REPUBLIQUE DU CAMEROUN
Paix – Travail – Patrie
****

UNIVERSITE DE YAOUNDE I
ECOLE NORMALE SUPERIEURE
DEPARTEMENT DE DEPARTEMENT DE MATHEMATIQUES
****

REPUBLIC OF CAMEROUN
Peace – Work – Fatherland
****

UNIVERSITY OF YAOUNDE I
HIGHER TEACHER TRAINING COLLEGE
DEPARTMENT OF DEPARTMENT OF MATHEMATICS
****

LE CRYPTOSYSTEME McELIECE

Mémoire de DI.P.E.S.II de mathématiques

Par :

NJOKOU NJUEYA SEMENOU EDMOND
Licencié en Mathématiques

Sous la direction
Dr NDJEYA SELESTIN
Chargé de Cours

Année Académique
2015-2016

♣ Dédicace ♣

Je dédie ce mémoire à mon père NJOKOU JOSEPH.

♣ Remerciements ♣

Bien que ce mémoire soit un projet personnel, je n'aurais jamais pu le réaliser sans l'aide de nombreuses personnes.

Tout d'abord un grand merci à mon encadreur, le docteur NDJEYA Selestin qui a eu l'amabilité de m'encadrer pour l'élaboration de ce travail malgré ses nombreuses sollicitations. Je ne peux oublier TALE Hervé qui m'a prodigué de nombreux conseils et fourni une bibliothèque riche et indispensable pour mon travail.

Merci à ATAMEWOUE Surdive, KAM Patrick, TSASSE Pelvilin, NGANMENI Hervé, DOUNTIO Martin pour les échanges intéressants que nous avons eu durant ces deux années et puis KANYOU Claude qui m'a permis d'apprendre beaucoup de choses sur l'utilisation des logiciels Maple et Matlab.

Une énorme pensée à toute ma famille qui a toujours cru en moi et soutenu sous diverses formes. Je pense bien sûr à mes tantes, mes cousins et cousines, mes neveux et nièces : tous autant adorables les uns que les autres. Ils se reconnaitront, enfin j'espère ! Surtout un grand merci à mes frères et sœurs NJOKOU Jerlaine, Eudoxie, Mariette, Edith, Gautier, Habib, Patrick, Marcel. Non, docteur NJOKOU Gaetan, je ne t'ai pas oublié ! Outre d'être un grand frère exemplaire, tu as été un modèle sur lequel je me suis appuyé constamment. Il est évident que je n'aurais jamais défendu ce mémoire sans toi.

Je pense également à la famille NGONGANG, à Rosita AKO, à tous mes amis qui ont toujours été là pour moi dans les moments importants. J'ai su profité de vos conseils.

Je remercie mes camarades de promotion pour la bonne humeur qui a toujours régné entre nous, pour leur disponibilité et même les remarques, suggestions qu'ils ont relevées afin de rehausser l'éclat de ce travail.

À toutes les personnes que je n'ai pas citées qui se reconnaitront je dis merci.

♣ Déclaration sur l'honneur ♣

Le présent travail est une œuvre du candidat et n'a été soumis nulle part ailleurs pour une autre évaluation académique. Les contributions externes ont été dûment mentionnées et recensées en bibliographie.

Signature du candidat

NJOKOU NJUEYA SEMENOU EDMOND

♣ Table des matières ♣

♣ Résumé ♣

Dans ce mémoire, nous étudions le cryptosystème McEliece. C'est un cryptosystème à clé publique basé sur les codes correcteurs d'erreurs de Goppa et dont la sécurité repose sur les problèmes difficiles de décodage d'un code aléatoire et de décodage par syndrome. Après une présentation des codes de Goppa binaires, nous implémentons ce cryptosystème à travers ses algorithmes de génération des clés, de chiffrement et de déchiffrement. Nous présentons également les attaques susceptibles d'être menées contre ce cryptosystème. L'inconvénient majeur de celui-ci est la taille des clés trop grande, ce qui empêche son utilisation courante. T. Berger et *all* proposent l'utilisation des codes de Goppa quasi-cycliques pour réduire la taille des clés.

Mots clés : Cryptographie, clé publique, corps finis, Codes de Goppa, McEliece.

♣ Abstract ♣

In this dissertation, we are interested in the study of the McEliece cryptosystem. It is a public key cryptosystem based on Goppa codes whose security rests on the difficult problem of decoding an unknown error-correcting code and hardness problem of syndrom decoding. We give an example of keys setup, encryption and decryption with the McEliece cyptosystem as well as a brief introduction to Goppa codes. Some examples of attacks to the cryptosystem are also given. One modification to this cryptosystem using quasi-cyclic Goppa codes proposed by T. Berger and *all* increases the security of the system and reduces the key size of the code.

Keywords : Cryptography, public key, finite fields, Goppa codes, McEliece.

♣ Table des figures ♣

♣ Liste des tableaux ♣

♣ Introduction générale ♣

Le monde est aujourd'hui extrêmement informatisé et la cryptographie qui sert à sécuriser les communications y est très présente. En 1978 Robert McEliece présente le premier cryptosystème asymétrique dont la sécurité est basée sur un problème considéré comme difficile dans la théorie des codes correcteurs McEliece(1978). L'avantage de ce cryptosystème est qu'il est à priori résistant à l'ordinateur quantique. Si les ordinateurs quantiques devenaient plus performants alors les cryptosystèmes basés sur la théorie des nombres tel que le RSA (Rivest et *al.* 1978) deviendraient obsolètes (Shor 1995) alors que ceux basés sur la théorie des codes correcteurs resteraient aussi sûr à priori.

Dans ce mémoire nous présentons le cryptosystème McEliece dont l'algorithme de chiffrement transforme un message fixe en un mot de code de Goppa bruité de sorte que seul son destinataire connaissant la structure exacte du code et son algorithme de décodage puisse déchiffrer le message. L'avantage que nous relevons ici est qu'il utilise des opérations simples pour fonctionner, il est donc plus rapide que RSA qui doit faire des exponentiations sur des grand nombres. Son inconvénient est d'utiliser des données de grandes tailles et les codes utilisés ont des matrices avec beaucoup de coordonnées.

Ce mémoire est organisé comme suit : dans le premier chapitre nous introduisons tout d'abord de façon générale la cryptographie moderne, en particulier la cryptographie à clé publique et par la suite nous explicitons la construction des corps finis $\mathbb{F}_{p^n}$ où p est un entier premier et n un entier non nul. Dans le deuxième chapitre nous définissons les objets appelés codes correcteurs et énonçons des résultats connus concernant leur utilisation en cryptographie suivi de la construction d'un code de Goppa binaire. Le troisième chapitre présente dans un premier temps le cryptosystème McEliece, ensuite une mise en œuvre de celui-ci et enfin quelques attaques connues contre le cryptosystème McEliece ; lequel cryptosystème est l'un des rares à encore résister à toute cryptanalyse (Cayrel 2008).

Chapitre Un

INTRODUCTION À LA CRYPTOGRAPHIE ET BASES MATHÉMATIQUES

1.1 Introduction à la cryptographie

La cryptographie est l'ensemble des techniques permettant d'écrire un message de façon brouillée afin que seul son destinataire légitime soit capable de comprendre la teneur du message. Il a toujours été très difficile de garantir la sécurité du canal dans lequel transite un message. La problématique d'établir des communications sécurisées en utilisant un canal non sécurisé a toujours été d'importance en vue d'applications financières ou encore du respect de la vie privée. En pratique, on demande à la cryptographie d'assurer les problèmes suivants :

- Confidentialité : Ceux qui ne sont pas les destinataires d'une information ne doivent pas avoir accès à cette information.
- Authentification : il doit être possible pour le récepteur du message de garantir son origine. Une tierce personne ne doit pas pouvoir se faire passer pour quelqu'un d'autre (usurpation d'identité).
- Intégrité : le récepteur doit pouvoir s'assurer que le message n'a pas été modifié durant sa transmission. Une tierce personne ne doit pas pouvoir substituer un message légitime (ayant pour origine l'émetteur) par un message frauduleux.
- Non répudiation : un émetteur ne doit pas pouvoir nier l'envoi d'un message.

Définition 1.1.1. *Un système de chiffrement ou cryptosystème est un 5-uplet $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ où $\mathcal{P}$ est l'espace des messages clairs, $\mathcal{C}$ est l'espace des messages chiffrés, $\mathcal{K}$ est l'espace*

des clés, $\mathcal{E} = \{E_k, k \in \mathcal{K}\}$ *avec* $E_k : \mathcal{P} \longrightarrow \mathcal{C}$ *(l'espace des fonctions de chiffrement) et* $\mathcal{D} = \{D_k, k \in \mathcal{K}\}$ *avec* $D_k : \mathcal{C} \longrightarrow \mathcal{P}$ *(l'espace des fonctions de déchiffrement).*

Remarque 1.1.1. A chaque clé e de $\mathcal{K}$ est associée une clé d de $\mathcal{K}$ telle que $D_d(E_e(m)) = m$ pour tout $m \in \mathcal{P}$. Un cryptosystème fonctionne de la façon suivante :

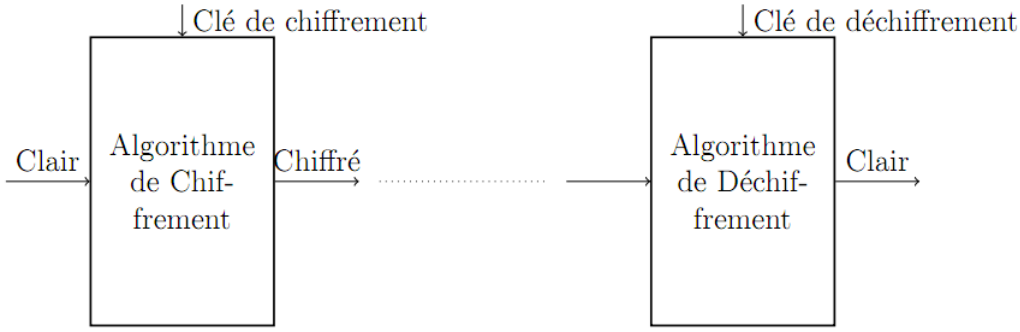

FIGURE 1.1 – Schéma illustratif du fonctionnement d'un cryptosystème.

Traditionnellement, la sécurité du cryptosystème reposait sur la non divulgation des algorithmes de chiffrement et de déchiffrement. En 1883, Auguste Kerchoff (Kerchoff 1883) énonce comme principe que la sécurité d'un cryptosystème doit reposer uniquement sur le secret des clés, et que les algorithmes doivent être supposés publiquement connus. Cette contrainte semble rendre plus ardue l'invention d'un cryptosystème fiable mais présente deux grands avantages. Premièrement, en cas de divulgation du secret, la réparation est simple. Si le cryptosystème est intrinsèquement sûr, il suffit de générer de nouvelles clés, et l'on peut reprendre les communications. Deuxièmement, l'inventeur d'un cryptosystème peut, sans trahir de secret, révéler ses algorithmes à ses collègues cryptographes. Ceux-ci soumettront le cryptosystème à de multiples examens, à la recherche de la moindre faille, et proposeront parfois des améliorations, avant que l'on aboutisse à un cryptosystème considéré comme sûr.

Définition 1.1.2.
- *Le chiffrement est la fabrication du message chiffré à partir du clair et de la clé de chiffrement.*
- *Le déchiffrement est l'extraction du message clair à partir du chiffré en utilisant la clé de déchiffrement.*
- *L'attaque est l'extraction du message clair à partir du chiffré sans connaitre la clé de déchiffrement.*

- *La cryptanalyse est l'étude théorique d'un système de chiffrement en vue de mettre au point des algorithmes de décryptage.*
- *La cryptologie est la branche des mathématiques qui regroupe la cryptographie et la cryptanalyse.*

Remarque 1.1.2. Un bon cryptosystème doit permettre un chiffrement et un déchiffrement rapide, tout en interdisant toute possibilité de décryptage. Pour alléger les charges en mémoire informatique, on préfère aussi que les clés soient de petite taille.

Remarque 1.1.3. Mathématiquement, la sécurité du cryptosystème semble impossible à atteindre. En effet, l'algorithme de déchiffrement est connu de tous, et seule la clé de déchiffrement est secrète or, si cette clé est de petite taille, un attaquant peut donc en théorie essayer toutes les clés possibles. S'il est capable de les essayer suffisamment vite, il finira par trouver la clé de déchiffrement et aura alors décrypté le message chiffré. On va donc supposer que la puissance de calcul disponible pour l'attaquant n'est pas infinie.

On définit donc un paramètre pratique de sécurité cryptographique, qui correspond à une limite maximale de la puissance de calcul d'un attaquant raisonnable. Actuellement, on considère que ce paramètre vaut 2^{80} Faure(2009) : si une attaque effectue en moyenne plus de 2^{80} opérations pour décrypter un message, cette attaque ne menace pas la sécurité du cryptosystème car n'étant pas à la portée d'une organisation humaine. Bien sûr, le nombre 2^{80} est valable à l'heure actuelle, il faudra le modifier dans le futur pour tenir compte de l'amélioration des performances informatiques.

Notation 1.1.1. Quand on construit un algorithme cryptographique, il est nécessaire d'estimer le temps de calcul ainsi que la taille du stockage nécessaires à son exécution. Pour simplifier ce genre d'estimation, on introduit la notation $\mathcal{O}$.

Définition 1.1.3. *Soient k un entier positif, $X, Y \subset \mathbb{N}^k$ et $F : X \longrightarrow \mathbb{R}^+$, $G : Y \longrightarrow \mathbb{R}^+$ On écrit : F=$\mathcal{O}(G)$ et F est dit en $\mathcal{O}(G)$ s'il existe des entiers positifs B et C tels que : $\forall\, (n_1, n_2, \cdots, n_k) \in X \cap Y$ où $n_i > B$, $1 \leq i \leq k$, $F(n_1, n_2, \cdots, n_k) \leq C\, Y(n_1, n_2, \cdots, n_k)$. Quand G est constante, on écrit $F = \mathcal{O}(1)$.*

Exemple 1.1.1. Pour tout $n \in \mathbb{N}$ et $n \geq 1$, on a $2n^2+n+1 \leq 4n^2$. Si $F : n \longmapsto 2n^2+n+1$ et $G : n \longmapsto n^2$ alors on dira que F est en $\mathcal{O}(G)$.

Remarque 1.1.4. Dans de nombreuses applications cryptographiques on doit ajouter, multiplier, ou diviser avec reste des entiers. Pour estimer le temps d'exécution de telles applications, il faut connaître le temps pris par ces opérations. Pour cela, on doit choisir dans Buchman(2006) un modèle de calcul le plus proche possible des ordinateurs.

Les estimations suivantes seront utilisées par la suite : Soient a et b des entiers de longueurs binaires respectives m et n.

- Ajouter a et b demande un temps $\mathcal{O}(max\{m, n\})$.
- Multiplier a par b demande un temps $\mathcal{O}(m \times n)$.
- Diviser a par b avec reste demande un temps $\mathcal{O}(n \times q)$, où q est la longueur binaire du quotient de la division. Tous ces algorithmes utilisent en espace $\mathcal{O}(m + n)$.

Définition 1.1.4. *Supposons qu'un algorithme reçoive en données des entiers $n_1, \cdots, n_k$ On dit que cet algorithme a un temps d'exécution polynomial ou qu'il s'agit d'un algorithme polynomial, si le temps d'exécution de l'algorithme est $\mathcal{O}(|n_1|^{e_1} \cdots |n_k|^{e_k})$ avec des entiers naturels $e_1, \cdots, e_k$.*

Définition 1.1.5. *Un algorithme est jugé efficace quand son temps d'exécution est polynomial.*

Remarque 1.1.5.
- Pour juger de la validité d'un mécanisme de chiffrement, on va considérer la complexité (ou encore le nombre d'opérations) de l'attaque la plus rapide contre le système. Généralement, on estime que le mécanisme peut être utilisé en cryptographie si la complexité de l'attaque croît exponentiellement avec les paramètres du système (taille de la clé ou du message par exemple), tandis que la complexité du chiffrement et du déchiffrement croît polynomialement avec ces mêmes paramètres. Comme l'exponentielle croît beaucoup plus vite que tout polynôme, on peut alors espérer que, pour certains paramètres, les attaques deviendront infaisables en pratique, alors que le chiffrement et le déchiffrement resteront rapides.
- Montrer qu'un cryptosystème n'est pas sûr est relativement facile : il suffit de trouver un algorithme d'attaque suffisamment rapide. Par contre, prouver la sécurité du système est extrêmement difficile. Même en considérant toutes les méthodes d'attaque connues, rien ne prouve qu'il n'existe pas un algorithme inédit, plus rapide que tout ce qui existe.

On distingue deux classes de cryptosystèmes :

Définition 1.1.6. *Les cryptosystèmes à clé secrète ou symétriques : Les clés de chiffrement et de déchiffrement sont identiques. L'expéditeur du message doit donc au préalable partager un secret sur la clé avec le destinataire.*

Remarque 1.1.6. L'avantage est que l'on peut atteindre la sécurité cryptographique en utilisant des clés très courtes (une centaine de bits) et en conservant un chiffrement et un déchiffrement extrêmement rapide.

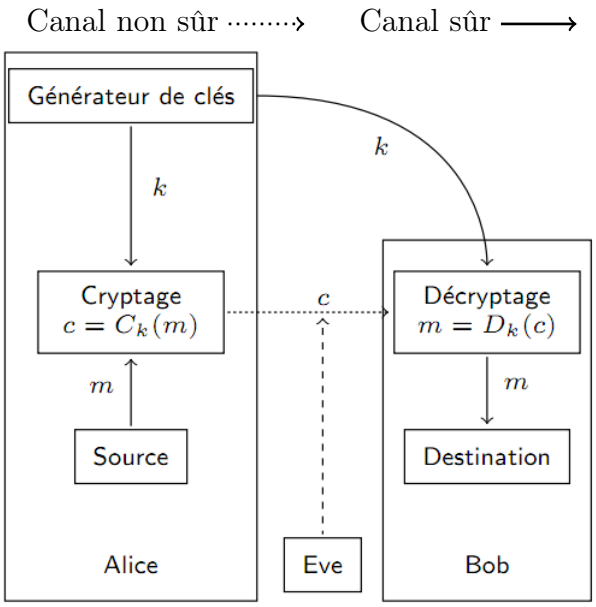

FIGURE 1.2 – Cryptosystème à clé secrète.

Exemple 1.1.2. Cryptosytème de Vigenère

Vigenère définit une correspondance lettre $\longleftrightarrow$ nombre : A $= 0$; B $= 1$; $\cdots$; Z $= 25$.

On a $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}^n$ $(n \in \mathbb{N}^*)$. Soit $k = (k_1, \cdots, k_n) \in \mathcal{K}$, $m = (m_1, \cdots, m_n) \in \mathcal{P}$ et $c = (c_1, \cdots, c_n) \in \mathcal{C}$.

La fonction de chiffrement E_k associée à k est telle que :

$E_k(m_1, \cdots, m_n) = (m_1 + k_1, \cdots, m_n + k_n) = (c_1, \cdots, c_n)$.

La fonction de déchiffrement D_k associée à k est telle que :

$D_k(c_1, \cdots, c_n) = (c_1 - k_1, \cdots, c_n - k_n) = m = (m_1, \cdots, m_n)$.

Le caractère clair « espace » a également pour chiffré le caractère « espace » et vice-versa. Ainsi pour $n = 3$ et $k = (2, 11, 4)$, le texte clair **CRYPTOGRAPHIE** est chiffré par blocs de 3 lettres en **ECCRESICERSMG** et la seule lettre claire **E**$\longleftrightarrow$**4** est chiffré en **G**$\longleftrightarrow$**6** avec la première composante de la clé k puisque $4 + 2 = 6$ *mod* 26.

Exemple 1.1.3. Dans cet exemple on considère $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$. L'espace des clés est l'ensemble des permutations de $\mathbb{Z}_{26}$. Pour $k \in \mathcal{K}$, on a $E_k = k$ et $D_k = k^{-1}$.

Ainsi pour la permutation suivante,

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
p	o	m	l	i	k	u	j	n	y	h	b	t	g	r	f	v	d	c	e	z	s	x	a	q	w

le texte clair **CRYPTOGRAPHIE** est chiffré en **MQDFERUDPFJNI**.

Définition 1.1.7. *Les cryptosystèmes à clé publique ou asymétriques : Les clés de chiffrement et de déchiffrement sont différentes, et la connaissance de la clé de chiffrement (ou clé publique) ne permet pas de retrouver la clé de déchiffrement (ou clé privée) associée. On considère donc que la clé publique est connue de tous et que le sécurité du système repose uniquement sur le secret de la clé privée. L'expéditeur et le destinataire n'ont pas besoin d'avoir échangé de secret sur la clé pour communiquer de façon chiffrée.*

Remarque 1.1.7. Ces cryptosystèmes sont très utiles pour les signatures numériques et l'initialisation d'une communication. Ils sont par ailleurs moins efficaces (clés plus longues, algorithmes plus lents) que les cryptosystèmes symétriques.

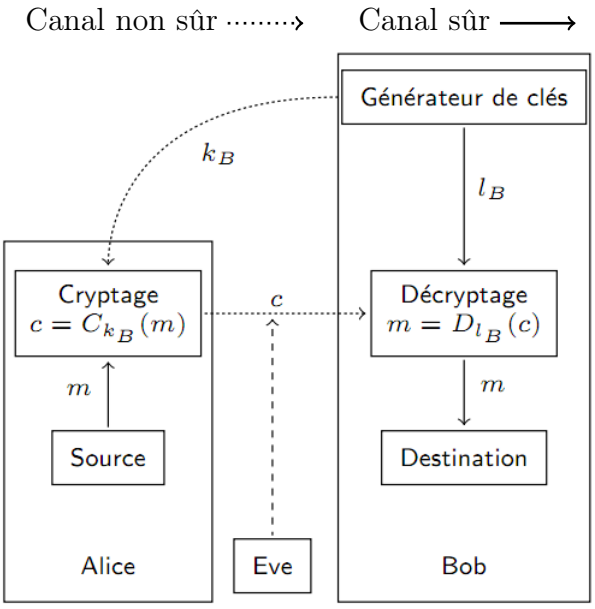

FIGURE 1.3 – Cryptosystème à clé publique.

Exemple 1.1.4. Le Cryptosystème RSA.

Le principe est le suivant

- ⋆ Bob choisit deux nombres premiers p_b et q_b distincts. Il note $n_b = p_b q_b$, le *module du cryptosystème* et calcule $\varphi(n_b) = (p_b - 1)(q_b - 1)$, l'indicatrice d'Euler de n_b.
- ⋆ Bob choisit ensuite un nombre e_b l'*exposant de la clé publique* premier avec $\varphi(n_b)$. Ainsi PGCD$(e_b, \varphi(n_b)) = 1$. En pratique on impose que $1 \leq e_b \leq \varphi(n_b) - 1$.
- ⋆ Puis Bob calcule l'inverse d_b de e_b modulo $\varphi(n_b)$, d_b est appelé l'*exposant de la clé secrète*. En pratique, $d_b \in \mathbb{N}$ tel qu'il existe $k_b \in \mathbb{N}$ avec $e_b \times d_b = 1 + k_b\varphi(n_b)$ et $1 \leq d_b \leq \varphi(n_b) - 1$.
- ⋆ Bob publie les nombres n_b et e_b qui forment sa *clé publique de chiffrement* et garde secret p_b, q_b et d_b qui forment sa *clé secrète de déchiffrement*.

Quand Alice veut communiquer avec Bob,

1. Alice commence par transformer son message m en une suite de chiffres, par exemple en remplaçant les lettres et les différents symboles utilisés par des chiffres (de 0 à 255 dans le cas du code ASCII).
2. Elle calcule $E_{e_b,n_b}(m) = c = m^{e_b}\ mod\ n_b$ et envoie le chiffré c à Bob.
3. Pour récupérer le texte en clair Bob calcule
   $D_{d_b}(c) = c^{d_b} \equiv m^{e_b d_b} = m^{1+k_b\varphi(n_b)}\ mod\ n_b$.
   D'après le théorème d'Euler on a $m^{\varphi(n_b)} \equiv 1\ mod\ n_b$
   et par conséquent $D_{d_b}(c) \equiv m\ mod\ n_b$.

Lorsque Bob choisit par exemple $p_b = 7$ et $q_b = 11$ alors $n_b = 77$ et $\varphi(n_b) = 6 \times 10 = 60$. Il choisit $e_b = 7$ car PGCD$(7, 60) = 1$. Avec l'algorithme étendu d'Euclide il calcule d_b

$$7 \times 43 = 5 \times 60 + 1 \Longrightarrow d_b = 43.$$

La clé publique de Bob est (7,77) et sa clé secrète est (7,11,43).
Alice veut transmettre le message 3 à Bob. Elle calcule donc

$$3^7 \equiv 31\ mod\ 77 = 7 \times 11$$

et elle envoie 31 à Bob. Bob décode en calculant

$$31^{43} \equiv 31^{1+2+8+32} \equiv 31.37.58.37.31 \equiv 3\ mod\ 77$$

Remarque 1.1.8. Les cryptosystèmes à clé publique sont difficiles à concevoir. La fonction de chiffrement (qui au clair associe le chiffré) doit être une fonction injective à sens unique (son inversion est difficile) avec trappe (la connaissance d'un secret rend l'inversion facile).

On connait peu d'objets de cette sorte, c'est pourquoi la plupart des cryptosystèmes asymétriques sont issus de l'un des domaines suivants :

- Arithmétique : L'inversion de la fonction à sens unique se rapporte au problème de factorisation des nombres entiers. Le système RSA utilise ce type de primitive.
- Courbes elliptiques : L'inversion de la fonction à sens unique se rapporte au problème du logarithme discret (Diffie et Hellman 1976) sur le groupe des points d'une courbe. Par exemple, le protocole de signature électronique ECDSA(Elliptic Curve Digital Signature Algorithm) utilise ce genre de constructions.
- Théorie des codes correcteurs : L'inversion de la fonction à sens unique se rapporte au problème du décodage d'un code aléatoire. Comme exemple, le schéma de chiffrement de McEliece.

Définition 1.1.8. *(Tison 2003) (Complexité des problèmes :NP, NP-dur, NP-complet).*

- *Un langage L est dit NP(Non-Déterministe Polynomial) si il existe un-polynôme Q et un algorithme polynomial à deux entrées et à valeurs booléennes tels que :*
  $L = \{u/\exists c,\ A(c,u) = Vrai, |c| \leq Q(|u|)\}$
  A : Algorithme de vérification ; c : certificat ou témoin ; $|c|$= taille du certificat.
  $|c| \leq Q(|u|)$ *: la taille du certificat est bornée polynômialement par la taille de l'entrée.*
- *On dit que A vérifie L en temps polynômial.*
- *Un problème est NP quand il existe un algorithme non déterministe polynômial qui décide de ce problème en un temps polynômial.*
- *Une propriété P est NP si il existe un algorithme non déterministe polynômial qui décide P.*
- *Une propriété R est dite NP-dure si toute propriété NP se réduit polynômialement en R.*
- *Une propriété est dite NP-complète si elle est NP et NP-dure.*

1.2 Notion de corps finis

Définition 1.2.1. *Un corps fini est un corps ayant un nombre fini d'éléments.*

Notation 1.2.1. Le corps fini de cardinal q est noté $\mathbb{F}_q$.

Exemple 1.2.1. Les $\frac{\mathbb{Z}}{p\mathbb{Z}}$ avec p premier sont des corps finis de cardinal p.

Proposition 1.2.1. *Soit K un corps fini à q éléments. Alors $\forall\ x \in K$, $x^q = x$.*

Théorème 1.2.1. *Si K est un corps fini, alors tout sous-groupe de K^* (groupe multiplicatif) est cyclique.*

Théorème 1.2.2. ***:Théorème de Wedderburn***

Tout corps fini est commutatif.

1.2.1 Construction des corps finis

Corps quotient sur un corps premier

Théorème 1.2.3. *(Kraus 2012). Soit p un nombre premier de $\mathbb{N}$ et $\mathbb{F}_p = \dfrac{\mathbb{Z}}{p\mathbb{Z}}$ le corps premier de cardinal p. Si f est un polynôme irréductible sur $\mathbb{F}_p$, alors l'anneau quotient $\dfrac{\mathbb{F}_p[X]}{(f)}$ est un corps fini.*

Proposition 1.2.2. *Soit n le degré d'un polynôme f irréductible sur $\mathbb{F}_p$. Alors :*

1. *Le corps $\dfrac{\mathbb{F}_p[X]}{(f)}$ est l'ensemble des polynômes g sur $\mathbb{F}_p$ de la forme $a_0x + a_1x + \cdots + a_{n-1}x^{n-1}$ muni de l'addition et de la multiplication modulo f.*
2. *Le cardinal du corps $\dfrac{\mathbb{F}_p[X]}{(f)}$ est p^n.*

Preuve.

1. Soit $g \in \dfrac{\mathbb{F}_p[X]}{(f)}$ alors $g = p + (f)$ où $p \in \mathbb{F}_p[X]$. Ainsi on a : $p = f.q + g$ avec $q \in \mathbb{F}_p[X]$. On déduit alors que le degré du polynôme g noté $deg\ (g)$ est inférieur à $deg\ (f)$.

2. L'application

$$\begin{array}{ccc} \dfrac{\mathbb{F}_p[X]}{(f)} & \longmapsto & (\mathbb{F}_p)^n \\ g & \to & (a_0, a_1, \ldots, a_{n-1}) \end{array}$$

est une bijection.
Le cardinal de $\dfrac{\mathbb{F}_p[X]}{(f)}$ est donc égale à $|(\mathbb{F}_p)^n| = |\mathbb{F}_p|^n = p^n$ ■

Remarque 1.2.1. Un corps fini K de caractéristique p admet p^n éléments où $n \in \mathbb{N}$. En effet, le nombre n est égale à la dimension de K considéré comme espace vectoriel sur $\mathbb{F}_p$: $n = dim_{\mathbb{F}_p}K$.

Exemple 1.2.2. Pour $p = 2$, on considère le polynôme f sur $\mathbb{F}_2$ défini par $f(x) = x^3 + x^2 + 1$. Puisque $f(0) = 1$ et $f(1) = 1$ (calculs dans $\mathbb{F}_2$) alors f n'a pas de racines dans $\mathbb{F}_2$. Il est donc irréductible sur $\mathbb{F}_2$ car sinon il aurait un diviseur de degré 1 et donc

une racine dans $\mathbb{F}_2$. Les éléments de $\frac{\mathbb{F}_2[X]}{(x^3+x^2+1)}$ sont les polynômes définis par : 0, 1, x, $x+1$, x^2, $1+x^2$, $x+x^2$ et $1+x+x^2$. Les calculs donnent :

TABLE 1.1 – Puissances de x et multiplication dans $\frac{\mathbb{F}_2[x]}{(x^3+x^2+1)}$.

i	0	1	2	3	4	5	6	7
x^i	1	x	x^2	$1+x^2$	$1+x+x^2$	$1+x$	$x+x^2$	1
$\times$	1	x	$1+x$	x^2	$1+x^2$	$x+x^2$	$1+x+x^2$	
1	1	x	$1+x$	x^2	$1+x^2$	$x+x^2$	$1+x+x^2$	
x	x	x^2	$x+x^2$	$1+x^2$	1	x	$1+x+x^2$	
$1+x$	$1+x$	$x+x^2$	$1+x^2$	1	x	$1+x+x^2$	x^2	
x^2	x^2	$1+x^2$	1	$1+x+x^2$	x	$1+x$	$x+x^2$	
$1+x^2$	$1+x^2$	$1+x+x^2$	x	$1+x$	$x+x^2$	x^2	1	
$x+x^2$	$x+x^2$	1	$1+x+x^2$	x	x^2	$1+x$	$1+x^2$	
$1+x+x^2$	$1+x+x^2$	x	x^2	$x+x^2$	1	$1+x^2$	$1+x$	

Proposition 1.2.3. *(Kraus 2012). Si K est un corps commutatif et P un polynôme irréductible dans $K[X]$ alors, $dim_K \frac{K[X]}{(P)} = deg(P)$.*

Preuve. Soit $A \in K[X]$. Notons $\bar{A}$ son image dans $\frac{K[X]}{(P)}$. La division euclidienne par P permet d'écrire : $A = q.P + R$ avec $deg\ (R) \leq deg\ (P)$. Cela définit une application à valeurs dans l'espace des polynômes de degré au plus égale à $P-1$. On a

$$\begin{array}{ccc} \frac{K[X]}{(P)} & \to & K[X]_{deg(P)-1} \\ \bar{A} & \mapsto & R \end{array}$$

C'est une application linéaire d'espace vectoriels. Elle est injective car si $R = 0$ alors $A = q.P$. Donc $\bar{A} = 0$. Elle est surjective car $\bar{R}$ a pour image R. ■

Remarque 1.2.2. On en déduit que si P est un polynôme irréductible dans $\mathbb{F}_p[X]$, alors le corps $\frac{K[X]}{(P)}$ a $p^{deg(P)}$ éléments.

1.2.2 Polynôme minimal

Proposition 1.2.4. *Soient K un corps fini de caractéristique p et $\beta \in K$. L 'ensemble I_β des polynômes à coefficients dans $\mathbb{F}_p$ ayant β comme racine dans K est un idéal de $\mathbb{F}_p[X]$.*

Preuve. Considérons K, un corps fini de cardinalité p^r. L'ensemble I_β n'est pas vide car le polynôme défini par $x^{p^r} - x$ est dans I_β. D'autre part, I_β est stable pour $+$. Enfin pour tout f dans $F_p[X]$, $fI_\beta \subset I_\beta$ car si $g = f.h$ avec $h(\beta) = 0$, alors $g(\beta) = 0$. ■

Définition 1.2.2. *Le générateur de l'idéal I_β de $\mathbb{F}_p[X]$ s'appelle le polynôme minimal de β. Il est noté m_β.*

Proposition 1.2.5. *Soit m_β un polynôme minimal de l'élément $\beta \in K$, alors :*

1. *m_β est un polynôme à coefficients dans $\mathbb{F}_p$ ayant β comme racine dans K et dont le coefficient dominant est égale à 1.*
2. *Tout polynôme à coefficients dans $\mathbb{F}_p$ ayant β comme racine dans K est un multiple de m_β.*
3. *m_β est irréductible sur $\mathbb{F}_p$.*

Preuve. Les propositions 1 et 2 sont des conséquences de la définition de m_β.
3. Soit $m_\beta = m_1 m_2$ avec m_1 et m_2 deux polynômes de $\mathbb{F}_p[X]$. Puisque $m_1(\beta) = 0$ et puisque $F_p[X]$ est intègre, alors $m_1(\beta) = 0$ ou $m_2(\beta) = 0$. Supposons par example que $m_1(\beta) = 0$. La proposition 2. impose alors que m_1 est un multiple de m_β. On obtient donc $m_\beta = m_\beta.g.m_2$ avec $g \in \mathbb{F}_p[X]$. En considérant les degrés des polynômes, on voit que g et m_2 sont constants et que $m_1 = m_\beta$. En conclusion, tout diviseur de m_β dans $\mathbb{F}_p[X]$ est soit constant, soit égale à m_β. ■
La partie 3 de la proposition ci-dessus implique le corollaire suivant :

Corollaire 1.2.1. *L'anneau quotient $\dfrac{\mathbb{F}_p[X]}{(m_\beta)}$ est un corps.*

Sous-corps $\mathbb{F}_p(\beta)$

Soit ϕ l'application de $\mathbb{F}_p[X]$ dans K qui associe à chaque polynôme de $\mathbb{F}_p[X]$ l'image de β par ce polynôme, c'est à dire $\phi(f) = f(\beta)$. ϕ est un morphisme d'anneau et son noyau est I_β. En conséquence, $\phi(\mathbb{F}_p[X])$ est isomorphe à $\dfrac{\mathbb{F}_p[X]}{(m_\beta)}$ et donc l'image $\phi(\mathbb{F}_p[X])$ est un corps. C'est donc aussi un sous-corps de K. On notera par $F_p(\beta)$, l'image $\phi(F_p[X])$.

Proposition 1.2.6. *(Kraus 2012) Soit K un corps commutatif fini de caractéristique p. Soit $\beta \in K$ et m_β le polynôme minimal de β. Alors l'ensemble $\mathbb{F}_p(\beta) = \{f(\beta)$ tel que $f \in \mathbb{F}_p[X]\}$ est un sous-corps de K et ce sous-corps est isomorphe à $\dfrac{\mathbb{F}_p[X]}{(m_\beta)}$.*

Définition 1.2.3. *Le sous-corps $\mathbb{F}_p(\beta)$ s'appelle l'extension simple de $\mathbb{F}_p$ par β.*

Le sous-corps $\mathbb{F}_p(\beta)$ est l'ensemble des expressions polynomiales en β à coefficients dans $\mathbb{F}_p$.

Proposition 1.2.7. *1. $\mathbb{F}_p(\beta)$ contient $\mathbb{F}_p$ et β.*

2. Tout sous-corps de K qui contient $\mathbb{F}_p$ et β contient aussi $\mathbb{F}_p(\beta)$

Preuve. 1. Chaque élément u de $\mathbb{F}_p$ est l'image de β par le polynôme constant ux^0 et donc $u \in \mathbb{F}_p(\beta)$.

2. Soit L un sous-corps de K qui contient $\mathbb{F}_p$ et β. Puisque L est stable pour les deux lois de K, il contient donc aussi toutes les expressions polynomiales en β à coefficients dans $\mathbb{F}_p$. Il contient donc également $\mathbb{F}_p(\beta)$. ■

Pour rappeler la proposition précédente, on dit que $\mathbb{F}_p(\beta)$ est le plus petit corps contenant $\mathbb{F}_p$ et β. Puisque $\mathbb{F}_p(\beta)$ a pour sous-corps $\mathbb{F}_p$, c'est un espace vectoriel sur $\mathbb{F}_p$. Si sa dimension est s ; alors $|\mathbb{F}_p(\beta)| = p^s$. Par ailleurs, $\mathbb{F}_p(\beta)$ a le même cardinal que $\frac{\mathbb{F}_p[X]}{(m_\beta)}$ car il lui est isomorphe. Or on sait que $\frac{\mathbb{F}_p[X]}{(m_\beta)}$ a pour cardinal p^n où n est le degré de m_β. En conséquence $s = n$ et donc : $dim_{\mathbb{F}_p}\mathbb{F}_p(\beta) = deg(m_\beta)$.

Revenons maintenant à l'application ϕ de la définition 1.2.3. Elle se décompose de la manière suivante :

$$\begin{array}{ccccc} \mathbb{F}_p[X] & \xrightarrow{\phi} & \mathbb{F}_p(\beta) & \xrightarrow{\text{t}} & K \\ {\scriptstyle g}\downarrow & \nearrow_{h} & & & \\ \mathbb{F}_p[X]/(m_\beta) & & & & \end{array}$$

avec

$\phi(f) = f(\beta)$;

$g(f) = cl(f) = cl(r)$;

$h(cl(f)) = h(cl(r))$.

où r est le reste de f dans la division par m_β. En conséquence $\phi(f) = r(\beta)$. Le polynôme r étant soit nul, soit de degré au plus égal à $s-1$ (avec $s = deg(m_\beta)$), il est de la forme $r_0 + r_1x + \cdots + r_ix^i + \cdots + r_{s-1}x^{s-1}$.

donc pour tout $f \in F_p[X]$, $f(\beta) = r(\beta) = r_0 + r_1\beta + \cdots + r_i\beta^i + \cdots + r_{s-1}\beta^{s-1}$.

Les éléments $1, \beta, \ldots, \beta^i, \ldots, \beta^{s-1}$ forment alors un système générateur de $\mathbb{F}_p(\beta)$. La dimension de $\mathbb{F}_p(\beta)$ étant plus précisément s, on déduit que les éléments $1, \beta, \ldots, \beta^i, \ldots^{s-1}$ forment une base de$F_p(\beta)$.

Théorème 1.2.4. *(Kraus 2012) Soit K un corps commutatif fini de caractéristique p. Soit $\beta \in K$ et m_β le polynôme minimal de β. Alors :*

1. *Le sous-corps $\mathbb{F}_p(\beta)$ est un espace vectoriel sur $\mathbb{F}_p$, dont la dimension est égale au degré de m_β.*
2. *Si $s = deg(m_\beta)$, alors $1, \beta, \ldots, \beta^i, \ldots, \beta^{s-1}$ forment une base de $\mathbb{F}_p(\beta)$.*

Corollaire 1.2.2. *Tout élément de $\mathbb{F}_p(\beta)$ est d'une manière unique combinaison linéaire à coefficient dans $\mathbb{F}_p$ de $1, \beta, \ldots, \beta^i, \ldots, \beta^{s-1}$.*

Cas particulier fondamental

Considérons le cas où β est un générateur du groupe multiplicatif $K^\star$.

Définition 1.2.4. *Un générateur du groupe multiplicatif d'un corps commutatif K s'appelle une racine primitive de K ou un élément primitif.*

Remarque 1.2.3. Si α est une racine primitive de K, alors tout élément de K est de la forme $u = \alpha^i$ avec $0 \leq i \leq |K| - 2$ car le cardinal de K^* est $|K| - 1$. Si f est défini par $f(x) = x^i$, alors $u = f(\alpha)$, ainsi $u \in \mathbb{F}_p(\alpha)$. et donc $u \in \mathbb{F}_p(\alpha) = K$.

Théorème 1.2.5. *(Kraus 2012) Soit K un corps commutatif fini de caractéristique p. Si α est une racine primitive de K, alors $\mathbb{F}_p(\alpha) = K$.*

On en déduit immédiatement le corollaire suivant :

Corollaire 1.2.3. *Soit K un corps commutatif fini de caractéristique p. Soit α une racine primitive de K. Soit $m_\alpha(x)$ le polynôme minimal de α et $s = deg(m_\alpha(x))$. Alors $|K| = p^s$ et :*

1. *$\forall u \in K \backslash \{0\}$, $\exists i \in \{0, 1, \ldots, |K| - 2\}$ tel que $u = \alpha^i$.*
2. *$\forall u \in K \backslash \{0\}$, $\exists (\lambda_0, \lambda_1, \ldots, \lambda_j, \ldots, \lambda_{s-1}) \in (\mathbb{F}_p)^s$ tel que $u = \lambda_0 + \lambda_1\alpha + \cdots + \lambda_i\alpha^i + \cdots + \lambda_{s-1}\alpha^{s-1}$*

Le théorème de Wedderburn ainsi que la Propriété 2 du Corollaire 1.2.3 permettent d'énoncer le théorème suivant :

Théorème 1.2.6. *Tout corps fini est isomorphe à un corps quotient d'un anneau de polynôme sur un idéal premier.*

On va maintenant généraliser, à un corps commutatif quelconque, la définition pour le polynôme minimal d'un élément d'un corps fini. Les résultats qui suivront seront vrais dans le cas général et donc aussi dans le cas d'un corps fini.

Proposition 1.2.8. *Soit L un corps commutatif quelconque et K un sous-corps de L. Soit $\beta \in L$ et soit I_β l'ensemble des polynômes dans K ayant β comme racine dans L. Alors I_β est un idéal de $K[x]$.*

Preuve. C'est la même que pour la Proposition 1.2.4, la deuxième phrase étant remplacée par l'hypothèse que le polynôme nul appartient à I_β. ■

Définition 1.2.5. *Si I_β n'est pas réduit à zéro, le générateur de l'idéal I_β de $K[x]$ s'appelle le polynôme minimal de β (il est noté $m_\beta(x)$. Si I_β n'est pas réduit à zero, on dit que β est algébrique sur K.*

Remarque 1.2.4. Les propriétés du polynôme minimal sont les mêmes que dans la proposition 1.2.5 en remplacent $\mathbb{F}_p$ par K.

Théorème 1.2.7. *Soit F un corps commutatif quelconque. Tout polynôme unitaire non constant de $F[x]$, irréductible sur F est le polynôme minimal d'un élément d'un sur-corps de F.*

Preuve. Soit $f(x) = \sum_{i=0}^{s} f_i x^i$ un polynôme irréductible unitaire sur F et $K = \frac{\mathbb{F}}{(f(x))}$ le corps quotient correspondant. Ce corps est un sur-corps du corps F. Soit $\alpha \in K$ qui est la classe modulo $f(x)$ du polynôme $i(x)) = x$. Puisque $f(x) \equiv 0(modf(x))$, on obtient $cl(f(x)) = cl(0)$. Soit en utilisant les propriétés de morphismes de l'application "classe" (cl) : $cl(\sum_{i=0}^{s} f_i x^i) = \sum_{i=0}^{s} f_i cl(x)^i = cl(0)$ c'est à dire, $\sum_{i=0}^{s} f_i x \alpha^i = 0$. Ce ci montre que α est racine de $f(x)$ dans K. Si maintenant $g(x) = \sum_{i=0}^{s} g_i x^i)$ est un polynôme de $F[x]$ tel que $g(\alpha) = 0$ dans K, alors comme ci-dessus : $\sum_{i=0}^{s} g_i \alpha^i = \sum_{i=0}^{s} g_i cl(x)^i = cl(\sum_{i=0}^{s} g_i x^i) = 0$. Ceci implique $g(x) \equiv 0(modf(x))$ ce qui veut dire aussi que $g(x)$ est un multiple de $f(x)$. En conclusion, tout polynôme sur F qui admet α comme racine est un multiple de $f(x)$. Autrement dit, puisque $f(x)$ est unitaire, $f(x)$ est le polynôme minimal de α. ■

Définition 1.2.6. *Soit F un corps commutatif quelconque et $f(x)$ un polynôme sur F. On dit qu'un sur-corps L de F est un corps de décomposition si $f(x)$ se décompose en un produit de facteurs de degré 1 sur L.*

Théorème 1.2.8. *(Rodier 2008) Si F est un corps commutatif quelconque et f est un polynôme non constant sur F alors il existe un corps de décomposition de f sur F c'est-à-dire un corps dans lequel f peut s'écrire comme un produit de facteurs de degré 1.*

1.2.3 Existence et unicité d'un corps de cardinal p^n

Si k un corps de caractéristique p^n, alors la caractéristique de K est p, et n est le degré du polynôme minimal d'une racine primitive de K.

Théorème 1.2.9. *(Kraus 2012) Si p est entier premier et n un entier strictement positif, alors il existe un corps fini de cardinal p^n.*

Preuve. Soit $p \in \mathbb{N}$, premier et $n \in \mathbb{N}^\star$. Considérons le polynôme π défini par $\pi(x) = x^{p^n} - x$ sur $\mathbb{F}_p$. Soit K_1 un corps de décomposition de π sur $\mathbb{F}_p$. La caractéristique de K_1 est donc p.

- Le polynôme π se décompose en facteurs de degré 1 sur K_1. Le nombre de ces facteurs est égale au degré de π, soit p^n, et chacun d'eux a une racine dans K_1.
- On a la relation $\pi(x) = x\rho(x)$ avec $\rho(x) = (x^{p^n-1} - 1)$. Le polynôme dérivée de ρ est donné par $\rho'(x) = (p^n - 1)x^{p^n-2} = -x^{p^n-2}$ (car $p^n = 0$ dans $\mathbb{F}_p$). La seule racine de ρ' est 0 qui n'est pas racine de ρ. Le polynôme ρ a exactement p^n racines dans K_1. On va maintenant montrer que l'ensemble des racines de π dans K_1 est un sous-corps de K_1. Il suffit de montrer que cet ensemble K est stable pour les lois de K_1. Si $a \in K$, et $b \in K$, alors $a^{p^n} = a$ et $b^{p^n} = b$. Donc $(a+b)^{p^n} = a^{p^n} + b^{p^n} = a + b$ et $(ab)^{p^n} = a^{p^n}b^{p^n} = ab$, ce qui montre que $a + b$ et ab sont dans K. En conclusion, K est un corps dont le cardinal est p^n.

■

Corollaire 1.2.4. *Si p est un entier premier et n un entier strictement positif, alors il existe un polynôme de degré n irréductible sur $\mathbb{F}_p$.*

Théorème 1.2.10. *Deux corps finis de même cardinal sont isomorphes.*

Preuve. Soit K et L deux corps finis de même cardinal p^n. Ces deux corps ont donc pour caractéristique p. Soit α une racine primitive de K. On sait d'après le 1.2.6 que $K = \mathbb{F}_p$ et que donc K est isomorphe à $\dfrac{\mathbb{F}_p[X]}{(m_\alpha)}$, où m_α est le polynôme minimal de α sur $\mathbb{F}_p$. D'autre part, on sait que α est racine du polynôme π avec $\pi(x) = x^{p^n} - x$. Ce polynôme étant à

coefficients sur $\mathbb{F}_p$ et ayant α pour racine, est donc multiple de m_α d'après la définition d'un polynôme minimal. Or les éléments de L sont aussi racines de π. L'un au moins de ces éléments, soit γ, est donc racine de m_α dans L. Ce polynôme étant à coefficients sur $\mathbb{F}_p$ et ayant γ pour racine, est donc multiple de m_γ. Puisque m_γ est irréductible sur $\mathbb{F}_p$ et unitaire, on en déduit que $m_\gamma = m_\alpha$. Le cardinal de $\mathbb{F}_p(\gamma)$ est donc p^n puisque n est le degré de m_γ. En conséquence $L = \mathbb{F}_p(\gamma)$ et donc L est isomorphe à $\frac{\mathbb{F}_p[X]}{(m_\alpha)}$. Ceci implique que L et K sont isomorphes. ■

Corollaire 1.2.5. *Soit p un entier premier et n un entier strictement positif. Soit f un polynôme de degré n irréductible sur $\mathbb{F}_p$, alors tout corps finis de cardinal p^n est isomorphe à $\frac{\mathbb{F}_p[x]}{(f)}$.*

Remarque 1.2.5. Il n'existe pas de formule donnant un polynôme irréductible pour chaque p et n. On dispose des tables : Voici quelques polynômes primitifs sur $\mathbb{F}_2$ de degré $1 \leq n \leq 16$, qui permettent de construire les corps finis $\mathbb{F}_{2^n}$.

x^2+x+1	x^7+x^3+1	$x^{12}+x^6+x^4+x+1$
x^3+x+1	$x^8+x^4+x^3+x^2+1$	$x^{13}+x^4+x^3+x+1$
x^4+x+1	x^9+x^4+1	$x^{14}+x^{10}+x^6+x+1$
x^5+x^2+1	$x^{10}+x^3+1$	$x^{15}+x+1$
x^6+x+1	$x^{11}+x^2+1$	$x^{16}+x^{12}+x^3+x+1$

Exemple 1.2.3. Construction du corps $\mathbb{F}_9$ au moyen d'une racine primitive.

Puisque $9 = 3^2$ alors on a besoin d'un polynôme unitaire de degré 2, irréductible sur $\mathbb{F}_3$. Soit le polynôme f défini sur $\mathbb{F}_3$ par $f(x) = x^2 + x + 2$, on a $f(0) = 2$, $f(1) = 1$ et $f(2) = 2$ (calcul dans $\mathbb{F}_3$). On voit que f n'a pas de racine dans $\mathbb{F}_3$ et donc pas de diviseur de degré 1 sur $\mathbb{F}_3$. Il est donc irréductible sur $\mathbb{F}_3$. Si α est une racine primitive dont f est le polynôme minimal, alors $f(\alpha) = 0$ c'est à dire $\alpha^2 + \alpha + 2 = 0$, donc $\alpha^2 = -\alpha - 2 = 2\alpha + 1$ (car dans $\mathbb{F}_3$, $-1 = 2$). On en déduit :

$$\begin{aligned}
\alpha^3 &= 2\alpha^2 + \alpha &&= 2(\alpha+1) + \alpha &&= 2\alpha + 2\\
\alpha^4 &= 2\alpha^2 + 2\alpha &&= 2(2\alpha+1) + 2\alpha &&= 2\\
\alpha^5 &= 2\alpha\\
\alpha^6 &= 2\alpha^2 &&= 2(2\alpha+1) &&= \alpha + 2\\
\alpha^7 &= \alpha^2 + 2\alpha &&= (2\alpha+1) + 2\alpha &&= \alpha + 1
\end{aligned}$$

Finalement $\mathbb{F}_9 = \{0, 1, \alpha, \alpha^2, \cdots, \alpha^7\}$ avec :

$$(1)\begin{cases} \alpha^2 &= 2\alpha+1 \\ \alpha^3 &= 2\alpha+2 \\ \alpha^4 &= 2 \\ \alpha^5 &= 2\alpha \\ \alpha^6 &= \alpha+2 \\ \alpha^7 &= \alpha+1 \end{cases}$$

et

(2) $\alpha^8 = 1$.

Les règles (1) et (2) permettent d'effectuer tous les calculs dans $\mathbb{F}_9$. Par exemple : $\alpha^3+\alpha^5 = (2\alpha+2)+2\alpha = \alpha+2 = \alpha^6$ et $\alpha^3\alpha^5 = \alpha^9 = \alpha$. Dans cet exemple et à chaque fois que $p = 3$, on peut remplacer 2 par -1, ce qui facilite les calculs.

Chapitre Deux

GÉNÉRALITÉS SUR LA THÉORIE ALGÉBRIQUE DU CODAGE

2.1 Introduction

Les codes correcteurs d'erreurs ont été développés dans la deuxième moitié du vingtième siècle, suite aux travaux de Shannon (Shannon 1949). L'objectif est alors de pouvoir établir des communications claires (sans parasites, sans brouillage).Plutôt que d'essayer sans cesse d'améliorer physiquement les systèmes de transmission, Shannon a l'idée d'une autre approche : Faire subir au signal un traitement informatique après sa réception afin de détecter et de corriger les erreurs de transmission. C'est le début de la digitalisation de l'information.Le principe est d'ajouter de la redondance dans le message transmis. On obtient ainsi un message plus long, mais dont l'information excédentaire peut être exploitée pour détecter voire corriger des erreurs de transmission.

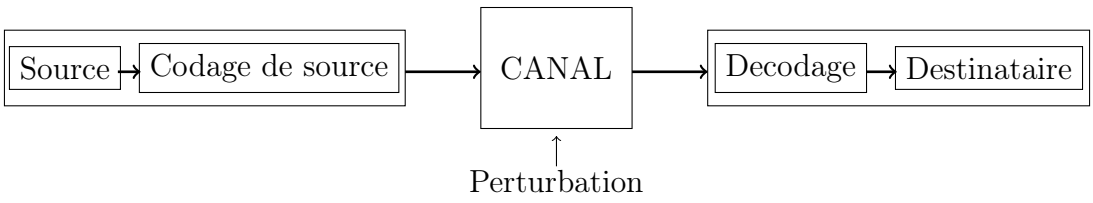

FIGURE 2.1 – Schéma illustratif de transmission.

Les codes correcteurs sont utilisés ainsi dans toutes les télécommunications, mais aussi dans le stockage de données (gravure de CD-rom, par exemple) ou encore dans les identifiants personnels (de comptes bancaires).

2.2 Définitions

Soit $\mathbb{F}_q$ le corps finis à q éléments. Soit n et k deux entiers tels que $k \leq n$. On définit sur $\mathbb{F}_q^n$ l'ensemble des mots de la forme $\underline{x} = (x_1, \cdots, x_n)$.

Notation 2.2.1. Nous noterons $\lfloor t \rfloor$ la partie entière de t, $Card(S)$ le cardinal de l'ensemble S et $[|1, n|]$, l'ensemble des entiers compris entre 1 et n.

Définition 2.2.1. *La distance de Hamming entre deux mots $\underline{x}, \underline{y} \in \mathbb{F}_q^n$ est le nombre de positions sur lesquelles les deux mots diffèrent. On le note d_H et on écrit :*

$d_H(\underline{x}, \underline{y}) = Card(\{i = 1, \cdots, n \ | x_i \neq y_i\})$

Exemple 2.2.1. La distance de Hamming entre les mots (1101001) et (1001011) de $\mathbb{F}_2^7$ est 2.

Définition 2.2.2. *Le poids de Hamming d'un mot $\underline{x} \in \mathbb{F}_q^n$ est simplement sa distance au mot nul. On le note W_H et on écrit : $W_H(\underline{x}) = Card(\{i = 1, \cdots, n \ | x_i \neq 0\})$*

Exemple 2.2.2. Le poids de Hamming du mot (1101001) sur $\mathbb{F}_2^7$ est 4.

Définition 2.2.3. *Un code linéaire $[n, k]$ sur l'alphabet $\mathbb{F}_q$ est un sous-espace vectoriel C de $\mathbb{F}_q^n$ de dimension k sur $\mathbb{F}_q$. Son taux de transmission est $R = \frac{k}{n}$.*
C est dit q-aire si $C = Im\varphi$ pour une application linéaire injective $\varphi : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n$.
L'élément $\varphi(\underline{u})$, pour un mot d'information $\underline{u}$ de $\mathbb{F}_q^k$ est appelé un mot code. Et on a $Card(C) = q^k$.

Définition 2.2.4. *La distance minimale notée d du code C est simplement le minimum de la distance entre deux mots de code différents. Sa capacité de correction théorique t et son taux de correction δ sont définis à partir de cette distance minimale.*
$d = min\{d_H(\underline{x}, \underline{y}),\ avec\ \underline{x}, \underline{y} \in C,\ \underline{x} \neq \underline{y}\}$, $t = \lfloor \frac{d-1}{2} \rfloor$ et $\delta = \frac{d}{n}$.

Si C est un code linéaire $[n, k]$ de distance minimale d, on le note $C[n, k, d]$

Remarque 2.2.1. La distance minimale d'un code linéaire est également son poids minimum. $d = min\{d_H(\underline{x}, 0)\ avec\ \underline{x} \in C,\ \underline{x} \neq 0\} = W_H(\underline{x})$.

Définition 2.2.5. *Soient C un code linéaire de capacité de correction théorique t et soit $\underline{x} \in \mathbb{F}_q^n$ un mot quelconque. On dit que C est t-correcteur d'erreurs s'il existe au plus un mot $\underline{c}$ de C tel que $d_H(\underline{x}, \underline{c}) \leq t$, c'est à dire :*

$\forall \underline{x} \in \mathbb{F}_q^n\ Card(\{\underline{y} \in C / d_H(\underline{x}, \underline{y}) \leq t\}) \leq 1$.

Théorème 2.2.1. *Si la distance minimale d d'un code linéaire C est telle que $d \geq 2t+1$ alors C peut corriger t erreurs.*

Preuve. Si $\underline{c}$ est le mot envoyé et $\underline{y}$ le mot reçu tels que $d_H(\underline{y}, \underline{c}) \leq t$ alors tout mot code $\underline{c'}$ de C est tel que $d_H(\underline{c}, \underline{c'}) \geq 2t+1$. Or, d_H estune distance, donc $d_H(\underline{y}, \underline{c'}) \geq d_H(\underline{c}, \underline{c'}) - d_H(\underline{c}, \underline{y})$. Par conséquent $d_H(\underline{y}, \underline{c'}) \geq t+1$. C peut donc corriger t erreurs ■

Exemple 2.2.3.

$$\begin{array}{rccc} \varphi : & \mathbb{F}_2^4 & \longrightarrow & \mathbb{F}_2^7 \\ & (a_1, a_2, a_3, a_4) & \longmapsto & (a_1, a_2, a_3, a_4, a_1+a_3+a_4, a_1+a_2+a_4, a_1+a_2+a_3) \end{array}$$

Soient $\underline{a}, \underline{b}$ des éléments de $\mathbb{F}_2^4$
Si $d(\underline{a}, \underline{b}) = 1$, alors $d(\varphi(a), \varphi(\underline{b})) = 3$ ou 4
si $d(\underline{a}, \underline{b}) = 2$, alors $d(\varphi(\underline{a}), \varphi(\underline{b})) = 3$ ou 4
Si $d(\underline{a}, \underline{b}) = 3$, alors $d(\varphi(\underline{a}), \varphi(\underline{b})) = 3$ ou 4
Si $d(\underline{a}, \underline{b}) = 4$, alors $d(\varphi(\underline{a}), \varphi(\underline{b})) = 7$.
Ceci dit, l'application φ écarte vraiment les mots d'information. En effet, on peut toujours supposer $b = (0,0,0,0)$, et on utilise directement la définition de φ pour le mot $\varphi(\underline{a})$. Donc $d = 3$, et le code corrige 1 erreur.

Remarque 2.2.2. Un des objectifs de la théorie du codage consiste à élaborer des codes dont les mots sont très éloignés les uns des autres au sens de la distance de Hamming. Un autre est de transmettre le maximum d'information et donc de garder des vitesses de transmission acceptables.

Définition 2.2.6. *Un bon code est un code corrigeant de nombreuses erreurs compte tenu de sa longueur tout en ayant une vitesse de transmission la plus élevée possible, ce qui correspond à des paramètres R et δ assez grands dans $[0,1]$.*

Définition 2.2.7. *(Matrice génératrice)*

On considère un code linéaire $C[n,k,d]$ sur $\mathbb{F}_q$. On appelle matrice génératrice du code C une matrice G de taille $k \times n$ sur $\mathbb{F}_q$ dont les lignes forment une base de C.

Remarque 2.2.3.
– Puisqu'un espace vectoriel admet plusieurs bases, une matrice génératrice d'un code linéaire n'est pas unique, par contre une matrice génératrice d'un code $C[n,k]$ est toujours une matrice $k \times n$ de rang k.

– Une matrice génératrice est suffisante pour définir un code linéaire associé. En effet tous les mots de code peuvent être générés grâce à cette matrice.
En effet pour tout $\underline{u} \in \mathbb{F}_q^k$, $\underline{u}.G = \varphi(\underline{u}) \in C \subset \mathbb{F}_q^n$.

Définition 2.2.8. *Un code linéaire est dit code systématique si les k premières colonnes de sa matrice génératrice forment matrice identité. Une telle matrice G dite matrice systématique prend alors la forme suivante :* $\begin{pmatrix} 1 & 0 & \cdots & 0 & c_{1k+1} & \cdots & c_{1n} \\ 0 & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 & c_{k-1k+1} & \cdots & c_{k-1n} \\ 0 & \cdots & 0 & 1 & c_{kk+1} & \cdots & c_{kn} \end{pmatrix} = (I_k | - A^t)$
où A^t est la transposée de $A \in M_{n-k,k}(\mathbb{F}_q)$.
Pour un mot d'information $\underline{u}$,

$$\underline{u}.G = \begin{pmatrix} u_1, \cdots, u_k \end{pmatrix} \begin{pmatrix} 1 & 0 & \cdots & 0 & c_{1k+1} & \cdots & c_{1n} \\ 0 & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 & c_{k-1k+1} & \cdots & c_{k-1n} \\ 0 & \cdots & 0 & 1 & c_{kk+1} & \cdots & c_{kn} \end{pmatrix} = \begin{pmatrix} u_1, \cdots, u_k, b_1, \cdots, b_{n-k} \end{pmatrix}$$

Remarque 2.2.4. Cette forme est particulièrement intéressante, le calcul d'un mot de code consiste à la détermination des $n-k$ dernières coordonnées, car les k premières correspondent à celles du message initial.

Lemme 2.2.1. *La plus grande famille de vecteurs de la base canonique engendrant un sous-espace vectoriel d'intersection réduite au vecteur nul avec le code est de cardinal $n-k$.*

Preuve. En effet, considérons une base du code, elle forme une famille libre. Le théorème de la base incomplète indique qu'il est possible de compléter cette base par des éléments d'une famille génératrice quelconque, par exemple la base canonique de $\mathbb{F}_q^n$. Les vecteurs complétant la famille sont au nombre de $n-k$ car le code est de dimension k et $\mathbb{F}_q^n$ de dimension n. Ces vecteurs forment une famille répondant aux hypothèses de la proposition. Toute famille de cardinal supérieur possède une codimension inférieure à la dimension du code et donc l'intersection de l'espace généré par une telle famille et le code est non réduit au vecteur nul. ■

Théorème 2.2.2. *Tout code linéaire est équivalent à un code systématique.*

Preuve. Remarquons tout d'abord qu'il est peut être nécessaire de modifier l'ordre de la base canonique de $\mathbb{F}_q^n$. En effet, si par exemple, le code est inclus dans l'espace vectoriel

engendré par les $n-1$ derniers vecteurs de la base canonique, alors il est vain de chercher une forme systématique sans modifier l'ordre de la base. Soit $(\underline{f}_j)$ où $j \in [|1,n|]$ la base canonique de $\mathbb{F}_q^n$. Quitte à la réordonner, supposons que les $n-k$ derniers vecteurs engendrent un sous-espace d'intersection réduite au vecteur nul avec le code. L'objectif est de trouver une base de $\mathbb{F}_q^k$, $(\underline{e}_i)$ où $i \in [|1,k|]$ tel que :

$\forall i \in [|1,k|], \exists (c_{ji})_{j\in[|k+1,n|]} \subset \mathbb{F}_q^{n-k}$ / $\varphi(\underline{e}_i) = \underline{f}_i + \sum_{i=k+1}^{n} c_{ji}.\underline{f}_j$. $(*)$

Soit V_i le sous-espace vectoriel de $\mathbb{F}_q^n$ engendré par les vecteurs $\underline{f}_j$ où $j \in \{i\} \cap [|k+1,n|]$, où $i \in [|1,n|]$. V_i contient une intersection non vide avec le code C. Un élément non nul de cette intersection possède une coordonnée sur $\underline{f}_i$ non nulle car les $n-k$ derniers vecteurs de la base engendre un sous-espace vectoriel d'intersection réduite au vecteur nul avec le code. Choisissons l'élément de l'intersection ayant une coordonnée égal à 1 sur le vecteur $\underline{f}_i$ et notons $\underline{e}_i$ son antécédent par φ. Les k égalités décrites en $(*)$ sont vérifiées. Il ne reste plus qu'à montrer que la famille $(\underline{e}_i)_{i\in[|1,k|]}$ est libre. Supposons qu'il existe $\lambda_1, \cdots, \lambda_k \in \mathbb{F}_q$ tels que : $\sum_{i=1}^{k} \lambda_i.\underline{e}_i = 0$. L'image par φ de cette relation de dépendance linéaire est encore nulle donc : $\varphi\left(\sum_{i=1}^{k} \lambda_i \underline{e}_i\right) = \sum_{i=1}^{k} \lambda_i \varphi(\underline{e}_i) = \sum_{i=1}^{k} \lambda_i \underline{f}_i + \sum_{j=k+1}^{n} \left(\sum_{i=1}^{k} c_{ji}\lambda_i\right) \underline{f}_j = 0.$

Or $\left(\underline{f}_j\right)_{j\in[|1,n|]}$ est une base, la relation de dépendance linéaire sur cette base est triviale et $\lambda_1 = \cdots = \lambda_k = 0$. La famille $(\underline{e}_i)_{i\in[|1,k|]}$ est donc libre, c'est une base car son cardinal est celui de la dimension de $\mathbb{F}_q^k$, ce qui termine la démonstration. ■

Remarque 2.2.5. On peut ainsi caractériser les codes linéaires à partir de matrices à coefficients dans $\mathbb{F}_q$ comme noyau d'une autre application linéaire $S : \mathbb{F}_q^n \longrightarrow \mathbb{F}_q^{n-k}$.

Exemple 2.2.4. Soit H une matrice $(n-k) \times n$ à coefficients dans $\mathbb{F}_q$ de rang $n-k$. Le noyau de l'application représentée par H est un sous-espace vectoriel de $\mathbb{F}_q^n$. On peut donc définir un code linéaire par un système d'équations linéaires :

$$C = \left\{\underline{c} = (c_1, \cdots, c_n) \in \mathbb{F}_q^n / H\underline{c}^t = 0\right\}$$

En effet, posons $H = \left(A, I_3\right) = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$. Ici $q = 2$, $n = 7$ et $k = 4$ (C est un code binaire).

On désire transmettre $\underline{a} = (a_1a_2a_3a_4)$. On le code en $\underline{c} = (a_1a_2a_3a_4c_5c_6c_7)$, avec c_5, c_6, c_7 tels que $H\underline{c}^t = 0$.

$$\text{Or } H\underline{c}^t = 0 \iff \begin{array}{l} a_1 + a_3 + a_4 + c_5 = 0 \\ a_1 + a_2 + a_4 + c_6 = 0 \\ a_1 + a_2 + a_3 + c_7 = 0 \end{array} \iff \begin{array}{l} c_5 = a_1 + a_3 + a_4 \\ c_6 = a_1 + a_2 + a_4 \\ c_7 = a_1 + a_2 + a_3 \end{array}$$

On obtient l'application linéaire injective

$$\begin{array}{rccc} \varphi : & \mathbb{F}_2^4 & \longrightarrow & \mathbb{F}_2^7 \\ & (a_1, a_2, a_3, a_4) & \longmapsto & (a_1, a_2, a_3, a_4, a_1 + a_3 + a_4, a_1 + a_2 + a_4, a_1 + a_2 + a_3) \end{array}$$

Définition 2.2.9. *Soit $C[n,k]$ un code linéaire sur $\mathbb{F}_q$, on appelle matrice de contrôle ou de parité de C une matrice $n-k \times n$ sur $\mathbb{F}_q$ de rang $n-k$ telle que : $\forall \underline{y} \in \mathbb{F}_q^n$, $H\underline{y}^t = 0 \iff \underline{y} \in C$*

Définition 2.2.10. *$G = (I_k | -A^t)$ est la matrice génératrice systématique du code linéaire C de matrice de contrôle $H = (A|I_{n-k})$*

Remarque 2.2.6. Pour tout mot code $\underline{y}$, on a $H\underline{y}^t = 0$ et $\underline{y} = \underline{a}G$. Donc

$GH^t = 0_{M_{k,n-k}(\mathbb{F}_q)}$ et $HG^t = 0_{M_{n-k,k}(\mathbb{F}_q)}$, puisque $H\underline{y}^t = HG^t\underline{a}^t = 0 \quad \forall \underline{a} \in \mathbb{F}_q^k$

Définition 2.2.11. *Soit H une matrice de contrôle d'un code linéaire $C[n,k,d]$ et $\underline{y} \in \mathbb{F}_q^n$. Le vecteur $S(\underline{y}) = H\underline{y}^t$ de longueur $n-k$ est appelé le syndrome de $\underline{y}$.*

2.3 Résolution algorithmique du décodage

Nous nous intéressons ici à la possibilité ou non de mettre en place des algorithmes de décodage d'un code linéaire, et la complexité de tels algorithmes. On considère donc un code linéaire $C[n,k,d]$ sur $\mathbb{F}_q$, un mot $\underline{y} \in \mathbb{F}_q^n$ et un entier t. Voici les formalisations classiques des problèmes de décodage (Faure 2009) les plus courants :

- Décodage au maximum de vraisemblance $(C, \underline{y})$

  Il consiste à trouver un mot $\underline{c} \in C$ vérifiant : $d_H(\underline{c}, \underline{y}) = d_H(C, \underline{y})$

  avec $d_H(C, \underline{y}) = min\{d_H(\underline{x}, \underline{y}),\ \underline{x} \in C\}$.
- Décodage borné $(C, \underline{y}, t)$

  Il consiste à trouver, s'il existe, un mot $\underline{c} \in C$ vérifiant $d_H(\underline{c}, \underline{y}) \leq t$.
- Décodage en liste $(C, \underline{y}, t)$

  Il consiste à trouver l'ensemble des mots $\underline{c} \in C$ vérifiant $d_H(\underline{c}, \underline{y}) \leq t$.

Définition 2.3.1. *Résoudre le décodage au maximum de vraisemblance revient, à partir d'un mot reçu $\underline{y}$ pouvant contenir des erreurs, à retrouver le mot de code $\underline{x} \in C$ le plus proche au sens de Hamming du mot $\underline{y}$, c'est-à-dire le mot qui a été le plus probablement envoyé par l'expéditeur. Le décodage en liste revient à établir un ensemble de candidats réalistes pour le mot envoyé.*

Remarque 2.3.1. Le problème de décodage borné est plus simple que chacun des deux autres. Cependant, si $t < \frac{d}{2}$, alors Décodage borné$(C, \underline{y}, t)$ et Décodage en liste $(C, \underline{y}, t)$ sont équivalents. On ne s'intéresse donc au décodage en liste que pour des distances excédant la capacité de correction du code. Deux algorithmes permettent de résoudre de façon générale le problème de décodage d'un code linéaire.

1. Décodage par Syndrome

Notons que pour encoder un message m on le multiplie par une matrice génératrice G du code C. On obtient un mot de code $\underline{c}$. Après passage par un canal bruité, une erreur $\underline{e}$ s'ajoute au mot de code $\underline{c}$ on obtient $\underline{y} = \underline{c} + \underline{e}$. Soit H la matrice de parité du code C. on a $H\underline{y}^t = H\underline{c}^t + H\underline{e}^t = H\underline{e}^t$. Par conséquent, si $\underline{c}$ est solution du problème de décodage borné, on peut calculer $\underline{e}$ à l'aide d'un mot de poids faible de même syndrome que $\underline{y}$.

Algorithme 1 : Décodage par syndrome
Entrée : G, $\underline{y} = (y_1, \cdots, y_n)$, t.
Sortie : $\underline{c} \in C$ tel que $d_H(\underline{c}, \underline{y}) \leq t$
Précalculs : Calculer une matrice de parité H associée à la matrice génératrice G Pour chaque mot $\underline{e} \in \mathbb{F}_q^n$ de poids inférieur ou egal à t, calculer $H\underline{e}^t$ et l'ajouter à la liste des syndromes de L
Décodage : Calculer $H\underline{y}^t$ et rechercher dans la liste L un mot $\underline{e}$ admettant le même syndrome. Si $\underline{e}$ existe, renvoyer $\underline{c} = \underline{y} - \underline{e}$. Sinon, renvoyer "Pas de solution".

Remarque 2.3.2. Cet algorithme doit cependant calculer et manipuler une liste L contenant les syndromes de tous les mots de poids de Hamming inférieur à t. Il existe au moins $C_n^t(q-1)^t$ mots d'un tel poids, la complexité de l'algorithme est donc exponentielle.

Exemple 2.3.1. Soit C un code binaire de matrice génératrice $G = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}$ de capacité de correction $t = 1$. On reçoit le mot $\underline{y} = (1110)$.

Après calcul, on obtient la matrice de contrôle $H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$.

Mots d'information	{	00	10	01	11	}
Mots de codes	{	0000	1010	0111	1101	ligne1
		1000	0010	1111	0101	ligne2
		0100	**1110**	0011	1001	ligne3
		$\underbrace{0001}$	1011	0110	1100	} ligne4
		Leaders de classe				

Les mots de codes des lignes 1, 2, 3 et 4 ont respectivement pour syndromes : $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$; $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$; $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$; $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$. Cependant le mot $\underline{y}$ a pour syndrome : $S(\underline{y}) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$. Il vient ensuite que le vecteur erreur est le leader de la classe correspondante ayant le même syndrome, donc $\underline{e} = (0100)$, et on décode $\underline{y}$ en $\underline{c} = \underline{y} - \underline{e} = (1010)$.

Remarque 2.3.3. Cette méthode est toute fois limitée, car pour de très grands codes il devient impossible de trouver des leaders de classe. Un code binaire de longueur 50 et de dimension 20 possède environ 10^9 classes.

2. Décodage par ensemble d'information

Définition 2.3.2. *(Ensemble d'information) Soit C un code linéaire $[n, k]$ sur $\mathbb{F}_q$, et G une matrice génératrice de ce code. Un ensemble d'indice $I \subset [|1, n|]$ est appelé ensemble d'information pour le code C si I est de cardinal k, et que la restriction de G aux colonnes d'indices $i \in I$ forme une matrice inversible.*

Proposition 2.3.1. *(Faure 2009) Si $[|1, k|]$ est un ensemble d'information du code linéaire C, alors il existe une unique matrice génératrice de C de la forme $(I_k|R)$ et une unique matrice de contrôle de C de la forme $(Q|I_{n-k})$où I_k est la matrice identité d'ordre k. De plus ces matrices vérifient $Q = -R^t$.*

Remarque 2.3.4. On peut calculer ces deux matrices à partir de G en temps $\mathcal{O}(n^3)$ par un pivot de Gauss.

Proposition 2.3.2. *(Faure 2009) Si $I = i_1, \cdots, i_k$ est un ensemble d'information du code linéaire C, et $(x_1, \cdots, x_k)$ un vecteur quelconque de $\mathbb{F}_q^k$, alors il existe un unique mot de code $\underline{c}$ tel que pour tout $j \leq k$, $c_{ij} = x_j$.*

Algorithme 2 : Décodage par ensemble d'information
Entrée : G, $\underline{y} = (y_1, \cdots, y_n)$, t.
Sortie : $\underline{c} \in C$ tel que $d_H(\underline{c}, \underline{y}) \leq t$
Interpolation : Choisir un sous-ensemble $I \subset [\|1, n\|]$ de cardinal k. Construire l'unique mot de code $\underline{c} \in C$ tel que pour tout $i \in I$, $c_i = y_i$. Si $d_H(\underline{c}, \underline{y}) \leq t$, renvoyer c. Sinon, recommencer avec un nouveau I.

Remarque 2.3.5. (Canteaut et Chabaud 1998) Si l'on a $\underline{y} = \underline{c} + \underline{e}$, avec $c \in C$ et $W_H(\underline{e}) = t$, alors l'interpolation de $\underline{y}$ sur I produira le mot de code $\underline{c}$ si et seulement si $\underline{e}$ est nul sur tous les éléments de I. Cela se produit avec une probabilité $\frac{C_{n-k}^t}{C_n^t} = \frac{C_{n-t}^k}{C_n^k}$.Chaque étape d'interpolation coûte grossièrement $\mathcal{O}(n^3)$ opérations. Donc,si les ensembles d'informations sont choisis au hasard, la complexité moyenne de l'algorithme vaut $\mathcal{O}\left(n^3 \frac{C_n^t}{C_{n-k}^t}\right)$. Cet algorithme est donc exponentiel, mais bien plus efficace que le Décodage par Syndrome.

2.4 Quelques problèmes difficiles liés aux codes et utilisés en cryptographie

Pour mettre au point un cryptosystème, il faut un problème algorithmique difficile. La sécurité d'un système doit reposer non pas sur la connaissance d'une clé (partagée secrètement par les utilisateurs) mais sur la difficulté d'inverser une fonction à sens unique avec trappe. Une fonction à sens unique est simplement une fonction difficile à inverser par calcul, une trappe est un algorithme secret rendant facile cette inversion. Ainsi la trappe n'est connue que d'une personne, seule à pouvoir déchiffrer les messages créés en utilisant la fonction à sens unique qui est elle publique. Voici quelques problèmes liés aux codes qu'aucun algorithme actuellement connu ne permet d'attaquer. (Cayrel 2008).

1. Problème de construction des codes

On peut encore énoncer le problème de la façon suivante :

Données : Le corps $\mathbb{F}_q$ et des paramètres n, k, d.

Problème : Construire un code $[n, k, d]$ sur$\mathbb{F}_q$.

De façon générale ce problème est difficile et n'a pas de solution polynomiale connue.

Toutefois, pour certains jeux de paramètres on a des solutions constructibles en temps polynomial, de même pour d'autres jeux, on sait qu'il n'existe pas de tels codes. La borne de Gilbert-Varshamov permet de déterminer des paramètres pour lesquels on est sûr qu'un tel code linéaire existe, mais elle n'est pas constructive. Elle s'énonce ainsi :

Théorème 2.4.1. ***Borne de Gilbert-Varshamov***

Si $\sum_{i=0}^{d}(C_{n-1}^{i}(q-1)^{i} < q(n-k)$ *alors il existe un code* $[n,k,d]$ *sur* $\mathbb{F}_q$.

Remarque 2.4.1. Il est aussi à noter qu'en général, vérifier que le code a bien la distance minimale que l'on annonce est aussi un problème NP-dur. (Cayrel 2008).

2. Problème de calcul des paramètres

En général, les paramètres d'un code linéaire sont sa longueur, sa dimension et sa distance minimale. Cela suffit en général à avoir une bonne idée des caractéristiques de ce code. Toutes ces grandeurs peuvent être déterminées à l'aide du polynôme énumérateur des poids du code et calculer ce polynôme nécessite d'énumérer tous les mots du code et de regarder leur poids.

a) Distance minimale

La distance minimale est l'une des données à laquelle on s'intéresse le plus pour un code. En effet, en fonction de sa longueur et de sa dimension, la valeur de la distance minimale du code va permettre de savoir si c'est un bon code. On cherche par exemple à construire des codes qui atteignent la borne de Gilbert-Varshamov. Cependant, calculer cette grandeur pour un code aléatoire est un problème NP-dur (Finiasz 2004). Pour les codes de Goppa, il est assez difficile de construire des mots de poids minimum, et il n'existe encore aucune preuve que de tels mots existent toujours (Finiasz 2004) : il est possible que certains codes de Goppa aient une distance minimale supérieure à leur distance construite.

b) Nombre de mots de poids w

De façon générale, calculer le nombre de mots de poids w d'un code est difficile. On sait qu'il y a toujours un mot de poids 0 (car on regarde des codes linéaires), et on sait que jusqu'à la distance minimale (et donc au moins jusqu'à la distance construite aussi) il n'y en a pas. Au-delà on ne sait pas du tout combien il y en a. L'ensemble des nombres de mots de poids w d'un code est ce que l'on appelle sa distribution des poids. Pour un code

aléatoire binaire, cette distribution des poids est en moyenne une distribution binomiale, c'est-à-dire que le nombre de mots de poids w est en moyenne : $N_w = \frac{C_n^w}{2^{n-k}}$. On peut grâce à cela déterminer la valeur moyenne de la distance minimale d'un code aléatoire qui est, approximativement, la plus petite valeur de w pour laquelle $N_w \geq 0.5$, et donc en déduire qu'un code aléatoire est en moyenne proche de la borne de Gilbert-Varshamov (Finiasz 2004).

c) Énumérateur des poids

Le polynôme énumérateur des poids d'un code C est le polynôme homogène défini par la formule suivante : $W_c(x,y) = \sum_{v \in C} x^{n-poids(v)} y^{poids(v)} = \sum_{i=0}^{n} N_i x^{n-i} y^i$. Avec N_i le nombre de mots de poids i dans C. Calculer tous les coefficients de ce polynôme nécessite de regarder les poids de tous les mots du code C et est donc quelque chose de très coûteux. On sait à moindre coût, évaluer ce polynôme en certains points particuliers. On le voit sur les exemples suivants :

- $W_c(1,1) = 2^k$, le nombre total de mots du code.
- $W_c(1,-1) = 0$ si le code contient des mots de poids impair, 2^k autrement.

En revanche, le nombre de tels points est fini et ces points sont tous bien connus (Finiasz 2004). Évaluer W_c en un autre point est un problème NP-dur, comme le calcul du polynôme en entier (Finiasz 2004). Ce polynôme est relativement important en théorie des codes car il contient presque toutes les informations concernant le code.

3. Problème de décodage par syndrome

Données :Une matrice H $(n-k \times n)$, un entier w, un syndrome $s \in \mathbb{F}_2^{n-k}$

Problème : Trouver un mot $\underline{e} \in \mathbb{F}_2^{n-k}$ tel que $w_H(\underline{e}) \leq w$ et $H(\underline{e})^t = s$.

Ce problème désigne la difficulté de retrouver des leaders de classe parmi des mots de longueur n (n assez grand) ainsi que la difficulté de stocker et manipuler une longue liste de syndromes de ces mots. Une variation de ce problème appelée problème de distance minimale (Minimum Distance) a aussi été prouvé NP-complet dans (Vardy 1997).

4. Problème de distinction d'un code de Goppa avec un code aléatoire

Données : Une matrice $H \in M_{n-k,n}(\mathbb{F}_q)$ binaire (la matrice de parité d'un code de Goppa(n,k)) ou une matrice binaire aléatoire).

Problème : Répondre $b = 1$ si H est un code de Goppa (n, k), et $b = 0$ sinon.
Ce que l'on appelle problème de distinction des codes de Goppa binaires désigne simplement le fait qu'il est difficile de parvenir à distinguer un code de Goppa dont on ne connaît ni le support, ni le générateur d'un code aléatoire de même longueur et même dimension. Il s'agit d'un problème NP-dur (Cayrel 2008).

Remarque 2.4.2.
- La sécurité de presque tous les cryptosystèmes liés aux codes connus à ce jour, ainsi que celle du cryptosystème Mc Eliece repose en majeure partie sur le problème de décodage par syndrome. La raison de ce choix vient du fait que ce problème est à la fois d'utilisation pratique et d'une sécurité remarquable.(même les meilleurs algorithmes sont très coûteux, et il est rarement possible de les améliorer).
- Par ailleurs, même si ce problème est pratique du point de vue de la sécurité, il a quand même quelques inconvénients : la manipulation d'une matrice de taille importante va faire que tous les systèmes auront besoin, à un moment ou à un autre, de manipuler des objets de grande taille. Ceci fait que dans les systèmes à clé publique utilisant des codes, la taille de la clé sera toujours assez importante et, de façon générale, de tels systèmes ne seront jamais très adaptés à une implantation sur des architectures limitées en mémoire (les cartes à puce par exemple).

2.5 Les codes de Goppa rationnels

Introduits par Valeri Goppa en 1970, les codes de Goppa étaient appréciés en théorie de codes pour leurs algorithmes de décodage efficaces (Massey 1969). Ils sont maintenant très utilisés en cryptographie, en particulier dans le chiffrement Mc Eliece.
Soit $g(x)$ un polynôme unitaire irréductible sur $\mathbb{F}_{q^m}$ où m est un entier naturel,
$L = \{\gamma_0, \gamma_1, \ldots, \gamma_{n-1}\} \subset \mathbb{F}_{q^m}$ avec $g(\gamma_i) \neq 0$ pour tout $i = 0, 1, 2, \ldots, n-1$.

Définition 2.5.1. *Le code de Goppa rationnel $\Gamma(L, g)$ est*

$$\Gamma(L, g) = \left\{ w = (w_0, w_1, w_2, \ldots, w_{n-1}) \subset \mathbb{F}_q^n | \sum_{i=0}^{n-1} \frac{w_i}{x - \gamma_i} \equiv 0 \ mod \ g(x) \right\}.$$

C'est-a-dire, la fonction à droite est de la forme d'une fonction irréductible $\dfrac{a(x)}{b(x)}$ avec $g(x)$ qui divise $a(x)$.

Remarque 2.5.1. la distance minimale du code notée $d(\Gamma(L,g))$ est telle que : $d(\Gamma(L,g)) \geq t+1$, $t = deg(g)$ puisque $\sum_{i=0}^{n-1} \frac{w_i}{x-\gamma_i} = \frac{a(x)}{b(x)} \neq 0$ avec $g(x)$ divisant $a(x)$ donc il y a au moins $s \geq t+1$ termes w_i non-nuls car après avoir amener au dénominateur commun on aura le degré du polynôme $a(x) \leq s-1$.

Soient $\underline{v} = (v_0, v_1, \ldots, v_{n-1}) = \underline{w} + \underline{e}$, un mot reçu, $\underline{w} = (w_0, w_1, \ldots, w_{n-1}) \in \Gamma(L,g)$ un mot de code, $\underline{e} = (e_0, e_1, \ldots, e_{n-1}) \in \mathbb{F}_q^n$ le mot d'erreur, alors on considère les fractions suivantes :

$$v(x) = \sum_{i=0}^{n-1} \frac{v_i}{x-\gamma_i}, \quad w(x) = \sum_{i=0}^{n-1} \frac{w_i}{x-\gamma_i}, \quad e(x) = \sum_{i=0}^{n-1} \frac{e_i}{x-\gamma_i}.$$

Construction de la matrice de contrôle(Pantchichkine 2004)

On veut écrire une matrice de contrôle $H \in Mat_{mt,n}(\mathbb{F}_q)$ du code $\Gamma(L,g)$. On utilise pour tout $i = 0, \ldots, n-1$ un unique polynôme $f_i(x) mod\ g(x)$ dans $\mathbb{F}_{q^m}[x]/(g)$ dont l'existence et l'unicité sont prouvés dans (Hoffmann 2011) tel que

$$f_i(x)(x-\gamma_i) \equiv 1\ mod\ g(x) \Rightarrow f_i(x) := -\frac{1}{g(\gamma_i)} \left[\frac{g(x)-g(\gamma_i)}{x-\gamma_i}\right]$$

(C'est a dire, $f_i(x)$ est l'inverse de $(x-\gamma_i) mod\ g(x)$). Soit

$$g(x) = \sum_{i=0}^{t} g_i x^i, \quad h_j = \frac{1}{g(\gamma_j)} \in \mathbb{F}_{q^m}^*$$

alors

$$\frac{g(x)-g(y)}{x-y} = \sum_{k+j \leq t-1} g_{k+j+1} y^j x^k \Rightarrow \sum_{i=0}^{n-1} w_i h_i \sum_{k+j \leq t-1} g_{k+j+1}.(\gamma_i^j) x^k \equiv 0\ mod\ g.$$

En effet

$$\frac{x^i - y^i}{x-y} = \sum_{k+j=i-1} y^j x^k$$

ceci implique

$$f_i(x) = -\frac{1}{g(\gamma_i)} \left[\frac{g(x)-g(\gamma_i)}{x-\gamma_i}\right] = -h_i \sum_{k+j \leq t-1} g_{k+j+1} \dot{(\gamma_i)}^j x^k.$$

Remarque 2.5.2. Pour pouvoir travailler avec les polynômes, on fait correspondre (de façon unique au mot $\underline{v}$ un polynôme $f_v(x)$ *modulo* g,

$$\underline{v} \mapsto f_v(x) = \sum_{i=0}^{n-1} v_i f_i(x) \Rightarrow \underline{w} \mapsto f_w(x) = \sum_{i=0}^{n-1} w_i f_i(x) \equiv 0\ mod\ g.$$

Si $\underline{v} = (v_0, v_1, \ldots, v_{n-1}) = \underline{w} + \underline{e}$ un mot reçu, $\underline{w} = (w_0, w_1, \ldots, w_{n-1}) \in \Gamma(L, g)$, $\underline{e} = (e_0, e_1, \ldots, e_{n-1}) \in \mathbb{F}_q^n$, alors

$$f_v(x) = S(x) = \sum_{i=0}^{n-1} e_i f_i(x) \equiv \sum_{i=0}^{n-1} v_i f_i(x) mod\ g$$

Puisque $\sum_{i=0}^{n-1} w_i f_i(x) \equiv 0\ mod\ g$. On obtient maintenant une matrice de contrôle de la forme explicite :

$$\sum_{i=0}^{n-1} w_i f_i(x) \equiv 0\ mod\ g \Rightarrow \sum_{i=0}^{n-1} w_i h_i \sum_{k+j \leq t-1} g_{k+j+1} \dot{(\gamma_i)}^j x^k \equiv 0\ mod\ g$$

ceci dit, pour tous les $k = 0, \ldots, t-1$ le coefficient de x^k a gauche est nul, donc une matrice de contrôle du code $\Gamma(L, g)$

$$\begin{pmatrix} h_0 g_t & h_1 g_t & ... & h_{n-1} g_t \\ h_0(g_{t-1} + g_t) & h_1(g_{t-1} + g_t\gamma_1) & ... & h_{n-1}(g_{t-1} + g_t\gamma_{n-1}) \\ \vdots & \vdots & \cdots & \vdots \\ h_0(g_1 + g_0\gamma_0 + \cdots + g_t\gamma_0^{t-1}) & h_1(g_1 + g_2\gamma_1 + \cdots + g_t\gamma_1^{t-1}) & ... & h_{n-1}(g_1 + g_2\gamma_{n-1} + \cdots + g_t\gamma_{t-1}^{n-1}) \end{pmatrix}$$

On obtient donc en faisant des transformations élémentaires une matrice de contrôle simplifiée :

$$H = \begin{pmatrix} h_0 & h_1 & \cdots & h_{n-1} \\ h_0\gamma_0 & h_1\gamma_1 & ... & h_{n-1}\gamma_{n-1} \\ \vdots & \vdots & \cdots & \vdots \\ h_0\gamma_0^{t-1} & h_1\gamma_1^{t-1} & ... & h_{n-1}\gamma_{n-1}^{t-1} \end{pmatrix}$$

Théorème 2.5.1. *Le code $\Gamma(L, g)$ a pour paramètres :*

1. *La dimension $k(\Gamma(L, g)) = n - mt$*
2. *la distance $d(\Gamma(L, g)) \geq t + 1$, $t = deg(g)$*

Preuve. La première assertion 1. est immédiate de la taille de la matrice H vue comme une matrice $mt \times n$ sur $\mathbb{F}_q$ après un choix d'une base de $\mathbb{F}_q^m$ sur $\mathbb{F}_q$.
L'assertion 2. est impliquée par l'unicité de décomposition d'une fraction rationnelle en éléments simples : Pour avoir un élément non nul

$$\sum_{i=0}^{n-1} \frac{c_i}{x - \gamma_i} = \frac{a(x)}{b(x)}$$

sous la forme d'une fraction irréductible $\frac{a(x)}{b(x)}$ avec $a(x)$ divisible par un polynôme irréductible $g(x)$ de degré t, il faut avoir au moins $s \geq t+1$ termes non-nuls car après avoir amener au dénominateur commun on aura le degré du polynôme $a(x) \leq s-1$. ■

Décodage des codes de Goppa rationnels(Pantchichkine 2004)

On utilise comme syndrome la fraction rationnelle

$$\sum_{i=0}^{n-1} \frac{e_i}{x-\gamma_i}$$

(le vecteur d'erreurs), représenté comme un polynôme *mod g*

$$S(x) = \sum_{i=0}^{n-1} e_i f_i(x) \ mod \ g$$

(Le polynôme de syndrome),

Remarque 2.5.3. Si $\underline{v} = (v_0, v_1, \ldots, v_{n-1}) = \underline{w} + \underline{e}$, un mot reçu, $\underline{w} = (w_0, w_1, \ldots, w_{n-1}) \in \Gamma(L, g)$, $\underline{e} = (e_0, e_1, \ldots, e_{n-1}) \in \mathbb{F}_q^n$, alors

$$S(x) = \sum_{i=0}^{n-1} e_i f_i(x) \equiv \sum_{i=0}^{n-1} v_i f_i(x) mod \ g$$

Puisque

$$\sum_{i=0}^{n-1} w_i f_i(x) \equiv 0$$

.

Ensuite, on utilise une version de l'algorithme de Berlekamp-Massey (Massey 1969). Soit $I := \{i \ tel \ que \ e_i \neq 0\}$, et on considère le polynôme locateur d'erreurs

$$s(x) = \prod_{i \in I} (x - \gamma_i),$$

et le polynôme évaluateur d'erreurs

$$u(x) = \sum_{i \in I} e_i \prod_{j \in I \setminus \{I\}} (x - \gamma_i)$$

On cherche $u(x)$, $s(x)$ a partir de la définition comme solution de la congruence :

$$u(x) \equiv s(x) S(x) (mod \ g(x))$$

Pour résoudre cette congruence, on utilise la division euclidienne et l'identité de Bezout. On calcule trois suites $s_n(x)$, $t_n(x)$, $u_n(x)$ avec la propriété

$$t_n(x)g(x) + s_n(x)S(x) = u_n(x)$$

où le degré de $u_n(x)$ décroît jusqu'à ce que $deg(u_{j+1}) < r$ et $deg(u_j) \geq r$ où $r = \dfrac{deg(g)-1}{2}$ à partir de

$$0.g(x) + 1.S(x) = S(x),\ 1.g(x) + 0.S(x) = g(x)$$

de telle façon que

$$(s_0(x), t_0(x), u_0(x)) = (1, 0, g(x)).$$

Il est clair que le couple

$$(u(x), s(x)) = (u_n(x), s_n(x))$$

est un unique couple satisfaisant la congruence ci-haut avec la propriété que le degré de $s(x)$ est inférieur au degré de $g(x)$.
Pour terminer le décodage, il faut vérifier que $s(x)$ est bien de la forme $s(x) = \prod_{i\in I}(x - \gamma_i)$ pour $I \subset \{1, \cdots, n\}$ et à déterminer explicitement, puis calculer les coefficients e_i.

Exemple 2.5.1. Prenons $q = 2$ et $g(x) = x^2 + x + 1 \in \mathbb{F}_2$. On considère
$\mathbb{F}_8 = \{0, 1, w, w^2, w^3, w^4, w^5, w^6\}$ où w est un générateur du groupe multiplicatif de $\mathbb{F}_8$. On a donc $m = 3$, $q = 2$, $t = 2$ et $n = 8$. Les zéros de $g(x)$ ne sont pas dans $\mathbb{F}_8$ mais dans $\mathbb{F}_4$. Si nous écrivons $\mathbb{F}_4 = \{0, 1, z, z^2\}$ alors il est clair que les zéros de $g(x)$ sont z et z^2. On énumère les éléments de L en posant $\gamma_1 = 0$ et $\gamma_i = w^{i-2}$ pour $2 \leq i \leq 8$. pour tout i, on a la formule générale

$$f_i(x) = -\frac{1}{g(\gamma_i)}\frac{g(x) - g(\gamma_i)}{x - \gamma_i}$$

Ce qui donne $f_i(x) = -\dfrac{1}{g(\gamma_i)}(x + \gamma_i + 1)$ et la matrice

$$H' = \begin{pmatrix} \frac{1}{g(0)} & \frac{1}{g(1)} & \cdots & \frac{1}{g(w^6)} \\ \frac{0}{g(0)} & \frac{1}{g(1)} & \cdots & \frac{w^6}{g(w^6)} \end{pmatrix}$$

est donc la matrice de parité pour le code dont les mots sont construits sur $(\mathbb{F}_{2^3})$. Elle est bien de rang 2. Pour obtenir une matrice de parité pour le code de Goppa $\Gamma(l, g) \subset \mathbb{F}_2^n$, il faut se rappeler que $(1, w, w^2)$ est une base de $\mathbb{F}_{2^3}$ sur $\mathbb{F}_2$ donc w est annulé par un

polynôme irréductible de degré 3, on suppose qu'il s'agit du polynôme $t^3 + t + 1$. On a par exemple

$$\frac{1}{g(w)} = \frac{1}{w^2 + w + 1} = \frac{1}{w^2 + w^3} = \frac{1}{w^2(w+1)} = \frac{1}{w^5} = w^2$$

et l'on peut ainsi obtenir la matrice de parité

$$H' = \begin{pmatrix} 1 & 1 & w^2 & w^2 + w & w^2 & w & w & w^2 + w \\ 0 & 1 & w+1 & w^2 + 1 & w^2 + w + 1 & w^2 + w + 1 & w^2 + 1 & w + 1 \end{pmatrix}$$

puis l'on déduit que

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

Ainsi le rang de H est 6, $\dim\Gamma(L, g) = 2$. En résolvant l'équation $H\underline{y}^t = 0$ pour $\underline{y} \in \mathbb{F}_2^8$, on obtient : $\Gamma(L, g) = (00000000; 11110100; 11001011; 00111111)$. D'où le code de Goppa $\Gamma(L, g)$ a une distance minimale égale à 5.

Chapitre Trois

LE CRYPTOSYSTÈME McELIECE

Il a été imaginé, comme son nom l'indique par Robert McEliece (McElice 1978), juste après l'invention des premiers cryptosystèmes à clé publique. C'est le plus ancien cryptosystème à clé publique utilisant les codes correcteurs d'erreurs et qui a la particularité d'être très rapide au niveau des calculs mais qui possède une clé publique très grosse. L'idée est de masquer un code de Goppa pour le faire passer pour un code aléatoire car décoder un code aléatoire est considéré comme un problème difficile. Le masquage ne doit ni altérer la structure du code ni empêcher l'algorithme de décodage.

3.1 Présentation du cryptosystème McEliece

Comme tous les cryptosystèmes à clé publique, ce système est constitué de 3 algorithmes : la génération de clés, le chiffrement et le déchiffrement.

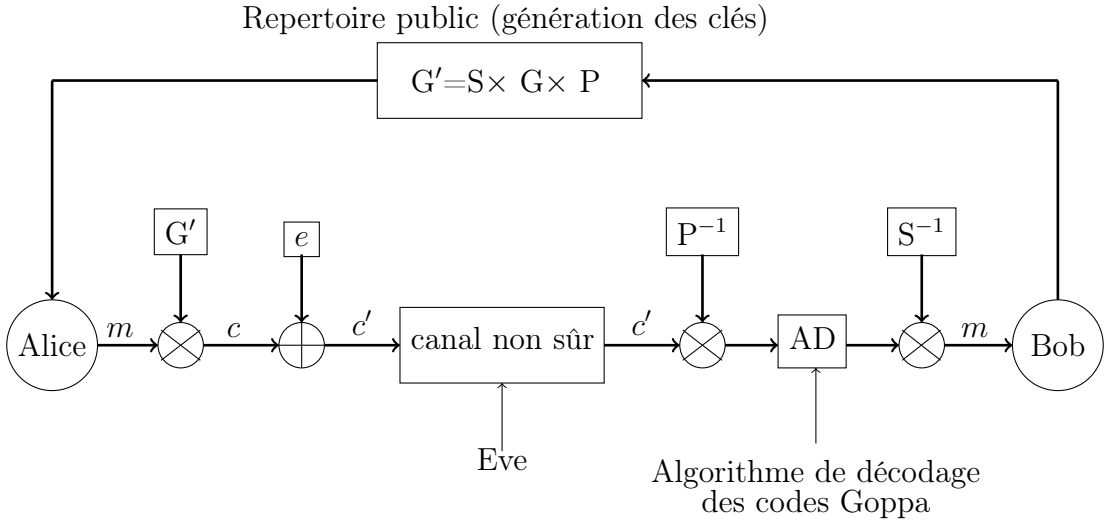

FIGURE 3.1 – Cryptosystème McEliece basé sur les codes de Goppa.

Génération des clés

Tout d'abord on se donne un code de Goppa corrigeant t erreurs et sa matrice génératrice G de taille $k \times n$ sur $\mathbb{F}_q$. Ensuite on masque cette matrice G pour la rendre indistinguable d'une matrice aléatoire (matrice dont les coefficients sont générés aléatoirement et ne respectent donc la structure d'aucun code) : pour y parvenir, on a génère une matrice de permutation P de taille $n \times n$ et une matrice inversible aléatoire S de taille $k \times k$ sur $\mathbb{F}_q$. On calcule $G' = SGP$, une matrice $k \times n$ à coefficients dans $\mathbb{F}_q$ qui est indistinguable d'un code aléatoire.

La clé publique est le couple (G', t).

La clé secrète est le quadruplet (S, G, P, t).

Chiffrement

Soit m un message de k bits que l'on veut chiffrer.

On commence par calculer le mot de code $\underline{c}$ de longueur n associé à m : $\underline{c} = m \times G'$.

On génère ensuite une erreur aléatoire ou encore une clé de sécurité $\underline{e}$ de longueur n et de poids t.

Le chiffré sera le mot code bruité :

$\underline{c}' = \underline{c} + \underline{e}$.

Déchiffrement

Pour déchiffrer en connaissant P, S et G il suffit de calculer :

$\underline{c}' \times P^{-1} = mG'P^{-1} + \underline{e}P^{-1} = mS \times G + \underline{e}P^{-1}$.

On obtient $mS \times G$ qui est un mot du code de Goppa et $\underline{e}P^{-1}$ qui est une erreur de poids t (car P est une permutation et conserve donc le poids des mots).

On peut donc décoder cette erreur et retrouver le message initial mS.

On multiplie enfin par S^{-1} pour retrouver le message m.

Remarque 3.1.1.

- La sécurité de ce cryptosystème repose à la fois sur le problème de distinguabilité d'un code de Goppa permuté d'un code aléatoire et sur le problème de décodage par syndrome.
- Avec cette construction, la clé publique fait nk bits, le taux de transmission est $\frac{k}{n}$ et la taille de blocs fait k bits. Le chiffrement est essentiellement un produit de matrice par un vecteur d'un coût de kn opérations. Le déchiffrement coûte environ $t^2(log_2 n)^3$ pour la part de décodage et n^2 pour le reste.

Remarque 3.1.2. Initialement, McEliece avait proposé dans (McEliece 1978) des pa-

ramètres $n = 1024$, $k = 524$, $t = 50$ qui étaient suffisant pour résister à une attaque au décodage par syndrome. Cependant avec les progrès faits en matière d'attaque, afin d'y résister avec une sécurité de 2^{80} opérations binaires, les paramètres qui semblent les mieux adaptés de nos jours sont : $n = 2048$, $k = 1685$ et $t = 33$. Cela donne un taux de transmission de $0,82$, des blocs de 1685 bits et une clé de 3 Méga-bits que l'on peut tout de même réduire en mettant la matrice de contrôle sous forme systématique.

3.2 Mise en œuvre du cryptosystème McEliece

Dans ce qui suit, nous travaillons sur $\mathbb{F}_2$ et Toutes les procédures écrites fonctionnent après avoir chargé le package LINALG dans le logiciel MAPLE (version MAPLE 12).

3.2.1 Génération des clés

- Génération du code de Goppa

Reprenons les paramètres utilisés pour la construction du code de Goppa dans l'exemple 2.5.1. On a donc $g(x) = x^2 + x + 1 \in \mathbb{F}_{2^3}[x]$ et $L = \{0, 1, w, w^2, w^3, w^4, w^5, w^6\} \subset \mathbb{F}_{2^3}$ puis une matrice de parité :

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

Ainsi on obtient le code de Goppa : $\Gamma(L, g) = (00000000; 11110100; 11001011; 00111111)$ qui est $2-$correcteur et a pour matrice génératrice : $G = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$

- Construction de la matrice inversible S

Il s'agit de construire une matrice de taille 2×2 inversible et aléatoire. La première idée consiste à générer une première matrice aléatoire. On regarde ensuite si elle est inversible (par la méthode du pivot de Gauss qui est rapide lorsque les coefficients sont dans $\mathbb{F}_2$. Si c'est le cas, on a trouvé une matrice S convenable, sinon on réitère le procédé jusqu'à réussite. La procédure suivante renvoie une matrice inversible et son inverse.

```
has2 := proc() rand() mod2 end
findinv := proc(2)
local dt;
global s, invs;
dt := 0;
while dt mod2 = 0 do s := randmatrix(2, 2, entries = has2);
dt := det(s) do;
invs := inverse(s) * dt;
invs := Z2(invs);
[evalm(s), evalm(invs)]
end
```

On notera que cette procédure fait appel à la sous-procédure Z2 qui permet de rester avec des coefficients dans $\mathbb{F}_2$.

On peut par exemple choisir

$$S = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \qquad \text{et} \qquad S^{-1} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

-Construction de la matrice de permutation P

Construire la matrice P revient à générer une permutation de l'ensemble $\{1, 2, \cdots, 8\}$. La première idée consisterait à engendrer un nombre aléatoire entre 1 et 8. On le compare ensuite à ceux engendrés précédemment : s'il est distinct de tous, on passe à l'élément suivant, sinon on essaye avec un autre nombre. La procédure suivante renvoie une matrice de permutation et son inverse.

```
perm := proc(n)
local lst, chx, i, j, r ;
global pm, invpm ;
lst := matrix([[seq(i, i = 1..n)]]) ;
pm := matrix([seq([seq(0, i = 1..n)], j = 1..n)]) ;
for r from 0 to n − 1 do
chx := (rand() mod(n − r)) + 1 ;
pm [r + 1, lst[1, chx]] := 1; lst[1, chx] := lst[1, n − r]
do ;
invpm := transpose(pm) ;
[evalm(pm), evalm(invpm)]
end
```

On obtient :

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{puis} \quad P^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

-Calcul de G'

On a : $G' = SGP$

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

La clé publique est : $(G', 2)$.

La clé secrète est : $(S, G, P, 2)$.

3.2.2 Chiffrement

Il s'agit tout d'abord de mettre son message sous forme d'une suite de bits. Ensuite, on découpe celui-ci en blocs de 2 bits. On multiplie chacun d'eux par la matrice G' qui est publique et on ajoute au résultat obtenu un vecteur d'erreur aléatoire e de poids égal à la valeur publiée $t = 2$. On met alors bout à bout tous les blocs de 8 bits obtenus pour avoir le message chiffré à transmettre.

- Message clair : $m = 100111$.

On obtient les blocs $m_1 = 10$; $m_2 = 01$; $m_3 = 11$.

- Calcul des mots de code associés : $c_i = m_i \times G'$

$c_1 = 11011001 \qquad c_2 = 01101111 \qquad c_3 = 10110110$

- Génération d'une erreur de poids 2 :

$e = 00100010$

- Calcul des blocs chiffrés ou mots de code bruités : $c'_i = c_i + e$

$c'_1 = 11111011 \qquad c'_2 = 01001101 \qquad c'_3 = 10010100$

- Message chiffré : $c' = 111110110100110110010100$.

Remarque 3.2.1. Si le message a par exemple un nombre de bits impair alors on peut décider de compléter la séquence par un 0 et s'arranger à rajouter un bit au chiffré également pour que le destinataire puisse le savoir.

3.2.3 Déchiffrement

On commence par mettre le message reçu sous forme d'une suite de bits et on découpe celui-ci en blocs de 8 bits. On suppose avoir reçu le bloc c'_1.

- On calcul : $c'_1 \times P^{-1}$

$c'_1 \times P^{-1} = 11110111$

-On calcule ensuite son syndrome $S(x)$:

$$\begin{aligned}\sum_{i=0}^{7} c'_{1i} f_i(x) &= f_1(x) + f_2(x) + f_3(x) + f_4(x) + f_6(x) + f_7(x) + f_8(x) \\ &= (x+1) + (x) + (w^2x + w^2 + w + 1) + (w^2x + wx + w + 1) + \\ &\quad (wx + w^2 + 1) + (wx + w^2 + w + 1) + (w^2x + wx + w^2 + 1) \\ &= w^2x + w\end{aligned}$$

On a donc $S(x) = w^2x + w$.

Soit $I = \{i \in [|1, 8|] avec.\ e_i \neq 0\}$, on considère les polynômes :

$$s(x) = \prod_{i \in I}(x - \gamma_i) \quad et \quad u(x) = \sum_{i \in I} e_i \prod_{j \in I \setminus \{I\}} (x - \gamma_i)$$

. Résolvons la congruence : $s(x)(w^2x + w) \equiv u(x)(mod\ (x^2 + x + 1))$.

On s'intéresse aux couples de polynômes non nuls $(s(x), u(x))$ vérifiant la relation : $S(x)s(x) \equiv u(x)(modg(x))$ où les degrés de $s(x)$ et $u(x)$ sont les plus petits possibles. On va donc chercher des solutions avec $deg(s(x)) = 1$ et $deg(u(x)) = 0$.

Remarque 3.2.2.

1. Si cette relation est vérifiée alors $u(x)$ est multiple du pgcd D de $g(x)$ et de $S(x)$; Une condition nécessaire à l'existence d'un tel couple $(s(x), u(x))$ est que le degré de D soit plus petit que 1.
2. Un tel couple est unique à multiplication par un scalaire près. En effet, si $(s_1(x), u_1(x))$ et $(s_2(x), u_2(x))$ en sont deux, on a $u_1(x)s_2(x) - s_1(x)u_2(x) \equiv 0(modg(x))$. Mais le polynôme de gauche est de degré au plus 1, donc est nul.

-Appliquons l'algorithme d'Euclide aux polynômes $g(x)$ et $S(x)$.

On considère trois suites $(A_0, A_1, \cdots, A_n)$, $(U_0, U_1, \cdots, U_n)$, $(V_0, V_1, \cdots, V_n)$ $(n \in \mathbb{N})$ de polynômes de $\mathbb{F}_{16}$[X] avec $(A_0, A_1) = (g(x), S(x))$, $(U_0, U_1) = (1, 0)$ et $(V_0, V_1) = (0, 1)$ et en notant q_{k+2} le quotient de la division euclidienne de A_k par A_{k+1} on a : $\forall k \in \mathbb{N}$

$(A_{k+2}, U_{k+2}, V_{k+2}) = (A_k, U_k, V_k) - (A_{k+1}, U_{k+1}, V_{k+1})q_{k+2}$.

Or $g(x) = (w^2x + w)((w^2 + w + 1)x + 1) + (w + 1)$.

Donc $(A_2, U_2, V_2) = (w + 1,\ 1,\ (w^2 + w + 1)x + 1)$ et 2 est le plus petit entier naturel pour lequel $deg(A_1) \geq 1 > deg(A_2)$.

On peut donc prendre $s(x) = (w^2 + w + 1)x + 1$ et $u(x) = w + 1$.

-Déterminons explicitement I

Sachant que $t = 2$ est le poids du vecteur d'erreur on effectue des produits de la forme : $(x - \gamma_i)(x - \gamma_j)$ modulo $g(x)$ avec $1 \leq i \leq 8$. On obtient finalement que

$$\begin{aligned}(x - \gamma_7)(x - \gamma_8) &= (x - w^5)(x - w^6) \\ &= (w + 1)x + (w^2 + w + 1) \\ &= (w^2 + w + 1)^2x + (w^2 + w + 1) \\ &= (w^2 + w + 1)s(x).\end{aligned}$$

On déduit alors $I = \{7, 8\}$.

-Le vecteur erreur est donc $e' = 000000011$.

-Le mot code de Goppa correspondant est déterminé par $c = (c'_1 \times P^{-1}) + e'$
soit $c = 11110100$

-On calcul aisément $mS = (a, b)$ connaissant G :
En effet $(a, b)G = (11110100)$ nous donne un système d'équations dont le nombre d'équations est supérieur au nombre d'inconnues. Ainsi :

$$\begin{cases} a+b &= 1 \\ a &= 1 \\ b &= 0 \end{cases}$$

Ceci entraine directement que $mS = 10$ et par conséquent $m = (10)S^{-1}$. D'où $m = 10$.
Ce qui est bien le message original envoyé.

Remarque 3.2.3.
- Le chiffrement et le déchiffrement avec le cryptosystème McEliece sont significativement plus rapides que les cryptosystèmes à clé publique plus répandus basés sur la théorie des nombres, tels que l'omniprésent RSA. L'inconvénient majeur qui empêche l'utilisation courante du cryptosystème McEliece est la taille des clés trop conséquentes.
- En 2009, les auteurs Berger, Cayrel, Gaborit et Otmani ont constaté une forte diminution de la taille des clés si on utilise des codes alternants quasi-cycliques dans(Otmani et *al* 2009). Ainsi l'utilisation des codes très structurés dans le but de limiter la clé publique à une sous matrice génératrice qui permet d'en reconstruire le reste serait d'un grand intérêt surtout si cela accroît davantage la sécurité de ce cryptosystème contre toute cryptanalyse.

3.3 Cryptanalyses du cryptosystème McEliece

Les notions de cryptographie et cryptanalyse sont fortement liées, car pour proposer des systèmes de chiffrement il faut connaître les attaques qu'ils vont subir. Bien qu'aucun algorithme efficace pour décomposer G' en (S, G, P) n'ait été découvert, une attaque structurelle a été proposée dans (Sendrier et Loidreau 2001). Cette attaque révèle une part de la structure de G' qui est généré à partir d'un code de Goppa binaire. Cependant, cette attaque peut être évitée simplement en n'utilisant pas de clé publique faible. Toutes les autres attaques connues sont utilisées pour déchiffrer le message chiffré sans casser la clé publique.

Ces attaques sont classés en deux grandes catégories dans (Cayrel 2008) : attaques critiques et attaques non-critiques. Les attaques non-critiques sont celles qui dépendent fortement des paramètres et peuvent alors être contrecarrées seulement en augmentant la valeur de ces paramètres. Les attaques critiques sont plus rapides que les attaques non-critiques et réalisables pour des paramètres réalistes. Elles peuvent en évitant les codes ayant une faiblesse structurelle car toutes les attaques critiques exploitent des faiblesses structurelles des codes utilisés ou nécessitent des informations additionnelles, comme par exemple une connaissance partielle des textes en clair. Sans ces informations additionnelles, aucun algorithme efficace n'est connu pour déchiffrer un texte chiffré arbitrairement à l'aide du cryptosystème McEliece.

3.3.1 Attaques non-critiques

Les deux attaques suivantes peuvent être rendues infaisables simplement en augmentant la taille des paramètres.

La cryptanalyse de McEliece

McElice proposa une attaque contre son protocole. Elle consiste à choisir aléatoirement k bits du mot reçu c' en espérant qu'il n'y ait pas d'erreurs sur ces bits là. Soient c'_k le vecteur uniquement formé par les k bits choisis et G'_k la sous matrice de G' dont on ne garde que les k colonnes correspondantes aux k bits choisis.
S'il n'y a effectivement pas d'erreurs dans les k bits choisis, alors : $c'_k G'^{-1}_k = m$.L'inversion de la matrice G'_k nécessite $\mathcal{O}(k^3)$ opérations. La probabilité de n'avoir aucune erreur en choisissant aléatoirement k bits est : $T = \dfrac{C^k_{n-t}}{C^k_n}$.

Le coût de l'algorithme est donc $\dfrac{k^3}{T}$. Comme G' est la matrice génératrice d'un code de distance minimale supérieure à $2t$, si $c'_k G'^{-1}_k$ n'est pas le véritable message m, alors $mG' + c'_k G'^{-1}_k G'$ est de poids au plus $2t$ (c'est la différence entre le vrai message et le message trouvé par le cryptanalyste). De là, si $c' + c'_k G'^{-1}_k G'$ est de poids inférieur ou égal à t, alors on peut dire que $c'_k\ G'^{-1}_k = m$.

On peut généraliser cet algorithme de la manière suivante en permettant d'avoir quelques erreurs dans le vecteur c'_k :
Soit j un paramètre dont la valeur sera discutée par la suite :

1. Soit c'_k un k-bit choisis aléatoirement parmi les bits du message reçu c'. Soit G'_k la matrice formée par les colonnes de G' correspondant aux k bits pris de c'. On calcule $G'^{-1}_k G'$ et $c' + c'_k G'^{-1}_k G'$.
2. On choisit un k-bit e_k de poids inférieur ou égal à j (e_k modélisera l'erreur). Si $(c'+c'_k G'^{-1}_k G')+e_k(G'^{-1}_k G')$ est de poids inférieur ou égal à t, on s'arrête : $m = c'_k G'^{-1}_k$.
3. Si l'on a utilisé tous les vecteurs possibles e_k tels que $W_H(e_k) \leq j$, on recommence à l'étape 1, sinon à l'étape 2.

Soit Q_i la probabilité qu'il y ait exactement i erreurs dans c'_k,

$$Q_i = C_t^i \frac{C_{n-t}^{k-i}}{C_n^k}.$$

On doit exécuter T_j fois l'étape 1 avec :

$$T_j = \frac{1}{\sum_{i=0}^{j} Q_i}.$$

Soit N_j le nombre de vecteurs e_k distincts, de poids inférieur ou égal à j :

$$N_j = \sum_{i=0}^{j} C_k^i.$$

Étant donné que l'étape 1 nécessite à peu près ak^3 opérations, avec a petit, et l'étape 2 nécessite bk opérations avec b petit, le coût de cet algorithme est de l'ordre de :

$$T_j(ak^3 + N_j bk)$$

Remarque 3.3.1. On choisit alors j minimisant ce coût. Dans la plupart des cas, avec des dimensions raisonnables, il faudra prendre $j = 2$. Par exemple, si l'on prend $n = 1024$, $k = 644$, $t = 38$, l'attaque nécessite 2^{73} opérations élémentaires.

Attaque par ensemble d'information

De toutes les attaques générales (pas sur des codes spécifiques) cette attaque semble être celle qui possède la plus petite complexité. On essaie de retrouver les k symboles d'informations de la manière suivante :

1. La première étape consiste à choisir k des n coordonnées aléatoirement en espérant avoir sélectionné toutes les t positions d'erreurs.

2. On essaie alors de retrouver le message en résolvant un système linéaire de taille $k \times k$ (binaire). Soit G'_k, c'_k et e_k représentant les k colonnes prisent dans G', c' et e respectivement. Elles vérifient la relation suivante :

$$c'_k = mG'_k + e_k$$

Si $e_k = 0$ et G'_k est non singulier, m peut être retrouvé par : $m = c'_k G'^{-1}_k$

Remarque 3.3.2. Le coût de calcul de cette version (appelée attaque par ensemble d'informations originale) est $T(k) \times P_{n,k,t}$ où

$$P_{n,k,t} = \prod_{i=0}^{k-1} \left(1 - \frac{t}{n-i}\right).$$

La quantité $T(k)$ est le coût moyen d'opérations nécessaires pour résoudre un système linéaire $k \times k$ sur $\mathbb{F}_2$. Comme mentionné dans (McEliece 1978), la résolution d'un tel système binaire nécessite environ k^3 opérations.

De plus, si $e_k \neq 0$, m peut être retrouvé en devinant k positions parmi les éléments de petits poids de Hamming. On appellera cette attaque, *attaque par ensemble d'informations généralisée*. Une itération de l'algorithme s'effectue de la manière suivante :

1. Permuter les colonnes de la matrice génératrice de manière aléatoire.
2. Faire une élimination gaussienne des colonnes de la matrice pour obtenir la forme $G = (I_k; A)$, avec les permutés correspondants du chiffré $c' = (c_1 + e_1|c2 + e2)$.
3. Deviner que l'erreur e_1 est de poids au plus p et vérifier que l'erreur $e = (e_1|e_2)$ est de poids t.

Remarque 3.3.3. La probabilité $\pi_{(p,n,k,t)}$ qu'une permutation des colonnes mène à une configuration favorable est

$$\pi_{(p,n,k,t)} = \sum_{i=0}^{p} \frac{C_{n-t}^{k-i} C_t^i}{C_n^k}$$

Pour chaque itération, une estimation du nombre d'opérations est :

1. $\dfrac{k^2 n}{2}$ pour l'élimination gaussienne ;
2. (Cayrel 2008) $\dfrac{k}{2} + \sum_{i=1}^{p} C_k^i i$ additions sur les mots de $(n-k)$-bits de A.

Alors une estimation du coût de cet algorithme est

$$W_{(p,n,k,t)} = \frac{\frac{k^2 n}{2} + (n-k)[\frac{k}{2} + \sum_{i=1}^{p} C_k^i i]}{\pi_{p,n,k,t}}$$

3.3.2 Attaques critiques

Les attaques suivantes ne peuvent pas être évitées en augmentant la taille des paramètres. Elles utilisent des faiblesses structurelles ou nécessitent de l'information supplémentaire.

Attaque à texte clair partiel

Avoir une connaissance partielle du texte clair réduit considérablement le coût de calculs des attaques contre le cryptosystème McEliece . Par exemple, soient m_l et m_r représentant les k_l bits de gauche et les k_r bits restant du message clair m donc $k = k_l + k_r$ et $m = (m_l|m_r)$. Supposons qu'un adversaire connaisse m_r. Alors, la difficulté de retrouver le message clair inconnu m_l dans le cryptosystème McEliece avec pour paramètres (n, k) est équivalent au fait de retrouver le message clair en entier dans le cas d'un chiffrement à l'aide de McEliece de paramètres (n, k_l), puisque

$$\begin{aligned} c &= mG' + e \\ c &= m_l G'_l + m_r G'_r + e \\ c + m_r G'_r &= m_l G'_l + e \\ c' &= m_l G'_l + e \end{aligned}$$

où G'_l et G'_r sont les k_l lignes supérieures et k_r les autres lignes de G', respectivement.

Attaque par renvoi de message

Supposons maintenant que, à cause d'un accident, ou du fait de l'action du cryptanalyste, à la fois $c_1 = mG' + e_1$ et $c_2 = mG' + e_2$, avec $e_1 \neq e_2$ sont envoyés. Dans ce cas, il est facile pour le cryptanalyste de retrouver m à partir du système des c_i.

Remarque 3.3.4. On remarque que $c_1 + c_2 = e_1 + e_2 (mod\ 2)$. Ainsi pour des paramètres proposés par McEliece $(n, k, t) = (1024, 524, 50)$, un renvoi de message peut être facilement détecté en observant le poids de Hamming de la somme de deux chiffrés. Quand les messages sont différents, le poids attendu de la somme est d'environ 512 (en général le poids attendu est k). Alors que quand les deux messages sont identiques, le poids de la somme ne peut excéder 100 (ou en général $2t$).

Tout d'abord nous calculons deux ensembles à partir de $(c_1 + c_2)$. On définit L_0 et L_1 de la manière suivante :

$$L_0 = \{i \in \{1 \cdots n\} : c_{1i} + c_{2i} = e_{1i} + e_{2i} = 0\}$$
$$L_1 = \{i \in \{1 \cdots n\} : c_{1i} + c_{2i} = e_{1i} + e_{2i} = 1\}$$

Nous voulons tirer un avantage du fait que :

- $i \in L_0$ est plus probable si ni c_{1i}, ni c_{2i} ne sont altérés par aucune erreur, alors que
- $i \in L_1$ implique qu'exactement un de c_{1i} ou c_{2i} est altéré par une erreur.

Supposons que les vecteurs d'erreurs e_1 et e_2 sont choisis indépendamment, alors pour n'importe quel $i \in \{1 \cdots n\}$, la probabilité que les deux vecteurs erreurs soient à 1 sur la position d'indice i est

$$P(e_{1i} = e_{2i} = 1) = \left(\frac{50}{1024}\right)^2 \approx 0.0024.$$

Remarque 3.3.5. Pour la plupart des $i \in L_0$ on a donc $e_{1i} = e_{2i} = 0$. Le cryptanalyste peut essayer de deviner les 524 colonnes non perturbées de celles indexées par L_0.

Soit p_s la probabilité que précisément s coordonnées sont perturbées simultanément par e_1 et e_2 alors

$$p_s = P(|\{i : e_{1i} = 1\} \bigcap \{i : e_{2i} = 1\} = s) = \frac{C_{50}^{s} C_{974}^{50-s}}{C_{1024}^{50}},$$

ainsi, e_2 doit choisir s positions d'erreurs parmi les 50 perturbées par e_1 et les $50 - s$ restantes parmi celles inchangées par e_1. De plus, le nombre attendu de L_1 est

$$E(|L_1|) = \sum_{i=0}^{50} (100 - 2s) p_s \approx 95.1,$$

Remarque 3.3.6. Lorsque que $|L_1| = 94$ par exemple, alors $|L_0| = 390$ où seulement 3 positions sont perturbées. On observe que la probabilité de deviner 524 colonnes non perturbées parmi celles indexées par L_0 est $\frac{C_{927}^{524}}{C_{930}^{524}} \approx 0.0828$. Le cryptanalyste peut alors espérer un succès dans ce cas avec seulement 12 tentatives. Ces résultats sont meilleurs d'un facteur de 10^{15} que ceux obtenus en devinant k colonnes non perturbées sans renvoi de message.

⋆ ⋆ Chapitre Quatre ⋆ ⋆

IMPLICATIONS PEDAGOGIQUES

L'opportunité de rédiger ce mémoire nous a offert l'occasion de faire nos premiers pas d'autonomie en termes d'investigations scientifiques et de déploiement de nos aptitudes à comprendre, à construire un raisonnement logique sans oublier la maîtrise indispensables des outils de nouvelles technologies de l'information et de la communication incontournables pour tout enseignant en ces moments où les apprenants sont aspirés par les téléphones androïds, ipad, tablettes...

Nous nous proposons de relever dans ce chapitre quelques repères de l'impact de cet exercice sur le système éducatif.

4.1 Construire et consolider des connaissances

Pour mener à terme ce travail, il a fallu puiser dans nos compétences à pouvoir comprendre une situation-problème, à proposer un formalisme et d'étudier les propriétés des objets mathématiques obtenus. Ce travail nous a permis :

☞ d'approfondir nos connaissances en arithmétique principalement sur le calcul dans les bases et précisément la base deux, motivé ici par leur utilisation dans les ordinateurs ;

☞ d'approfondir nos connaissances sur les polynômes notamment les critères d'irréductibilité de certains polynômes ;

☞ de découvrir un champ d'application des mathématiques que nous présenterons aux élèves afin de les motiver.

Par transposition, les documents de travail sur lesquels porte notre étude, peuvent être perçus comme une ressource pédagogique dont le contenu doit être partager avec des apprenants. C'est ainsi que nous avons décomposé la ressource pour en donner une reconstitution en termes de définitions des concepts à étudier ; en termes de propriétés

que vérifient les concepts et en termes de résultats qu'on peut déduire en les mettant ensemble. Cette démarche est fondamentale dans le quotidien d'un enseignant de mathématiques que nous aspirons à être. Nous pensons notamment à l'élaboration de nos futurs cours à construire à partir de diverses ressources éducatives.

4.2 Aptitude à mener un raisonnement logique

Le raisonnement mathématique permet d'associer, de différencier, de catégoriser, de mesurer, d'évaluer, de tester des hypothèses, de démontrer un processus, de tirer des conclusions à partir d'informations données ou de lois générales, de retrouver des informations manquantes par logique, d'aller des causes aux conséquences et inversement, de mettre au jour les contradictions ou incohérences, de justifier un résultat... C'est donc un type de raisonnement essentiel pour comprendre et analyser le monde mais aussi pour beaucoup d'opérations de la vie quotidienne et professionnelle nous demandant d'analyser logiquement des situations et de prendre des décisions. On pourra donc ainsi développer chez les élèves des capacités qui contribueront à l'avancement vers des nouvelles connaissances. Celles ci ont pour but d'initier l'élève à la pratique de la recherche de la vérité par le biais du raisonnement en les rendant capable d'utiliser adéquatement les ressources documentaires, les méthodes d'investigation et d'analyse appropriées. L'élève sera donc à même de développer les capacités suivantes :

☞ réfuter une proposition lorsqu'elle est fausse en donnant un contre-exemple ;

☞ prouver une proposition lorsqu'elle est vraie en utilisant un raisonnement logique ;

☞ la valeur et la pertinence du nouveau savoir produit par le travail ;

☞ l'aptitude à intégrer les différentes connaissances ;

☞ l'aptitude à construire une problématique ;

☞ l'aptitude à la recherche : rigueur méthodologique, logique de l'argumentation et de la démonstration ;

☞ la qualité de la présentation selon les normes d'un travail scientifique ;

☞ la qualité de la présentation matérielle et typographique.

4.3 Initiation aux technologies de l'information et de la communication

Pendant la rédaction de ce mémoire, nous avons eu à utiliser les outils des technologies de l'information et de la communication ; il s'agit : du **micro-ordinateur**, du **vidéoprojecteur**, des logiciels **Matlab**, **Maple**, **GeoGebra**, **Latex** (de son compilateur Miktex, de ses éditeurs TeXnicCenter, TeXmaker, TeXstudio et Winedit) ou d'**internet**. L'emploi de ces outils informatiques peut nous aider à faire des recherches sur internet pour actualiser les contenus à enseigner, à préparer un cours, à saisir une épreuve et à présenter une leçon. Notons que l'usage du vidéoprojecteur peut permettre d'illustrer une définition ou une propriété lorsqu'elle est introduite.

Ce mémoire a certainement contribué au développement de nos compétences à savoir :

☞ réaliser les supports de qualité telles que les épreuves de mathématiques ;

☞ préparer et présenter une leçon ;

☞ utiliser un vidéprojecteur lors d'une leçon.

♣ Conclusion et Perspectives ♣

Nous avons étudié dans le cadre de ce mémoire le cryptosystème McEliece et constaté qu'il s'agit d'un cryptosystème proposant un chiffrement et un déchiffrement usant d'opérations très rapides car basées sur l'algèbre linéaire, possédant des paramètres très facilement modifiables et une résistance face à l'ordinateur quantique. Mais le principal et peut-être le seul handicap qui empêche l'utilisation courante du cryptosytème McEliece est la taille des clés publiques trop grande. Une méthode permettant d'accroître le taux de transmission et de compresser la taille des clés a été proposé en 2009 par Thierry Berger, Pierre-louis Cayrel, Philippe Gaborit et Ayoub Otmani dans (Berger et *al* 2009). Il est important de noter que cette méthode utilise les codes alternants quasi-cyclique.
Parvenu au terme de ce travail, il serait intéressant d'aborder ce problème en ajoutant autant d'erreurs que permet la méthode de décodage en liste des codes de Goppa pour une taille de clé fixée. Ce procédé pourrait augmenter le niveau de sécurité, tant que le nombre d'erreur reste inférieur à la borne de Gilbert Varshamov.

♣ Bibliographie ♣

[1] J. Buchman (2006) Introduction à la cryptographie. Dunod, France Quercy, Mercuès, 273 pages.

[2] T. Berger, P.L. Cayrel, P. Gaborit et A. Otmani (2009) Reducing key length of the McEliece Cryptosystèm. Dans Bart PRENEEL, editeur. Progress in cryptography-AFRICACRYPT 2009, volume 5580 de Lectures Notes in Computer Science : 77-97. Springer Berlin / Heidelberg.

[3] A. Canteaut and F. Chabaud (1998) A new algorithm for finding minimum-weight words in a linear code : Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. IEEE Transactions on Information Theory, 44 (1) :367-378.

[4] P. L. Cayrel (2008) Construction et optimisation de cryptosystème basés sur les codes correcteurs d'erreurs. Thèse Ph.D, Université de Limoges, Faculté des Sciences et Techniques, France.

[5] W. Diffie and M. Hellman (1976) New Directions in Cryptography. IEEE Transactions on Information Theory, 22 : 644-654.

[6] C. Faure (2009) Études des systèmes cryptographiques construits à l'aide des codes correcteurs d'erreurs en métrique de Hamming et en métrique rang. Thèse Ph.D, École Polytechnique de Paris, INRIA, France.

[7] M. Finias (2004) Nouvelles constructions utilisant les codes correcteurs d'erreurs en cryptographie à clé publique. Thèse Ph.D, École Polytechnique de Paris, INRIA, France.

[8] G. Hoffmann (2011) Implementation of McElice using quasi dyadic Goppa codes. Darmstadt. Allemagne : 21-30.

[9] A. Kerckhoff (1883) La cryptographie militaire. Journal des sciences militaires, vol.IX : 161-191.

[10] A. Kraus (2012) Cours de cryptographie, Université Pierre et Marie Curie, France. Available from : www.usthb.dz/fmath/IMG/pdf/chapitre3.pdf. (accessed 18/08/2015)

[11] P. Loidreau and N. Sendrier (2001) Weak keys in McEliece public key cryptosystem. IEEE Transansactions on Information Theory.

[12] J. L. Massey (1969) Shift-register synthesis and BCH decoding. IEEE Transactions on Information Theory, 15(1) : 122-127.

[13] R. J. McEliece (1978) A public-key cryptosystem based on algebraic coding theory. DSN Prog.Rep., Jet Prop. Lab., California Inst. Technol., Pasadena, CA : 114-116.

[14] A. A. Pantchichkine (2004/2005) Cours de Mathématiques des codes correcteurs d'erreurs (master 2 de mathématiques (M2P),Cryptologie, Sécurité et Codage d'Information, Université de Grenoble, France.

[15] F. Rodier (2008) Rappel des corps finis. Available from : http ://iml.univ-mrs.fr/ rodier/Cours/Rappelcorpsfinis.pdf (accessed : 18/08/2015)

[16] R. Rivest, A. Shamir, and L. Adleman (1978) A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM, 21 : 120-126.

[17] C. E. Shannon (1949) Communication Theory of Secrecy Systems. Bell system technical journal, 28 :656-715.

[18] P. W. Shor (1995) Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM Journal on Computing, 26 : 1484-1509.

[19] S. Tison (2003) Complexité des problèmes.Available from : http ://www.fil.univ-lille1.fr/tison/AAC/C09/C7impr.pdf. (accessed : 05/05/2016)

[20] A. Vardy (1997) The intractability of computing the minimum distance of a code. IEEE Transactions on Information Theory 43(6) : 1757-1766.