha^9)^2 + \alpha^9 &= \alpha^3 + \alpha^3 + \alpha^1 = \alpha^1 \\ (\alpha^3)^2 + \alpha^3 &= \alpha^3 + \alpha^2 + \alpha^3 = \alpha^2 \end{aligned} \tag{4.9}$$

On se donne un exemple quelconque, supposons que $c = \alpha^8$ alors on cherche sa représentation polynomiale dans la table 3.4 :

$$\alpha^8 = \alpha^2 + 1 = \alpha^2 + \alpha^0$$

et on cherche les solutions de base pour α^2 et α^0 dans 4.9

$$\begin{aligned} y_0 &= \alpha^3 + \alpha^5 \\ y_0 &= \alpha^3 + (\alpha^2 + \alpha) \\ y_0 &= \alpha^{11} \end{aligned}$$

y_0 est la première racine de degré deux. La deuxième racine peut être trouvée en remplaçant la valeur de y_0 dans :

$$y_1 = y_0 + \alpha^0 \tag{4.10}$$

La preuve de l'équation 4.10 est dans [1].
pour notre exemple

$$\begin{aligned} y_1 &= y_0 + \alpha^0 \\ y_1 &= \alpha^{11} + \alpha^0 \\ y_1 &= (\alpha^3 + \alpha^2 + \alpha) + \alpha^0 \\ y_1 &= \alpha^{12} \end{aligned} \tag{4.11}$$

4.1.2 Calcul des racines du polynôme de degré 3

Pour les polynômes de degré trois, l'utilisation d'une table de correspondance implique la utilisation d'un polynôme général de degré trois de la forme :

$$f(x) = x^3 + \sigma_1 x^2 + \sigma_2 x + \sigma_3 \tag{4.12}$$

Le polynôme 4.12 peut aussi être représenté de la façon suivante :

$$f(y) = y^3 + ay + b \tag{4.13}$$

où

$$a = \sigma_1^2 + \sigma_2 \quad \text{et} \quad b = \sigma_1 * \sigma_2 + \sigma_3 \tag{4.14}$$

Si on utilise une table de correspondance, la table doit contenir un ensemble de solutions pour chaque valeur possible de a et b.
Pour résoudre cette équation, [13] propose l'astuce suivante :

$$y = x + \frac{a}{x} \tag{4.15}$$

En remplaçant cette dernière équation dans 4.13 on aura :

$$(z^3)^2 + bz^3 + a^3 = 0 \tag{4.16}$$

Qui est une équation quadratique avec z^3 comme variable. Cette dernière équation peut être résolue par la procédure décrite dans la section 4.1.1
Une fois trouvée une racine $y_0 = \alpha^{3k}$, avec $k \in \mathbb{N}^*$, une racine quadratique de l'équation 4.16, il faut vérifier que $y_0 = \alpha^{3k} \equiv z_0 = \alpha^k$. [13] propose une méthode simple pour faire cette vérification. Finalement, on aura z_0 de la forme α^k.
Les trois autres deux valeurs de z peuvent être calculées par :

$$z_1 = z_0 * \alpha^{\frac{2^{2m}-1}{3}}$$
$$z_2 = z_1 + z_0$$

Ces 3 valeurs de $Z = \{z_1, z_2, z_3\}$ sont remplacées dans 4.15 pour avoir les trois racines $y = \{y_0, y_1, y_2\}$ qui sont des racines de l'équation 4.13.
Cette dernière méthode qui est difficile à implémenter peut être remplacé par l'algorithme de Chien.

Algorithme de Chien

L'algorithme de Chien est un algorithme récursif qui est souvent utilisé pour déterminer les racines des polynômes définis sur un corps fini.
Cet algorithme est du type *brute force*, c'est-à-dire, qu'il évalue toutes les possibilités. Par exemple pour un code BCH[15,7,2], on évalue l'équation 4.13 pour tous les éléments du corps F_2^4, sauf pour l'élément nul.
La sortie de cet algorithme est une séquence d'éléments du corps de Galois stokes dans une matrice de m colonnes par n files où m est le nombre de racines calcules et n et la taille du corps de Galois. Lorsque les symboles sont nuls, ceux-ci nous indiqueront qu'une racine n'a pas été détectée. Cette méthode est recommandée pour des applications de théorie des codes dans les quelles la taille du code n est petit.

Exemple 4.2 On se place dans le corps F_2^4. On va trouver les racines quadratiques pour $c = \alpha^8$ en utilisant le méthode de Chien : Il faut évaluer la équation 4.8 avec tous les éléments du corps F_2^4 :

$$(\alpha^0)^2 + \alpha^0 + \alpha^8 = 0$$
$$\alpha^0 + \alpha^0 + \alpha^8 = 0$$
$$\alpha^8 \neq 0$$

$$(\alpha^1)^2 + \alpha^1 + \alpha^8 = 0$$
$$\alpha^2 + \alpha^1 + \alpha^8 = 0$$
$$\alpha^4 \neq 0$$

ainsi de suit...

$$(\alpha^{11})^2 + \alpha^{11} + \alpha^8 = 0$$
$$\alpha^7 + \alpha^{11} + \alpha^8 = 0$$
$$0 = 0$$

$$(\alpha^{12})^2 + \alpha^{12} + \alpha^8 = 0$$
$$\alpha^9 + \alpha^{12} + \alpha^8 = 0$$
$$0 = 0$$

On peut noter que α^{11} et α^{12} sont les deux racines que on avait trouvé en utilisant la méthode proposé par [13].

Pour ce stage, les tables de correspondance sont remplis en utilisant : la méthode proposé par [13] pour le calcul des racines de degré 2 et le méthode de Chien pour le calcul des racines de degré 3.

4.2 Dissimulation des données en utilisant le codage par syndrome

L'algorithme proposé par *R. Zhang, V. Sachnev, H. Joong Kim*, dans [1] est basé sur le codage par syndrome $BCH[n, k, t]$, où $n = 2^m - 1$ est la longueur du code. La capacité de camouflage pour chaque bloc de données est $M_b = m * t$.
La méthode proposée dans [1] utilise $t = 2$ (capacité de correction du code BCH).
Comme $t = 2$ alors $\omega\,^2(e(x)) = 3$ c'est-à-dire, le nombre maximal de camouflages qu'on peut réaliser par syndrome est 3.
Soient :

- le vecteur m de longueur k qui représenté le message secret.
- le vecteur de couverture v de taille n
- q_{2x15} la table des racines de degré 2 calculée en avance.
- c_{3x15} la table des racines de degré 3 calculée en avance.

La technique consiste à diviser le message m en deux parties m_1 et m_2 et de calculer les syndromes s_1 et s_2 pour chaque message avec les équations

2. Poids de Hamming. Voir définition 3.1.1

suivantes.

$$s_1 = m_1^t – vH^t$$
$$s_2 = m_2^t – vH^t$$

où H^t est la transposé de la matrice de control H.
La quantité de bits qu'on pourra *flipper* dépendra des valeurs des syndromes trouvés.
Par exemple, si $s_1^3 + s_2 = 0$, alors on peut basculer un bit dans v. Si c'est le cas, le syndrome s_1 est un polynôme unitaire (voir définition 2.2.2) ce qui nous donne la position à basculer $j = log(s_1)$.
Tous ces calculs sont faites sous la représentation polynomiale des vecteurs.
L'algorithme 4 montre le pseudo-code de cette technique.

4.3 Considérations pour l'implémentation

L'implémentation de l'algorithme 4 a été codée en C++ et une version *beta* est disponible sur `http://hugo.alatristasalas.free.fr` sur la license *GNU General Public License*.
Les opérations algébriques sur le corps de Galois $GF(q^m)$ sont réalisées grâce à la bibliothèque *the Galois Field Arithmetic Library* qui a été developpée par *Arash Partow* et est disponible sur *http ://www.partow.net/projects/galois/index.html*
Cette bibliothèque utilise comme générateur du corps de Galois un polynôme primitif qui doit être fourni avant la création du GF.

NOTE : La plupart des implémentations des codes BCH utilisent un polynôme primitif fourni à la main parce que la création de ce polynôme est compliquée.

Par exemple, pour créer le corps de Galois $GF(2^4)$ de polynôme irréductible $x^4 + x + 1$ il faut ajouter les lignes de code suivantes :

```
unsigned int polyPrim[6] = {1,1,0,0,1};\\
galois::GaloisField gf(4,polyPrim);
```

Ces deux lignes de code nous permettent de créer un corps de Galois appelé gf du degré 4.

Algorithme 4 Data hiding using Syndrome coding

Données :

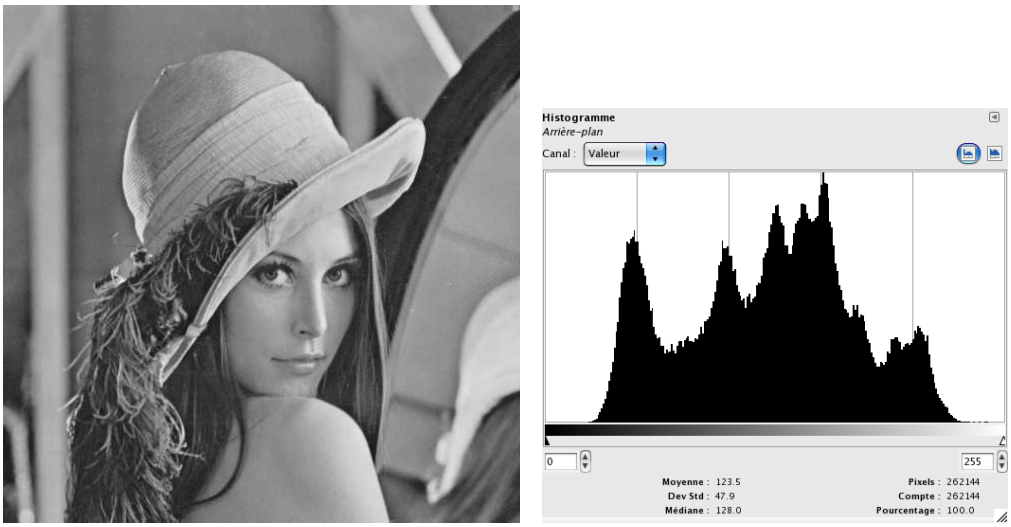
m :vecteur du message à insérer de taille k
v : vecteur de coverture de taille n
q : matrice des racines du degré 2
c : matrice des racines du degré 3

Début de l'algorithme :

$m \leftarrow [m_1, m_2]^t$
$s_1 \leftarrow m_1 - v * H^t$
$s_2 \leftarrow m_2 - v * H^t$
if $s_1 = s_2 = 0$ **then**
    print "aucune modification n'est nécessaire"
else if $s_1^3 + s_2 = 0$ **then**
    $\beta_1 \leftarrow s_1$ // la racine du polynôme $\sigma(x)$ est s_1
    $j_1 \leftarrow log(\beta 1)$
    return j_1 // La position du coefficient a être modifié
else
    $u \leftarrow \dfrac{s_2 + s_1^3}{s_1^3}$
    $y_1 \leftarrow q(u, 1)$ // chercher u dans le look-up table q
    if $y_1 \neq -1$ **then**
        $\beta_1 \leftarrow s_1 * y_1$
        $\beta_2 \leftarrow s_1 * y_1 + y_1$
        $j_1 \leftarrow log(\beta 1)\,;\, j_2 \leftarrow log(\beta 2)$
        return $j_1; j_2$
    else
        for all i **in** *look-up table* c **do**
            $u \leftarrow k(i)$ // u est l'indice dans le look-up table c
            $y_1 \leftarrow c(u, 1)$
            $y_2 \leftarrow c(u, 2)$
            $y_3 \leftarrow c(u, 3)$
            $p \leftarrow \left(\dfrac{S_1^3 + S_2}{u}\right)^{1/3}$
            $\beta_1 \leftarrow p * y_1 + s_1$
            $\beta_2 \leftarrow p * y_2 + s_1$
            $\beta_3 \leftarrow p * y_3 + s_1$ // les trois racines de $\sigma(x)$
            $j_1 \leftarrow log(\beta 1)$,
            $j_2 \leftarrow log(\beta 2)$
            $j_3 \leftarrow log(\beta 3)$ // les positions des coefficients a être modifiés
            return $j_1\,;\, j_2\,;\, j_3$
        end for
    end if
end if

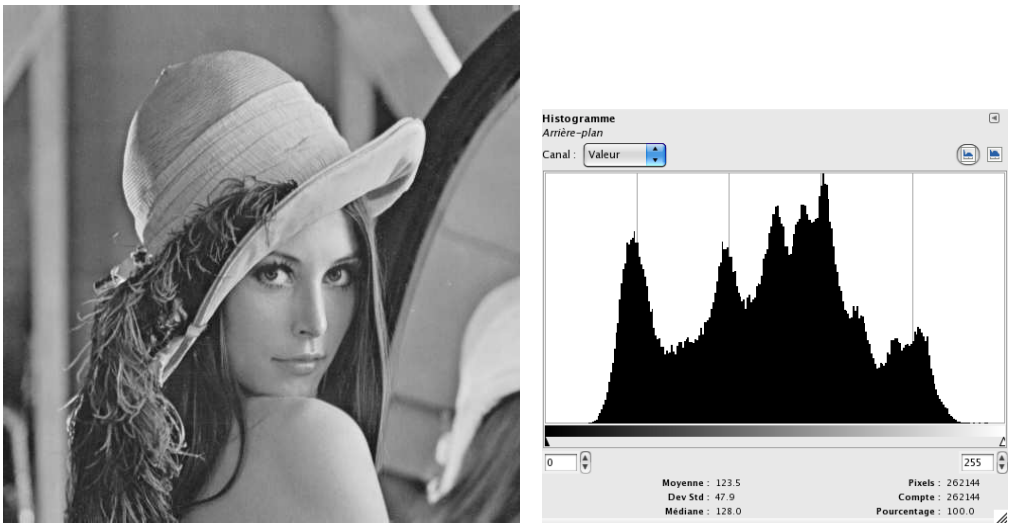
4.4 Tests et résultats

Pour vérifier l'invisibilité et la différence des histogrammes, l'algorithme proposé par [1] a été testé sur différentes images en niveau de gris sur le format PGM (512×512).
La capacité d'insertion du médium de couverture est la taille de l'image, c'est-à-dire (512×512).
On a utilisé pour ces preuves un message de 23 octets, équivalent à 184 bits.
La première et la plus simple évaluation à faire est l'imperceptibilité. Les changements dans la image stéganographie sont invisibles à l'œil humain.
Les images et leurs histogrammes sont montrés, pour l'image hôte dans la figure 4.2 et pour l'image stéganographiée dans la figure 4.3.



(a) (b)

FIGURE 4.2 – Image de Lena (a) et son histogramme (b)



(a) (b)

FIGURE 4.3 – Image de Lena stéganographiée (a) et son histogramme (b)

Il existe une faible différence entre les histogrammes des images hôte et stego. Cette différence est plus perceptible si la taille du message à insérer augmente.

Chapitre 5

Conclusion et Perspectives

Dans ce stage, nous avons a étudié la stéganographie en utilisant le codage par syndrome BCH. Cette technique a été introduite par Schönfeld et Winkler [14]. La technique de Schönfeld et Winkler a été améliorée par Zhan, Sachnev et Kim dans [1].
La comparaison de cette technique avec d'autres techniques moins performantes est montrée dans [1]. L'utilisation des *look-up tables* pour le calcul des racines de degré 2 et 3 réduisent le temps de calcul. Cette amélioration a été baptisée *fastBCH*.
Cette amélioration permet de réduire la complexité de calcul des racines du polynôme générateur d'erreurs $\sigma(x)$ de $O(2^k)$ à $O(n)$ (voir [1]).
L'algorithme utilisé pour remplir les tables de correspondance est l'algorithme de Chien. La complexité de cet algorithme croit exponentiellement avec la taille du code BCH. Dans ce travail, j'ai proposé l'utilisation d'une autre méthode pour le calcul des racines de degré 2. Cette technique permet le calcul en utilisant les éléments de base d'un corps de Galois.
Aussi, dans ce travail, je présente un algorithme pour la création de corps de Galois de la forme F_2^m. Cette table contenant des éléments du corps de Galois sert comme une table de correspondance pour le calcul des opérations sur le corps ainsi que pour le calcul des indices du polynôme d'erreur $e(x)$.
La méthode stéganograhique étudiée dans ce travail est encore jeune et il y a beaucoup de chemin à parcourir. Les perspectives pour la suite de ce travail sont :

- Utiliser les techniques de traitement d'image, comme par exemple, filtre pas bas, manipulation d'histogramme, et cætera, afin de rendre une estimation plus précise de l'évaluation de cette technique.
- L'analyse de la sécurité et la comparaison des performances avec d'autres algorithmes moins performantes.

Annexes

Annexe A

Factorisation de $x^n - 1$

La factorisation du polynôme $x^n - 1$ joue un rôle important dans la recherche de tous les codes cycliques de longueur n sur F_q. Comme le polynôme générateur d'un code cyclique de longueur n sur F_q est un diviseur de $x^n - 1$, on est intéressé à déterminer les facteurs irréductibles de ce polynôme.

A.1 Racines n-ièmes de l'unité

Étant donné un nombre entier naturel non nul n, la racine n-ième de l'unité ou numéro de Moivre, est un nombre complexe dont la puissance n-ième vaut 1, c'est-à-dire, pour un entier naturel non nul n donné, une racine n-ièmes de l'unité est l'ensemble des nombres complexes x solution de l'équation $x^n = 1$.
Il existe n différentes racines n-ièmes de l'unité. Les n racines n-ièmes de l'unité forment un groupe cyclique (cf. définition 2.1.1) d'ordre n pour la multiplication des nombres complexes avec 1 comme élément neutre.
Les racines n-ièmes de l'unité sont de la forme $e^{2\pi ik/n}$ où n et k sont premiers entre eux et $k = (0, 1, 2, ..., n-1)$.
Une racine n-ième de l'unité est dite primitive quand elle est un générateur de ce groupe cyclique (cf. définition 2.1.1).

A.2 Classes cyclotomiques

Les classes cyclotomiques permettent de déterminer le nombre de facteurs irréductibles de $x^n - 1$. Elles permettent de trouver tous les polynômes minimaux de $x^n - 1$ (i.e. tous les facteurs de $x^n - 1$).
On appelle classe cyclotomique q-aire modulo n de l'entier i, l'ensemble des entiers modulo n de la forme iq^j

Pour $0 \leqslant i < n$ on note $C_i = \{i, iq, iq^2, ..., iq^j\}$ la classe cyclotomique de q modulo n qui contient l'entier i avec j le plus petit entier positif tel que :

$$iq^j \equiv 1 \bmod n.$$

Il existe une correspondance bijective entre les facteurs irréductibles de $x^n - 1$ et les classes cyclotomiques q-aires module n. En particulier, le nombre de facteurs irréductibles de $x^n - 1$ est égal au nombre de classes cyclotomiques. Le degré d'un facteur irréductible est égal au cardinal de la classe cyclotomique associée.
L'entier i est souvent appelé *chef de classe* ou *coset leader* en anglais.

Exemple A.1 On se place sur le corps fini de caractéristique 2 et longueur 3, c'est-à-dire, sur un corps F_2^3. On veut construire ses classes cyclotomiques :

Pour $i = 0$
$j = 1, 2^1 \times 0 = 0 \bmod 7 = 0$
Alors $C_0 = \{0\}$

Pour $i = 1$
$j = 0, 2^0 \times 1 = 1 \bmod 7 = 1$
$j = 1, 2^1 \times 1 = 2 \bmod 7 = 2$
$j = 2, 2^2 \times 1 = 4 \bmod 7 = 4$
Alors $C_1 = \{2, 4, 1\}$

Pour $i = 2$
$j = 0, 2^0 \times 2 = 2 \bmod 7 = 2$
$j = 1, 2^1 \times 2 = 4 \bmod 7 = 4$
$j = 2, 2^2 \times 2 = 8 \bmod 7 = 1$
Alors $C_2 = \{4, 1, 2\}$

Pour $i = 3$
$j = 0, 2^0 \times 3 = 3 \bmod 7 = 3$
$j = 1, 2^1 \times 3 = 6 \bmod 7 = 6$
$j = 2, 2^2 \times 3 = 12 \bmod 7 = 5$
Alors $C_1 = \{6, 5, 3\}$

Et caetera.
Finalement, on a :

$C_0 = \{0\}$
$C_1 = C_2 = C_4 = \{1, 2, 4\}$

$C_3 = C_5 = \{3, 5, 6\}$

La décomposition en facteurs irréductibles dans F_2^3 de $x^7 - 1$ comporte donc un facteur de degré 1, et deux facteurs de degré 3.
Remarques :
Comme on a vu dans la section 2.1.3 un corps de Galois $GF(q^m)$ de longueur $n = q^m - 1$ peut être représente par ses éléments primitifs α :

$$GF(q^m) = \{1, \alpha, \alpha^2, \ldots \alpha^{q^m-2}\}$$

où α est une racine d'un polynôme $p(x)$ de degré m irréductible dans $GF(q^m)$. Ce polynôme doit de plus être primitif, c'est-à-dire qu'on ne trouve pas d'autre puissance j inférieure à $q^m - 1$ telle que $\alpha^j = 1$, pour tout j tel que :

$$\left(\alpha^j\right)^{q^m-1} = 1$$

Donc $1, \alpha, \alpha^2, \ldots \alpha^{q^m-2}$ sont les racines du polynôme $x^n - 1$.
Comme ce polynôme de degré $q^m - 1$ a $q^m - 1$ racines, on peut écrire :

$$x^{q^m-1} - 1 = (x-1)(x-\alpha)(x-\alpha^2)(x-\alpha^3)\ldots(x-\alpha^j)\ldots(x-\alpha^{q^m-2})$$

comme $n = q^m - 1$, on peut écrire aussi :

$$x^n - 1 = \prod_{i=0}^{n-1}(x - \alpha^i)$$

Avec les classes cyclotomiques, on peut trouver les facteurs irréductibles de $x^n - 1$.
Soient $m_i(x)$ les facteurs irréductibles de $x^n - 1$, alors :

$$m_i(x) = \prod_{j \in C_i}(x - \alpha^j)$$

Exemple A.2 En continuant avec l'exemple A.1, pour $n = 7$ et $q = 2$, on avait les classes cyclotomiques suivantes :

$C_0 = \{0\}$
$C_1 = C_2 = C_4 = \{1, 2, 4\}$
$C_3 = C_5 = \{3, 5, 6\}$

A l'aide de la table des éléments du corps F_2^3 (table 3.2), on construit les polynômes irréductibles sur F_2^3. Attention : la somme et la différence

sont des opérations similaires dans les *corps*.

$m_0 = x - 1 = x + 1$

$m_1 = (x - \alpha)(x - \alpha^2)(x - \alpha^4)$
$m_1 = (x^2 + \alpha x + \alpha^2 x + \alpha^3)(x - \alpha^4)$
$m_1 = x^3 + \alpha x^2 + \alpha^2 x^2 + \alpha^3 x + \alpha^4 x^2 + \alpha^5 x + \alpha^6 x + \alpha^7$
$m_1 = x^3 + (\alpha + \alpha^2 + \alpha^4)x^2 + (\alpha^3 + \alpha^5 + \alpha^6)x + \alpha^7$
$m_1 = x^3 + x + 1$

$m_3 = (x - \alpha^3)(x - \alpha^6)(x - \alpha^5)$
$m_3 = (x^2 + \alpha^6 x + \alpha^3 x + \alpha^9)(x - \alpha^5)$
$m_3 = x^3 + \alpha x^6 + \alpha^3 x^2 + \alpha^9 x + \alpha^5 x^2 + \alpha^{11} x + \alpha^8 x + \alpha^{14}$
$m_3 = x^3 + (\alpha^6 + \alpha^3 + \alpha^5)x^2 + (\alpha^9 + \alpha^{11} + \alpha^8)x + \alpha^{14}$
$m_3 = x^3 + x^2 + 1$

Finalement les facteurs irréductibles de $x^7 - 1$ sont :

$$x^7 - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1)$$

La décomposition de $x^n - 1$ nous permet de connaître tous les codes cycliques binaires de longueur n.

Exemple A.3 Le tableau A.1 montre les codes cycliques binaires qu'on peut former sur le corps F_2^3 avec les résultats de l'exemple A.2

	$g(x)$	Dimension
C_0	$m_0(x)m_1(x)m_3(x) = x^7 - 1$	0
C_1	$m_1(x)m_3(x) = x^6 + x^5 + x^4 + ... + 1$	1
C_2	$m_0(x)m_1(x) = x^4 + x^3 + x^2 + 1$	3
C_3	$m_0(x)m_3(x) = x^4 + x^2 + x - 1$	3
C_4	$m_1(x) = x^3 + x + 1$	4
C_5	$m_3(x) = x^3 + x^2 + 1$	4
C_6	1	7

TABLE A.1 – Codes cycliques binaires sur le corps F_2^3

Bibliographie

[1] Rongyue Zhang, Vasiliy Sachnev, Hyoung Joong Kim, *Fast BCH syndrome coding for steganography* ; S. Katzenbeisser and A.-R. Sadeghi (Eds.), IH 2009, LNCS 5806, pp. 44-58, Springer-Verlag Berlin Heiderbelg 2009.

[2] Vasiliy Sachnev, Hyoung Joong Kim, Rongyue Zhang, *Less detectable JPEG steganography method based on heuristic optimization and BCH syndrome coding* ; MM& Sec '09, Princeton, New Jersey, USA, Septembre 2009.

[3] Bruno Martin, *Codage, cryptologie et applications* ; 1^e édition, Presses polytechniques et universitaires romandes, France, 2004 ; pp. 77 – 85.

[4] Richard E. Crandall, *Some notes on Steganography* ; Available at `http://os.inf.tu-dresden.de/~westfeld/crandall.pdf` - 1998.

[5] A. Westfeld : *High Capacity Despite Better Steganalysis (F5 - A Steganographic Algorithm)* ; In : LNCS, vol. 2137, Springer-Verlag, New York, pp. 2001, 289-302

[6] Krzysztof Wesolowski *Introduction to Digital Communication Systems* ; 1^e édition, John Wiley and Sons, England 2009 ; pp. 131 - 134.

[7] Michel Cohen, Jean-Luis Dornstetter, Philippe Godlewsky , *Codes correcteurs d'erreurs* ; 1^e édition ; Masson et CNET-ENST, Paris, 1992 ; chapitres 3 et 4.

[8] Michel Demazure , *Cours d'algèbre* ; 1*e* édition ; Cassini, Paris, 1997, chapitres 8 et 9.

[9] Peter Sweeney , *Error control coding : from theory to practice* ; 1*e* édition ; John Wiley and Sons, England, 2002, chapitre 4.

[10] Khira Lamèche, *Les codes en informatique* ; 1^e édition, Hermes, France, 1995 ; chapitre 2, pp. 35 – 53.

[11] Claude Berrou *Codes et turbocodes* ; 1^e édition, Springer - Verlag, France, 2007 ; pp. 125 - 130.

[12] Jean-Guillaume Dumas, Jean-Louis Roch, Eric Tannier, Sébastien Varrette *Théorie des codes : Compression, cryptage, correction* ; 1^e édition, Dunod, France, 2007 ; chapitre 4.

[13] Lih-Jyh Weng, Shrewsbury, Mass, *System and method for determinig the cube root of an element of a Galois Field gf(2)* ; United States Patent, US, 1998.

[14] Schönfeld, D., Winkler, A. *Embedding with syndrome coding based on BCH codes.* In : Proceedings of the 8th ACM Workshop on Multimedia and Security, pp. 214–223 (2006)