THÈSE / UNIVERSITÉ DE BRETAGNE OCCIDENTALE

sous le sceau de l'Université européenne de Bretagne

pour obtenir le titre de

DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE

Mention : Sciences et Technologies de l'Information et de la Communication - Spécialité Communications Numériques

École Doctorale SICMA-ED 373

présentée par

Marion Candau

Préparée au LMBA UMR-CNRS 6205
et au Lab-STICC UMR-CNRS 6285

Codes correcteurs d'erreurs convolutifs non-commutatifs

Thèse soutenue le 9 décembre 2014
devant le jury composé de :

Delphine BOUCHER
Maître de Conférences, Université de Rennes 1 / E*xaminatrice*

Jean François DIOURIS
Professeur, Ecole Polytechnique de l'Université de Nantes / E*xaminateur*

Caroline FONTAINE
Chargée de Recherche CNRS, Télécom Bretagne / E*xaminatrice*

Roland GAUTIER
Maître de Conférences, Université de Bretagne Occidentale / C*o-directeur de thèse*

Johannes HUISMAN
Professeur, Université de Bretagne Occidentale / C*o-directeur de thèse*

Pierre LOIDREAU
Ingénieur de l'armement à la DGA MI et chercheur associé à l'IRMAR, Université de Rennes 1 / R*apporteur*

Emanuel RADOI
Professeur, Université de Bretagne Occidentale / E*xaminateur*

Jean-Pierre TILLICH
Directeur de Recherche, INRIA Rocquencourt / R*apporteur*

Remerciements

Thèse réalisée sans café, ni thé, et avec une pomme par jour

Ce doctorat s'est réalisé grâce au soutien financier de la Région Bretagne que je remercie. C'est une région chère à mon cœur et je suis heureuse d'avoir pu réaliser ma thèse à l'Université de Bretagne Occidentale au sein du Laboratoire de Mathématiques de Bretagne Atlantique (LMBA) et du Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la Connaissance (Lab-STICC).

Tout d'abord, ce doctorat a pu se conclure dans la joie et l'allégresse grâce aux membres du jury. Je remercie sincèrement Jean Francois Diouris, professeur à l'Université de Nantes, qui a accepté d'être président du jury. Je remercie également Caroline Fontaine, chargée de recherche CNRS à Télécom Bretagne, et Emanuel Radoi, professeur à l'UBO d'avoir accepté d'être examinateur/trice de ma thèse. Je remercie plus particulièrement Delphine Boucher, maître de conférences à l'Université de Rennes 1, pour d'une part, avoir également accepté d'être membre du jury et d'autre part, pour m'avoir invitée à présenter mes travaux lors du Séminaire Calcul Formel et Complexité de Rennes. J'exprime ma gratitude à Jean Pierre Tillich, directeur de recherche chez INRIA Rocquencourt, qui a accepté de rapporter cette thèse et pour les nombreux conseils qu'il m'a donné pour améliorer ce document. Enfin, je souhaite remercier chaleureusement le dernier membre du jury et non des moindres, Pierre Loidreau, mon maître de stage de master pour m'avoir aidée à trouver ce doctorat et pour m'avoir fait l'honneur d'être rapporteur. Ses nombreux conseils m'ont permis d'étoffer certaines parties de mon manuscrit.

Un doctorat n'est pas une aventure solitaire, c'est une aventure humaine avec de nombreuses personnes dont les directeurs de thèse. Sans eux, on est vite perdu et on ne sait pas trop où l'on va. Donc je remercie Johannes Huisman, professeur à l'UBO, pour sa patience face à mes connaissances mathématiques assez limitées au début de ce doctorat, pour ses explications au tableau toujours très claires et pédagogiques et pour sa disponibilité, malgré ses nombreuses responsabilités au sein de l'université. Tout le côté mathématique de ma thèse lui doit beaucoup. J'adresse ma profonde gratitude à mon deuxième directeur, Roland Gautier, maître de conférences à l'UBO. Il m'a donné de nombreux conseils pour les applications de mes

codes et m'a expliqué très simplement la problématique de cette thèse. De plus, sa bonne humeur et son soutien permanent m'ont permis de réaliser cette thèse dans d'excellentes conditions. Merci également de m'avoir aidée dans les démarches administratives, pas toujours faciles dans le monde impitoyable de la recherche publique.

J'ai eu la chance d'effectuer mon doctorat dans deux laboratoires différents, j'ai donc deux équipes à remercier. Je remercie tout d'abord l'équipe coté Lab-STICC, Ken-Ji, Charles, Phillipe, Ludovic, Mélanie, Emanuel, Yasamine, Gilles, Koffi et Quang pour leurs nombreuses discussions intéressantes dans les couloirs du premier étage du bâtiment C ou lors de pots ou de réunions. Puis, je remercie tous mes collègues du laboratoire de mathématiques, avec lesquels j'ai pu échanger lors de séminaires ou de pots ou à la bibliothèque. Je tiens à remercier Annick et Marie-Pierre, qui m'ont beaucoup aidée pour tout le côté administratif et toujours avec le sourire. Ensuite, je tiens à remercier en particulier les doctorants du bureau C302 ou des autres bureaux des bâtiments H ou N. Je remercie en premier lieu les anciens, Jean-Baptiste, Christine, Brice, Hayk, Grace et Cuong. Leur bonne humeur et leur goût pour le jeu m'ont permis de démarrer ce doctorat dans une excellente ambiance. Ensuite je remercie les nouveaux, Thi Hien (alias Jeune Hien), Mokdad, Lara, Inas, Nathalie, Elsa et Kamel pour les nombreux repas partagés ensemble au RU Armen (avec Nam aussi). Jeune Hien, j'espère que tu vas continuer à apprendre le français avec Nathalie comme professeur désormais. Il y a une doctorante que je dois plus particulièrement remercier, c'est Thi Thu Hien, alias Madame Hien. On a commencé notre thèse presque au même moment, on a partagé de nombreux repas, au cours desquels on a échangé sur nos cultures respectives, notamment l'égalité homme/femme. Je l'ai beaucoup taquiné, mais j'espère qu'elle ne m'en tient pas trop rigueur. C'est pour toutes ses raisons que je te remercie chaleureusement mon amie. J'espère que ta thèse se terminera sereinement et que tu soutiendras bientôt, avec succès je n'en doute pas.

Lors d'un doctorat, on doit valider un certain nombre de formations. J'ai eu la chance de participer aux Doctoriales 2013, semaine en immersion où l'on rencontre des professionnels afin d'élargir nos horizons pour trouver un travail après le doctorat. J'y ai rencontré de nombreux doctorants et j'ai gagné avec mon équipe le prix du projet innovant. Je remercie donc toute mon équipe des Doctoriales : Marianne, Georges, Thomas, Iulia, Mathieu, Carole et les autres, ainsi que d'autres doctorants très sympas : July, Mahrez, Mathilde, Tram et Anne.

Pour se changer les idées lors de 3 ans de doctorat, les activités extra-bureau sont indispensables. J'ai pu, grâce à Hien, jouer au badminton (presque) tous les vendredis soirs à Télécom Bretagne avec Nhan, Huong, Ven et les autres, que je remercie pour m'avoir acceptée dans leur groupe. Le badminton, c'est un sport sympa, mais pas autant que le football. Il me parait donc naturel de remercier l'ensemble du club du FC Bergot, notamment les coachs Brice, Jacques x2, Gilles, Martial, Bernard, Marie et Armelle. Ces entraînements et ces matchs de football étaient un vrai bonheur au cours de ces 3 ans. J'y ai rencontré de nombreuses amies, Julia, Myriam, Pauline, Christelle, Estelle, Aurore, Sophie, Valérie et Manue que je remercie pour

tous les bons moments partagés ensemble. Allez les vertes !

A l'heure des communications numériques, on rencontre beaucoup de gens sur les réseaux sociaux, notamment sur Twitter. On échange sur les difficultés du doctorat, on se conseille mutuellement, on s'encourage. Je remercie donc pour cela @PhDelirium, @Cernach, @Lazarrean, @cepcam, @Fantaroux (et la #TeamTintinThèse), @Lulle1, @PresquePartout, @misanthropolog, @stephaniecouv et @annenat.

Un doctorat, ce sont 3 ans précédés d'un master et d'une licence. Je tiens donc à remercier mes professeurs de licence de l'Université de Pau et mes professeurs de master de l'Université de Bordeaux, notamment Gilles Zémor qui m'a encouragée dans la voie de la recherche. Durant ces études, j'ai fait de belles et nombreuses rencontres, je remercie donc Magali, Laurent, Hélène, Patxi, Vanessa, Camille, Emilie, Elodie, Yoan, Daton, Tania, Lionel, Adrian, Prince et Romain, pour m'avoir supportée et encouragée (et pour les nombreux moments de rigolade aussi).

Mes prochains remerciements vont à ma famille. Tout d'abord j'adresse mes remerciements à la famille Landeau qui m'a accueillie chaleureusement à Bruz lors de mon stage à la DGA MI. Puis, je remercie mes oncles et tantes, cousins et cousines, grands parents qui ont essayé de comprendre ce que je faisais, mais je ne suis pas sûre qu'ils aient vraiment compris. Enfin, je ne remercierai jamais assez mes parents et mon frère, qui m'ont soutenue de façon inconditionnelle.

Enfin, comme dirait Hien « le meilleur pour la fin », je tiens à remercier Jérémy, qui, depuis un an (et ces fameuses Doctoriales), m'apporte tout l'amour et le soutien dont j'ai besoin pour être heureuse.

Table des matières

Introduction **1**
Contexte 1
Plan 1

1 Les codes convolutifs **3**
1.1 Contexte 3
1.1.1 Chaîne de transmission 3
1.1.2 Codage de canal 4
1.2 Codes correcteurs d'erreurs en bloc 4
1.3 Codes convolutifs 9
1.3.1 Codage 9
1.3.2 Décodage 13
1.3.3 Propriétés 18
1.3.4 Reconnaissance de codes 19
1.3.5 Aspects cryptographiques 20

2 Introduction aux objets mathématiques **23**
2.1 Théorie des groupes 23
2.2 Actions de groupe 28
2.3 Algèbre de groupe 31
2.4 Domaine fondamental et topologie algébrique 33
2.5 Diviseurs de zéro dans une algèbre d'un groupe fini 34

3 Codes convolutifs définis sur un groupe non-commutatif **37**
3.1 Codes convolutifs sur un groupe non-commutatif infini 37
3.2 Codes convolutifs sur un groupe non-commutatif fini 38

4 Codes convolutifs sur le groupe diédral infini **43**
4.1 Groupe diédral infini 43
4.2 Algèbre de groupe 44
4.3 Injectivité du code 45
4.4 Codage par un registre 48
4.5 Décodage 56
4.6 Propriétés 56
4.7 Reconnaissance de code 58
4.8 Généralisation de ces codes 59
4.9 Conclusion 61

5 Codes sur le produit semi-direct $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ 63
5.1 Théorie de Fourier . . . 64
5.2 Codes sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ . . . 68
5.3 Parcours des éléments . . . 70
5.3.1 Parcours avec une droite de pente irrationnelle . . . 72
5.3.2 Parcours basé sur un système dynamique discret . . . 75
5.3.3 Intervalles d'encodage de longueur choisie . . . 77
5.3.4 Problèmes de précision . . . 78
5.4 Codage . . . 81
5.5 Décodage . . . 83
5.6 Propriétés . . . 85
5.6.1 Actions de groupe et distance minimale . . . 85
5.6.2 Calcul de la distance minimale et comparaison avec celle des codes linéaires en bloc . . . 87
5.6.3 Reconnaissance de code et aspects cryptographiques . . . 88
5.7 Exemple complet . . . 90
5.8 Conclusion . . . 94

6 Étude des sous-groupes de congruence $\Gamma(N)$, $\Gamma'_0(N)$ et $\Gamma'_1(N)$ 97
6.1 Géométrie hyperbolique . . . 97
6.1.1 Demi-plan de Poincaré . . . 98
6.1.2 Disque de Poincaré . . . 99
6.2 Étude des groupes . . . 99
6.2.1 Étude de $\Gamma(N)$ . . . 101
6.2.2 Étude de $\Gamma'_1(N)$ et de $\Gamma'_0(N)$ . . . 104
6.3 Parcours des éléments . . . 104
6.3.1 Parcours sur les domaines de $\Gamma(8)$, $\Gamma'_1(16)$ et $\Gamma'_0(42)$ . . . 105
6.3.2 Parcours sur des domaines construits à partir de copies de $\Gamma(2)$ 109
6.4 Conclusion . . . 111

7 Codes sur des quotients finis du groupe du triangle 113
7.1 Groupe du triangle et construction . . . 113
7.2 Étude des quotients du groupe du triangle . . . 117
7.3 Encodage du message . . . 122
7.3.1 Intervalles d'encodage . . . 122
7.3.2 Auto-intersections génériques . . . 123
7.3.3 Intervalles de longueur k sans répétitions . . . 125
7.3.4 Problèmes de précision . . . 127
7.4 Fonctions de transfert et propriétés . . . 127
7.5 Exemples de codage . . . 132
7.6 Décodage . . . 133
7.7 Reconnaissance de code et aspects cryptographiques . . . 134
7.8 Conclusion . . . 134

Conclusion et perspectives 137

A Représentations irréductibles du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ 139

B Fonctions de transfert optimales pour $G = \mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ **143**

C Fonctions de transfert optimales pour $G = D_6$ **161**

Communications et publications **165**

Bibliographie **170**

Table des figures

1.1 Chaîne de transmission numérique 3
1.2 Code convolutif de rendement $\frac{1}{n}$ 9

3.1 Encodage sur un groupe non-commutatif infini 38
3.2 Encodage sur un groupe non-commutatif fini 40
3.3 Décodage sur un groupe non-commutatif fini 41

4.1 Circuit d'encodage pour les codes convolutifs sur D_∞ 50
4.2 Exemple de circuit d'encodage avec $\tau = [0, 1, 1, 0, 1]$ 51
4.3 Exemple de circuit d'encodage avec $G_T = (011\ \ 010\ \ 011)$ 55
4.4 Exemple de l'algorithme de Viterbi adapté 57

5.1 Domaine fondamental de H 71
5.2 Longueur moyenne d'encodage en fonction de la pente de la droite pour $N = 7$ et $M = 3$ 73
5.3 Longueur moyenne d'encodage en fonction de la pente de la droite pour $N = 13$ et $M = 3$ 73
5.4 Longueur moyenne d'encodage en fonction de la pente de la droite pour $N = 11$ et $M = 5$ 73
5.5 Parcours de la droite de pente $M - 1 = 2,0000001$ avec $N = 7$ et $M = 3$ 74
5.6 Actions H et D 74
5.7 Comparaison entre les longueurs des intervalles d'encodage associées aux cases traversées par le système dynamique discret sur $\mathbb{Z}/7\mathbb{Z} \times \mathbb{Z}/3\mathbb{Z}$ (en bleu) et la probabilité qu'un intervalle d'encodage soit de longueur donnée en abscisse (en rouge) 77

6.1 Géodésiques sur le demi-plan de Poincaré 99
6.2 Géodésiques sur le disque de Poincaré 100
6.3 Domaine fondamental (en gris) de l'action du groupe modulaire sur $\mathbb{H}$ 100
6.4 Domaine fondamental de l'action de $\Gamma(2)$ sur $\mathbb{H}$ 101
6.5 Domaine fondamental de l'action de $\Gamma(4)$ sur $\mathbb{H}$ 103
6.6 Domaine fondamental de l'action de $\Gamma'_0(4)$ et de l'action de $\Gamma'_1(4)$ sur $\mathbb{H}$105
6.7 Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma(8)$ 106
6.8 Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_1(16)$ 106

6.9 Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_0(42)$ . . . 106
6.10 Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma(8)$ en supprimant les cycles . . . 108
6.11 Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_1(16)$ en supprimant les cycles . . . 108
6.12 Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_0(42)$ en supprimant les cycles . . . 108
6.13 Pourcentages de chaque longueur des intervalles d'encodage calculés sur un groupe $\Gamma(2)/H_1$ construit aléatoirement de cardinal 32 en supprimant les cycles . . . 110
6.14 Pourcentages de chaque longueur des intervalles d'encodage calculés sur un groupe $\Gamma(2)/H_2$ construit aléatoirement de cardinal 32 en supprimant les cycles . . . 110
6.15 Pourcentages de chaque longueur des intervalles d'encodage calculés sur un groupe $\Gamma(2)/H_3$ construit aléatoirement de cardinal 32 en supprimant les cycles . . . 112

7.1 Triangle hyperbolique et réflexions σ_1, σ_2 et σ_3 . . . 114
7.2 Groupe du triangle $T(2,3,7)$ . . . 114
7.3 Points p_1, p_2, p_3 sur le disque de Poincaré . . . 115
7.4 Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré pour $G = S_4$ . . . 119
7.5 Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré avec les identifications des côtés pour $G = S_4$ . . . 120
7.6 Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré pour $G = D_6$ . . . 121
7.7 Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré avec les identifications des côtés pour $G = D_6$ . . . 122
7.8 Longueur moyenne des intervalles d'encodage en fonction de la pente pour $G = S_4$ . . . 123
7.9 Exemple et contre-exemple d'une auto-intersection générique . . . 124
7.10 Longueur moyenne des intervalles d'encodage en fonction de la pente en ne tenant pas compte des auto-intersections génériques pour $G = S_4$ 124

Liste des tableaux

4.1 Ordre naturel des monômes de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ 48
4.2 La redondance (en gras) se place à gauche et à droite 49
4.3 Codage de u avec le circuit associé à G_T 55
4.4 Distance libre maximale et nombre de matrices de transfert atteignant ce maximum pour des codes de longueur de mémoire et de rendement donnés 58
4.5 Exemple de la longueur de la redondance pour $n = 2$ 61

5.1 Calcul de $\frac{1}{2100}\sum_{i=0}^{2099}\phi_i(a,a+e)$ en fonction des écarts e entre les pentes des droites 75
5.2 Calcul de $\sum_{i=0}^{2099}\frac{\phi_i((x_1,y_1),(x_2,y_2))}{2100}$ en fonction des écarts entre (x_1,y_1) et (x_2,y_2) 76
5.3 Calcul des occurrences des éléments g dans une suite de 2100 éléments parcourus sur le groupe $\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}$ 78
5.4 Nombre moyen de cases à parcourir pour avoir 100 intervalles de longueur k 80
5.5 Comparaison des distances minimales des codes linéaires en bloc connus avec celles des codes convolutifs sur $\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}$ 87
5.6 Nombre d'ensembles E atteignant telle distance sur $\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}$ . . 88
5.7 Distance moyenne atteinte par les fonctions de transfert τ optimales pour tous les ensembles E de longueur k 89

6.1 Liens entre les arêtes de $\Delta(2)$ et celles de $\Delta(4)$ 103
6.2 Nombre d'éléments du quotient considéré placés dans les intervalles d'encodage si on parcourt 32 000 faces du domaine 109

7.1 Identifications des côtés du domaine pour $G = S_4$ 119
7.2 Identifications des côtés du domaine pour $G = D_6$ 121
7.3 Nombre moyen de cases à parcourir pour avoir 100 intervalles de longueur k pour $G = D_6$ et $G = S_4$ 126
7.4 Calcul de $\sum_{i=0}^{2399}\frac{\phi_i(p_1,p_2)}{2400}$ en fonction des écarts entre les pentes des géodésiques 127
7.5 Calcul des occurrences des éléments g dans une suite de 2400 éléments parcourus sur le groupe S_4 128
7.6 Comparaison des distances minimales des codes linéaires en bloc connus avec celles des codes convolutifs sur D_6 129
7.7 Nombre d'ensembles E atteignant telle distance sur D_6 129

7.8 Distance moyenne atteinte par les fonctions de transfert τ optimales pour tous les ensembles E de longueur k 130
7.9 Comparaison entre la distance moyenne approchée des codes sur S_4, la borne inférieure de la distance minimale maximale pour les codes sur S_4 et la distance minimale des codes linéaires en bloc 131
7.10 Distance moyenne et rendement approchés pour chaque type de parcours du domaine de S_4 131

A.1 Représentations irréductibles du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ 140
A.2 Représentations irréductibles du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ sur $\mathbb{F}_{64}$ 142

Liste des Algorithmes

5.1 Calcul de la moyenne des longueurs des intervalles d'encodage en fonction de la pente 72
5.2 Calcul d'intervalles d'encodage de longueur k sur le groupe $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ 79
5.3 Calcul de la matrice de parité du code défini par un intervalle d'encodage E et une fonction de transfert τ sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ 84
6.1 Calcul des longueurs des intervalles d'encodage 105
7.1 Calcul d'intervalles d'encodage de longueur k sur un groupe fini G . . 125
7.2 Calcul de la distance moyenne approchée des codes sur S_4 en fonction de k la longueur des intervalles d'encodage 130

Notations

τ	la fonction de transfert d'un code
G	le groupe sur lequel la convolution est définie
G_T	la matrice génératrice d'un code
H	la matrice de parité d'un code
c	le mot de code émis
r	le mot de code reçu
$Hom(G, G')$	l'ensemble des homomorphismes (ou morphismes) du groupe G vers le groupe G'
$End(G)$	l'ensemble des endomorphismes du groupe G
$Aut(G)$	le groupe des automorphismes du groupe G
$Card(G)$	le cardinal du groupe G
$Ker(f)$	le noyau de l'application f
$Im(f)$	l'image de l'application f
$Tr(f)$	la trace de l'application f
$car(K)$	la caractéristique de l'anneau K

Propres au chapitre 4

u	le message à encoder
k	la longueur de u
n	le nombre de sorties du codeur
$\frac{1}{n}$	le rendement des codes
m	la taille de la mémoire du registre à décalage et la longueur de τ
d_f	la distance libre des codes

Propres aux chapitres 5, 6 et 7

u	le bloc de message à encoder
k	la longueur de u
n	la longueur du mot de code et le cardinal de G
$\frac{k}{n}$	le rendement des codes en bloc
E	le sous-ensemble de G sur lequel on va encoder le message, appelé intervalle d'encodage
X^E	l'ensemble des applications de l'ensemble E dans l'ensemble X.

Introduction

Contexte

Depuis la fin des années 1950 et le début des premiers réseaux de télécommunications, les transmissions numériques assurent la communication entre deux ordinateurs. Les ordinateurs s'échangent des suites de « 0 » et de « 1 » appelée information numérique. Cette information transite par un canal de transmission qui est soumis à des perturbations. A cause de ces dernières, des erreurs apparaissent dans l'information transmise. Pour assurer l'intégrité de l'information, l'émetteur encode l'information en un mot de code, c'est-à-dire ajoute quelques bits de redondance au message, qui donnent une information sur le message en lui-même. Ces bits permettent au récepteur de détecter et/ou corriger les erreurs survenues lors de la transmission.

Cette opération d'encodage n'a pas pour objectif d'empêcher une tierce personne d'intercepter le message et de le comprendre. Pour effectuer cela, on utilise la cryptologie, c'est-à-dire qu'avant l'opération d'encodage, on chiffre l'information à transmettre et c'est l'information chiffrée qui va être encodée pour la transmission. Le récepteur corrige les erreurs sur le mot de code chiffré puis déchiffre le message s'il possède la clé du système de chiffrement.

Dans cette thèse, nous nous sommes intéressés à des codes correcteurs particuliers, les codes convolutifs, dont la définition et les propriétés sont rappelées dans le chapitre 1. Ces codes convolutifs sont très performants et très utilisés dans les télécommunications actuelles. Ces codes sont basés sur une opération mathématique simple, la convolution. Dans le cas d'un code classique, cette convolution s'effectue dans un groupe particulier, le groupe des entiers relatifs $\mathbb{Z}$. Néanmoins, l'opération de convolution peut être effectuée dans tous les groupes. Nous nous sommes donc intéressés aux codes convolutifs que l'on pourrait construire avec d'autres groupes. Notre idée est de choisir des groupes non-commutatifs et de faire de l'encodage qui varie dans le temps. Ainsi, nous espérons obtenir des codes avec des propriétés cryptographiques, avec lesquels il serait difficile, voire impossible, pour une tierce personne de retrouver le message transmis à partir du train de bits intercepté, si elle ne possède pas certaines données de l'encodage.

Plan

Dans cette thèse, nous introduirons tout d'abord les codes correcteurs d'erreurs, en bloc, puis convolutifs. Nous verrons leurs opérations d'encodage, de décodage

ainsi que leurs propriétés. La reconnaissance de ces codes sera brièvement expliquée.

Dans le second chapitre, nous rappellerons quelques notions de théorie des groupes, notamment les actions de groupes et nous ferons le lien avec les fonctions définies sur les groupes que nous allons étudier par la suite. Nous présenterons également un critère pour déterminer si une fonction à valeurs dans $\mathbb{F}_2$ est un diviseur de zéro ou non sur un groupe fini.

Puis, dans le troisième chapitre, nous poserons les bases des codes convolutifs définis sur des groupes non-commutatifs finis ou infinis.

Ensuite, nous présenterons dans le quatrième chapitre, les codes convolutifs sur le groupe diédral infini. L'algèbre de ce groupe sera étudiée en détails afin de déterminer les bonnes fonctions de transfert des codes. Puis nous verrons que l'on peut encoder le message grâce à un circuit électronique avec un registre à décalage, légèrement différent du circuit des codes convolutifs classiques. Pour le décodage de ces codes, un algorithme de Viterbi adapté sera présenté. Enfin, nous montrerons que les propriétés de ces codes sont identiques aux propriétés des codes convolutifs classiques, avec, dans certains cas, plus de codes optimaux. Néanmoins, ces codes ne permettent pas d'avoir les propriétés cryptographiques que nous recherchons.

Pour espérer avoir ces propriétés, nous allons faire du codage en bloc dans les trois chapitres suivants. Le chapitre 5 présentera les codes convolutifs en bloc définis sur le produit semi-direct $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$. Après rappel de la théorie de Fourier, nous verrons un critère basé sur la transformée de Fourier pour décider si une fonction peut être utilisée comme fonction de transfert d'un code. Ensuite, nous présenterons le parcours chaotique des éléments du groupe afin d'avoir un codage variable dans le temps. Enfin, les propriétés de distance et cryptographiques seront présentées.

Puis nous verrons dans le chapitre 6, une étude des sous-groupes de congruence $\Gamma(N)$, $\Gamma'_0(N)$ et $\Gamma'_1(N)$ dans laquelle nous expliquerons pourquoi nous n'avons pas pu construire des codes de rendement suffisant sur ces groupes.

Enfin, des codes sur des quotients du groupe du triangle seront présentés dans le chapitre 7. Le parcours chaotique des éléments sur le plan hyperbolique sera expliqué. Puis, nous verrons des exemples, les groupes S_4 et D_6 sur lesquels nous étudierons la distance minimale des codes tout en la comparant à la distance des codes linéaires en bloc. Enfin nous présenterons les propriétés cryptographiques de ces codes.

Chapitre 1

Les codes convolutifs

Dans ce chapitre, nous allons tout d'abord rappeler dans quel contexte s'inscrivent les codes correcteurs d'erreurs, puis les définitions et propriétés des codes en bloc et des codes convolutifs. Enfin, nous présentons le principe de la reconnaissance des codes. Les définitions de ce chapitre sont tirées de [Neu07, Moo05].

1.1 Contexte

1.1.1 Chaîne de transmission

Pour transmettre l'information numérique d'un point A à un point B, on utilise une chaîne de transmission numérique. L'information qui transite sur cette chaîne est sous la forme d'une suite binaire, c'est-à-dire des 0 et des 1. Le support physique qui permet de transmettre cette information peut être soit un câble, soit l'air dans le cas du WiFi, soit l'espace dans le cas des satellites. Le schéma 1.1 représente une telle chaîne de transmission.

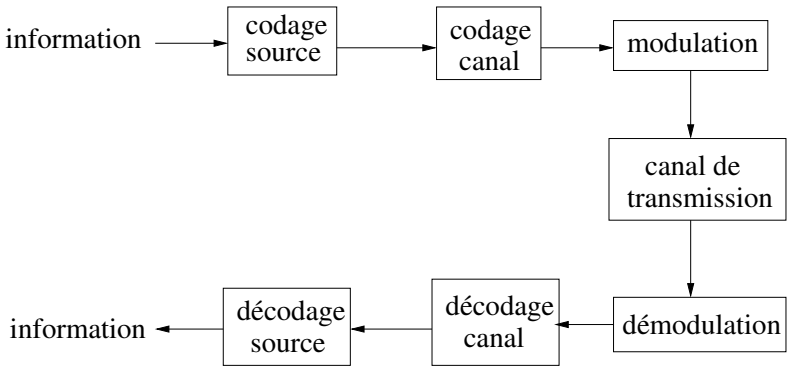


FIGURE 1.1 – Chaîne de transmission numérique

Chaque bloc du schéma a un rôle bien spécifique dont l'objectif global est la fiabilité et la rapidité de la transmission. La première étape, notée codage de source, consiste à compresser les données et ainsi augmenter l'efficacité de la transmission. La deuxième étape, le codage de canal, ajoute de la redondance au message afin de

pouvoir éventuellement corriger les erreurs qui surviendront au cours de la transmission. La troisième étape est la modulation qui transforme l'information numérique en un signal physique adapté au support qui va assurer le transport. Le destinataire réalise les opérations en sens inverse à la réception du signal afin de retrouver le message envoyé.

Si l'expéditeur souhaite, de plus, protéger son message contre des personnes qui souhaiteraient le lire alors qu'elles ne sont pas destinataires, il va chiffrer son message avec un algorithme de chiffrement. La sécurité de cet algorithme sera assurée par le caractère secret de la clé de chiffrement. Cette opération sera réalisée entre le codage de source et le codage de canal.

1.1.2 Codage de canal

Le codage de canal permet de protéger l'information des perturbations qui produisent des erreurs sur le train binaire dans le canal de transmission. Pour réaliser ceci, l'idée est de transformer le message en un mot de code, plus long, dont la redondance ajoutée permettra de corriger ces erreurs.

On distingue deux grandes familles de codes correcteurs d'erreurs, les codes en bloc et les codes convolutifs. Dans le premier cas, l'information est découpée en blocs de k bits. Ces k bits sont transformés en un bloc de n bits (avec $n > k$) qui est le mot de code. Dans le deuxième cas, les n bits en sortie du codeur dépendent des k bits en entrée mais également d'autres bits introduits précédemment. Nous allons voir plus précisément ces deux familles de codes en commençant par les codes en bloc.

1.2 Codes correcteurs d'erreurs en bloc

L'alphabet avec lequel nous allons travailler sera le corps à deux éléments $\mathbb{F}_2 = \{0, 1\}$ sur lequel les messages et les mots de code seront définis.

Définition 1.1. *Un (n, k)* ***code en bloc*** *C sur $\mathbb{F}_2$ est un ensemble de 2^k vecteurs de taille n appelés mots de code. Chaque message u de taille k est associé de façon unique à un mot de code de taille n.*

Cette correspondance entre les messages et les vecteurs pourraient être représentée par une liste exhaustive, mais pour un k assez grand, le stockage serait compliqué. La complexité peut être réduite en imposant une structure mathématique au code. La structure la plus courante est la linéarité du code.

Définition 1.2. *Un code en bloc C sur le corps $\mathbb{F}_2$ de longueur n (avec 2^k mots de code) est un (n, k) code* ***linéaire*** *si et seulement si l'ensemble des mots de code forme un sous-espace vectoriel de dimension k sur $\mathbb{F}_2^n$. Le nombre n est appelé la longueur du code et le nombre k est la dimension du code. Le rendement du code est $\frac{k}{n}$.*

Dans un code linéaire, toute combinaison linéaire de mots de code est un mot de code.

Définition 1.3. *Le* ***poids de Hamming*** $w(c)$ *d'un mot de code* c *est le nombre d'éléments non nuls de* c*. Le poids minimal* w_{min} *d'un code* C *est le plus petit poids de Hamming de tous les mots de code non nuls* $w_{min} = \min_{c \in C, c \neq 0} w(c)$*.*

Définition 1.4. *La* ***distance de Hamming*** *entre deux séquences* $x = (x_0, \ldots, x_{n-1})$ *et* $y = (y_0, \ldots, y_{n-1})$ *est le nombre de positions où les éléments diffèrent entre les deux séquences.*

$$d_H(x, y) = \sum_{i=0}^{n-1} [x_i \neq y_i]$$

avec $[x_i \neq y_i] = 1$ *si et seulement si* $x_i \neq y_i$ *et 0 sinon.*

Définition 1.5. *La* ***distance minimale*** d_{min} *d'un code* C *est la plus petite distance de Hamming entre n'importe quelle paire de mots de code.*

$$d_{min} = \min_{c_i, c_j \in C, c_i \neq c_j} d_H(c_i, c_j)$$

Théorème 1.1. *Pour un code linéaire* C*, la distance minimale* d_{min} *satisfait* $d_{min} = w_{min}$*.*

Démonstration. Le résultat est relié au fait que toute combinaison linéaire de mots de code est un mot de code. Si c_i et c_j sont des mots de code alors $c_i - c_j$ en est également un. Le calcul de la distance peut être « translaté à l'origine ».

$$d_{min} = \min_{c_i, c_j \in C, c_i \neq c_j} d_H(c_i, c_j) = \min_{c_i, c_j \in C, c_i \neq c_j} d_H(c_i - c_j, c_j - c_j) = \min_{c \in C, c \neq 0} w(c)$$

□

Un (n, k) code ayant une distance minimale d_{min} peut être appelé un (n, k, d_{min}) code.

Théorème 1.2. *Le nombre d'erreurs que l'on peut corriger dans un* (n, k, d_{min}) *code est* $\left\lfloor \frac{d_{min}-1}{2} \right\rfloor$*. Le nombre d'erreurs que l'on peut détecter dans un* (n, k, d_{min}) *code est* $d_{min} - 1$*.*

Démonstration. Preuve réalisée dans la section 1.8.1 de [Moo05]. □

Comme un code linéaire en bloc est un espace vectoriel de dimension k, il existe k vecteurs linéairement indépendants $g_{T0}, g_{T1}, \ldots, g_{Tk-1}$ tels que chaque mot de code $c \in C$ peut être représenté comme une combinaison linéaire de ces vecteurs,

$$c = u_0 g_{T0} + u_1 g_{T1} + \ldots + u_{k-1} g_{Tk-1}$$

où $u_i \in \mathbb{F}_2$. Avec ces vecteurs, on peut former la matrice G_T de taille $k \times n$ comme suit :

$$G_T = \begin{pmatrix} g_{T0} \\ g_{T1} \\ \vdots \\ g_{Tk-1} \end{pmatrix}$$

Soit $u = (u_0, u_1, \ldots, u_{k-1})$. Alors on peut écrire :

$$c = uG_T$$

Comme les lignes de G_T génèrent le (n, k) code linéaire C, G_T est appelée la **matrice génératrice** de C, et l'opération $c = uG_T$ est l'opération d'encodage d'un message u en un mot de code c.

Cette représentation du code par cette matrice G_T n'est pas unique. Étant donné une matrice G_T, une autre matrice G'_T peut être obtenue en effectuant des combinaisons linéaires de ses lignes. Alors l'opération d'encodage définie par $c = uG'_T$ envoie le message u sur un mot de code de C, mais ce n'est pas nécessairement le même mot de code que l'on aurait obtenu en utilisant la matrice génératrice G_T.

Définition 1.6. *Soit C un (n, k) code en bloc. Un encodeur est dit **systématique** si les symboles du message $u_0, u_1, \ldots, u_{k-1}$ peuvent être trouvés explicitement et de façon inchangée dans le mot de code. Cela signifie qu'il existe des coordonnées $i_0, i_1, \ldots, i_{k-1}$ telles que $c_{i_0} = u_0, c_{i_1} = u_1, \ldots, c_{i_{k-1}} = u_{k-1}$. Pour un code linéaire, le générateur d'un encodeur systématique est appelé un générateur systématique.*

Il est évident qu'être systématique est une propriété de l'encodeur et non du code. Pour un code en bloc linéaire, l'opération d'encodage représentée par G_T est systématique si une matrice identité peut être identifiée parmi les lignes de G_T. Un générateur systématique est souvent écrit sous la forme

$$G_T = (I_k \ P) = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & p_{0,0} & p_{0,1} & \cdots & p_{0,n-k-1} \\ 0 & 1 & 0 & \ldots & 0 & p_{1,0} & p_{1,1} & \cdots & p_{1,n-k-1} \\ 0 & 0 & 1 & \ldots & 0 & p_{2,0} & p_{2,1} & \cdots & p_{2,n-k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} \end{pmatrix}$$

où I_k est la matrice identité de taille $k \times k$ et P est une matrice de taille $k \times (n-k)$ qui génère les symboles de redondance (ou de parité). L'opération d'encodage est :

$$c = u(I_k \ P) = (u \ uP)$$

Le mot de code est divisé en deux parties : la partie u consiste en les symboles du message et la partie uP consiste en les symboles de parité.

Effectuer des opérations linéaires sur les lignes du générateur permet de produire le même code. Si deux colonnes du générateur sont interchangées, alors la position correspondante du code change mais la distance du code est préservée.

Définition 1.7. *Deux codes linéaires qui sont les mêmes à permutation des éléments près sont dits **équivalents**.*

Soient G_T et G'_T deux matrices génératrices de deux codes équivalents. Alors G_T et G'_T sont reliées par les opérations suivantes :

— permutations des colonnes

— opérations élémentaires sur les lignes

Étant donné une matrice génératrice G_T, il est possible de la mettre sous la forme systématique en utilisant le pivot de Gauss.

Comme un code linéaire est un sous-espace vectoriel de $\mathbb{F}_2^n$ de dimension k, il y a un espace dual à C de dimension $n-k$.

Définition 1.8. *L'espace dual d'un (n,k)-code C de dimension k est le $(n, n-k)$ code dual de C, noté $C^\perp$.*

Comme tout espace vectoriel, $C^\perp$ a une base que l'on note $(h_0, h_1, \ldots, h_{n-k-1})$. On forme une matrice H en utilisant ces vecteurs de base comme lignes :

$$H = \begin{pmatrix} h_0 \\ h_1 \\ \vdots \\ h_{n-k-1} \end{pmatrix}$$

Cette matrice est appelée la **matrice de parité** du code C. La matrice génératrice et la matrice de parité d'un code vérifient :

$$G_T {}^tH = 0$$

La matrice de parité a la propriété importante suivante :

Théorème 1.3. *Soit C un (n,k)-code linéaire sur $\mathbb{F}_2$ et soit H la matrice de parité de C. Un vecteur $v \in \mathbb{F}_2^n$ est un mot de code si et seulement si*

$$v^t H = 0$$

Démonstration. Soit $c \in C$. Par la définition du code dual, on a $c.{}^th = 0$ pour tout $h \in C^\perp$. Chaque vecteur ligne $h \in C^\perp$ peut être écrit comme $h = xH$ pour un vecteur x. Comme x est arbitraire, et peut en fait sélectionner des lignes individuelles de H, on doit avoir $c\,{}^th_i = 0$ pour $i = 0, 1, \ldots, n-k-1$, donc $c\,{}^tH = 0$.

Inversement, supposons que $v^tH = 0$. Alors $v^th_i = 0$ pour $i = 0, 1, \ldots, n-k-1$, donc v est orthogonal à une base du code dual et donc est orthogonal au code dual lui-même. Donc v doit être dans le code C. □

Quand G_T est sous forme systématique, une matrice de parité est facilement déterminée par :

$$H = \begin{pmatrix} {}^tP & I_{n-k} \end{pmatrix} \tag{1.1}$$

Souvent, une matrice de parité pour un code est obtenue en calculant une matrice génératrice sous forme systématique et en utilisant la formule 1.1.

Théorème 1.4. *Soit un code en bloc linéaire C qui a une matrice de parité H. La distance minimale d_{min} de C est égale au plus petit nombre de colonnes de H qui sont linéairement dépendantes. Autrement dit, toutes les combinaisons de $d_{min}-1$ colonnes sont linéairement indépendantes, donc il existe un ensemble de d_{min} colonnes qui sont linéairement dépendantes.*

Démonstration. Désignons les colonnes de H par $h_0, h_1, \ldots, h_{n-1}$. Alors comme $c^t H = 0$ pour n'importe quel mot de code c, on a :

$$0 = c_0 h_0 + c_1 h_1 + \ldots + c_{n-1} h_{n-1}$$

Soit c le mot de code de plus petit poids $w = w(c) = d_{min}$ avec les éléments non nuls positionnés aux indices $i_1, i_2, \ldots, i_w$. Alors

$$c_{i_1} h_{i_1} + c_{i_2} h_{i_2} + \ldots + c_{i_w} h_{i_w} = 0$$

Clairement, les colonnes de H correspondant aux éléments de c sont linéairement dépendantes. D'un autre coté, s'il y avait un ensemble linéairement dépendant de $z < w$ colonnes de H alors il y aurait un mot de code de poids z. □

Définition 1.9. *Soit r un vecteur de taille n sur $\mathbb{F}_2$ et soit H la matrice de parité d'un code C. Le vecteur*

$$s = r^t H$$

est appelé le syndrome de r.

Par le théorème 1.3, $s = 0$ si et seulement si r est un mot de code de C. Le syndrome s aide à diagnostiquer si r est un mot de code ou a été corrompu par le bruit. Comme nous allons le voir, ça aide aussi à déterminer quelle est l'erreur.

Le syndrome peut être utilisé comme un schéma de détection d'erreur. Supposons qu'un mot de code c dans un code en bloc linéaire C sur $\mathbb{F}_2$ est transmis et que le vecteur r de taille n est reçu. On peut écrire :

$$r = c + e$$

dans $\mathbb{F}_2$ et où e est le vecteur d'erreur. Le vecteur reçu r peut être n'importe lequel des vecteurs de $\mathbb{F}_2^n$ donc tous les vecteurs d'erreurs sont possibles. Soit H la matrice de parité du code C. Alors le syndrome

$$s = r^t H = (c + e)^t H = e^t H$$

Avec le théorème 1.3, nous en déduisons que $s = 0$ si r est un mot de code. Cependant, si $s \neq 0$, alors une erreur est détectée, mais on ne sait pas sur quel bit elle se situe.

Le syndrome peut également être utilisé pour corriger les erreurs. Pour cela, il faut chercher le plus petit ensemble $I \subset \{0, 1, \ldots, n-1\}$ tel que

$$\sum_{i \in I} h_i = s$$

avec h_i les colonnes de la matrice de parité H du code C. Ainsi le mot de code le plus proche de r est

$$r + \sum_{i \in I} e_i$$

où e_i est le vecteur constitué de 1 en position i et de zéros partout ailleurs.

On peut trouver une table des meilleurs codes linéaires en termes de distance minimale en [Gra07].

1.3 Codes convolutifs

Les codes convolutifs ont été introduits par Peter Elias [Eli55] en 1955. Ce sont des codes dont les n bits de sortie dépendent des k bits d'entrée mais également d'autres bits précédents du message. Nous allons nous intéresser ici uniquement aux codes de rendement $\frac{1}{n}$, car ce sont les plus simples et les plus utilisés en pratique.

1.3.1 Codage

1.3.1.1 D'un point de vue électronique

Les codes convolutifs de rendement $\frac{1}{n}$ peuvent être représentés par la figure 1.2. Les bits u_i du message entrent les uns à la suite des autres dans un registre à décalage

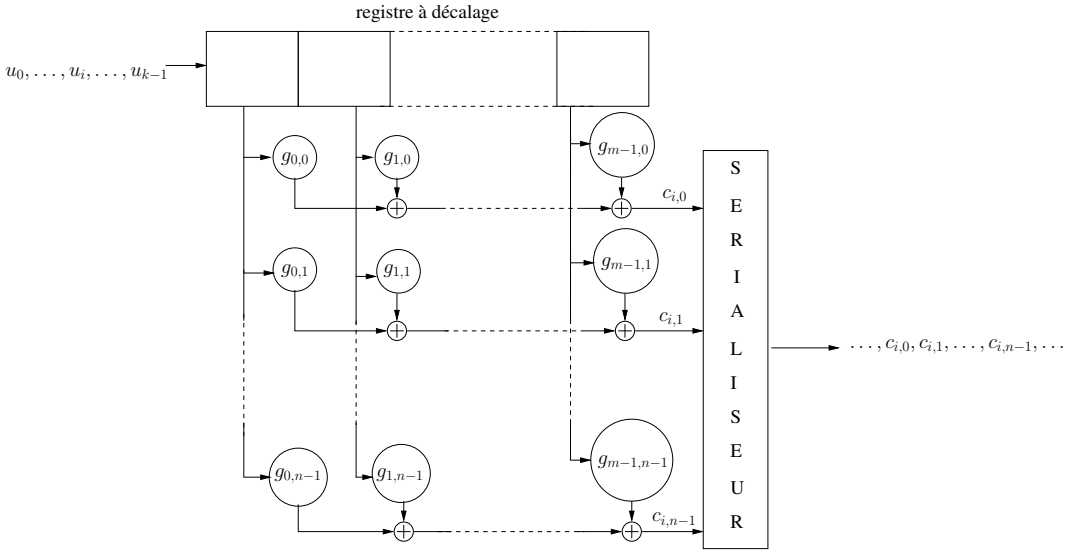
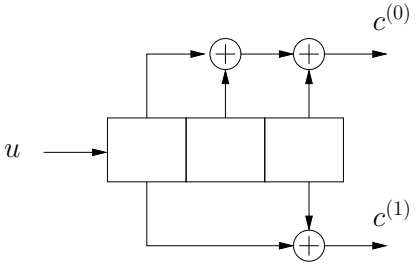


FIGURE 1.2 – Code convolutif de rendement $\frac{1}{n}$

de longueur m. Les bits de la mémoire de ce registre sont xorés ou non selon la valeur des $g_{i,j}$. Nous obtenons n bits de sortie qui sont sérialisés pour former n bits du mot de code. Donc, à un bit entrant dans le registre correspond n bits en sortie du codeur.

Exemple 1.1. *Nous allons étudier le circuit de l'exemple ci-dessous qui a un rendement de* $\frac{1}{2}$.



Soit $u = (1, 1, 0, 1, 0, 0, \ldots)$. *La sortie* $c^{(0)}$ *de l'encodeur sera* $c^{(0)} = (1, 0, 0, 0, 1, 1, \ldots)$ *et la sortie* $c^{(1)} = (1, 1, 1, 0, 0, 1, \ldots)$. *Après le sérialiseur (aussi appelé multiplexage)*

le mot de code sera $c = (1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, \ldots)$. *Dans cet exemple, on a :*

$$g_{0,0} = 1, g_{1,0} = 1, g_{2,0} = 1 \Rightarrow g^{(0)} = (1, 1, 1)$$
$$g_{0,1} = 1, g_{1,1} = 0, g_{2,1} = 1 \Rightarrow g^{(1)} = (1, 0, 1)$$

Ainsi chaque sortie peut être déduite des $g_{i,j}$ *telle que*

$$c_i^{(0)} = \sum_{l=0}^{m-1} u_{i-l} g_{l,0} \leftrightarrow c^{(0)} = u * g^{(0)}$$
$$c_i^{(1)} = \sum_{l=0}^{m-1} u_{i-l} g_{l,1} \leftrightarrow c^{(1)} = u * g^{(1)}$$

Donc de manière générale, chaque sortie de l'encodeur est :

$$c_{i,j} = \sum_{l=0}^{m-1} u_{i-l} g_{l,j}$$
$$\Rightarrow c^{(j)} = u * g^{(j)}$$

Chaque sortie de l'encodeur est donc la convolution entre le message u et les « lignes du circuit » $g^{(j)}$. D'où le nom de codes convolutifs. Nous allons nous limiter aux codes convolutifs dont la mémoire est remplie de zéro avant encodage et dont l'encodage s'arrête lorsque la mémoire est revenue à son point de départ. Pour les autres types de codes, on peut se référer à [Neu07, Moo05].

Comme pour les codes en bloc, l'opération d'encodage peut être décrite comme la multiplication du message avec une matrice génératrice $c = uG_T$. Cependant, le message u et le mot de code c peuvent être de taille infinie. Donc la matrice génératrice aura aussi une structure infinie. Elle est construite à partir des matrices de taille $1 \times n$ suivantes :

$$G_i = \begin{pmatrix} g_{i,0} & g_{i,1} & \ldots & g_{i,n-1} \end{pmatrix}$$

La matrice génératrice est alors :

$$G_T = \begin{pmatrix} G_0 & G_1 & \ldots & G_{m-1} & 0 & 0 & \ldots \\ 0 & G_0 & G_1 & \ldots & G_{m-1} & 0 & \ldots \\ 0 & 0 & G_0 & G_1 & \ldots & G_{m-1} & \ldots \\ 0 & 0 & 0 & \ddots & \ddots & \ddots & \ddots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{pmatrix}$$

Exemple 1.2. *Revenons à l'exemple précédent. On a :*

$$G_0 = (g_{0,0}\ g_{0,1}) = (1\ 1)$$
$$G_1 = (g_{1,0}\ g_{1,1}) = (1\ 0)$$
$$G_2 = (g_{2,0}\ g_{2,1}) = (1\ 1)$$

Donc la matrice génératrice G_T est :

$$G_T = \begin{pmatrix} 11 & 10 & 11 & 00 & 00 & \dots \\ 00 & 11 & 10 & 11 & 00 & \dots \\ 00 & 00 & 11 & 10 & 11 & \dots \\ 00 & 00 & 00 & \ddots & \ddots & \ddots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots \end{pmatrix}$$

Ainsi l'encodage s'effectue comme :

$$\begin{aligned} c = uG_T &= (1,1,0,1,0,0,\dots) \begin{pmatrix} 11 & 10 & 11 & 00 & 00 & \dots \\ 00 & 11 & 10 & 11 & 00 & \dots \\ 00 & 00 & 11 & 10 & 11 & \dots \\ 00 & 00 & 00 & \ddots & \ddots & \ddots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots \end{pmatrix} \\ &= (11,01,01,00,10,11,00,\dots) \end{aligned}$$

1.3.1.2 D'un point de vue mathématique

Chaque message u et mot de code c peuvent être vus différemment, comme des polynômes (avec l_u la longueur de u et l_c la longueur de c), c'est-à-dire :

$$u(X) = u_0 + u_1 X + u_2 X^2 + \dots + u_{l_u-1} X^{l_u-1} = \sum_{i=0}^{l_u-1} u_i X^i$$

$$c(X) = c_0 + c_1 X + c_2 X^2 + \dots + c_{l_c-1} X^{l_c-1} = \sum_{i=0}^{l_c-1} c_i X^i$$

De plus, les $g^{(j)}$ peuvent également être vus comme des polynômes en X. Ainsi l'opération d'encodage est

$$\left(c^{(0)}(X)\ c^{(1)}(X)\ \dots\ c^{(n-1)}(X)\right) = u(X)G_T(X)$$

où $G_T(X) = \left(g^{(0)}(X)\ g^{(1)}(X)\ \dots\ g^{(n-1)}(X)\right)$ est la matrice génératrice du code convolutif de taille $1 \times n$.

Définition 1.10. *On définit ainsi un* ***code convolutif*** *comme un sous-$\mathbb{F}_2[X]$-module de $\mathbb{F}_2[X]^n$ engendré par la matrice génératrice $G_T(X)$ de $\mathbb{F}_2[X]^n$.*

Exemple 1.3. *Reprenons l'exemple précédent. On a*

$$G_T(X) = \left(1 + X + X^2 \quad 1 + X^2\right)$$

et

$$\begin{aligned} \left(c^{(0)}(X) \quad c^{(1)}(X)\right) &= \left(1 + X + X^3\right)\left(1 + X + X^2 \quad 1 + X^2\right) \\ &= \left(1 + X^4 + X^5 \quad 1 + X + X^2 + X^5\right) \end{aligned}$$

L'opération de multiplexage est :

$$\begin{aligned} c(X) &= c^{(0)}\left(X^2\right) + Xc^{(1)}\left(X^2\right) \\ &= \left(1 + X^8 + X^{10}\right) + X\left(1 + X^2 + X^4 + X^{10}\right) \\ &= 1 + X^8 + X^{10} + X + X^3 + X^5 + X^{11} \end{aligned}$$

De manière générale, l'opération de multiplexage est réalisée par :

$$c(X) = \sum_{i=0}^{n-1} X^i c^{(i)}(X^n)$$

Cet encodage qui est un produit « polynôme-matrice de polynômes » suivi d'un multiplexage peut être remplacé par une multiplication de deux polynômes. Pour cela, il faut conserver l'information contenue dans la matrice génératrice $G_T(X)$ dans ce polynôme. Nous allons noter ce polynôme τ et l'appeler la **fonction de transfert** du code. Il est défini comme

$$\begin{aligned} \tau(X) &= g^{(0)}(X^n) + Xg^{(1)}(X^n) + \ldots + X^{n-1}g^{(n-1)}(X^n) \\ &= g_{0,0} + g_{1,0}X^n + \ldots + g_{m-1,0}X^{n(m-1)} \\ &+ g_{0,1}X + g_{1,1}X^{n+1} + \ldots + g_{m-1,1}X^{n(m-1)+1} \\ &\vdots \\ &+ g_{0,n-1}X^{n-1} + g_{1,n-1}X^{2n-1} + \ldots + g_{m-1,n-1}X^{n(m-1)+n-1} \end{aligned}$$

Dans ce polynôme τ, on retrouve tous les bits $g_{i,j}$, chacun devant un monôme différent. L'information de la matrice de transfert G_T est donc bien conservée.

L'opération d'encodage avec τ s'effectue ainsi :

$$c(X) = u(X^n)\tau(X)$$

Exemple 1.4. *Reprenons l'exemple précédent. On avait :*

$$G_T(X) = \begin{pmatrix} 1 + X + X^2 & 1 + X^2 \end{pmatrix}$$

Donc

$$\begin{aligned} \tau(X) &= (1 + X^2 + X^4) + X(1 + X^4) \\ &= 1 + X^2 + X^4 + X + X^5 \end{aligned}$$

Et le mot de code est :

$$\begin{aligned} c(X) &= u(X^2)\tau(X) \\ &= (1 + X^2 + X^6)(1 + X + X^2 + X^4 + X^5) \\ &= 1 + X + X^3 + X^5 + X^8 + X^{10} + X^{11} \end{aligned}$$

On retrouve bien le même mot de code que précédemment.

Les polynômes que nous utilisons ici pour l'encodage du message sont dans $\mathbb{F}_2[X]$. Cet anneau de polynômes est l'algèbre du groupe $\mathbb{Z}$. En effet, si on considère les coefficients polynomiaux de u et de τ, on peut les voir comme des fonctions :

$$\begin{aligned} u : \mathbb{Z} &\to \mathbb{F}_2 \\ i &\mapsto u_i \end{aligned} \qquad\qquad \begin{aligned} \tau : \mathbb{Z} &\to \mathbb{F}_2 \\ i &\mapsto \tau_i \end{aligned}$$

Ainsi l'encodage se ramène à l'opération suivante :

$$c_i = \sum_{j \in \mathbb{Z}} u_j \tau_{i-j}$$

Cette opération est la convolution de u et de τ sur le groupe des entiers relatifs $\mathbb{Z}$.

1.3.2 Décodage

Quand les codes convolutifs ont été découvert en 1955 par Elias, il n'y avait pas d'algorithme efficace de décodage. Il a fallu attendre 1957 avec l'algorithme de décodage séquentiel développé par Wozencraft [Woz57], amélioré ensuite en 1963 par Fano [Fan63]. L'algorithme le plus efficace pour les petites longueurs de mémoire m apparaîtra en 1967 avec Viterbi [Vit67]. Ces deux algorithmes sont des algorithmes dits de « décisions dures » : les bits forment le mot de code le plus probable. Un algorithme de décision souple a été développé en 1974 par Bahl, Cocke, Jelinek et Raviv [BCJR74] et est nommé BCJR, APP ou MAP.

Nous allons présenter plus en détails l'algorithme le plus efficace pour les petites longueurs de mémoire m : l'algorithme de Viterbi.

C'est un algorithme de décodage à maximum de vraisemblance utilisé dans le contexte d'un canal bruité sans mémoire. L'objectif d'un tel décodage est de minimiser la probabilité d'erreur entre le mot reçu qu'on notera r et le mot de code émis c. Minimiser cette probabilité, cela revient à maximiser la probabilité $P(r|c)$ qu'on ait reçu r sachant que c a été envoyé. Si cette probabilité est 1, alors $r = c$ et on a décodé. L'encodage d'un message par un code convolutif est un processus de Markov. On a une mémoire de taille m qui peut contenir 2^m valeurs possibles. A chaque instant t, on note x_t un des 2^m états de la mémoire au temps t. On suppose que le processus commence au temps 0 et termine au temps K et que les états initiaux et finaux, x_0 et x_K sont connus ; la séquence des états est alors représentée par un vecteur fini $x = (x_0, \ldots, x_K)$. Le processus est markovien dans le sens où la probabilité $P(x_{t+1}|x_0, x_1, \ldots, x_t)$ dépend uniquement de l'état x_t au temps t :

$$P(x_{t+1}|x_0, x_1, \ldots, x_t) = P(x_{t+1}|x_t)$$

On n'impose pas à cette probabilité de transition d'être fixe quelque soit t, elle peut varier dans le temps. A chaque étape de l'encodage, un bit du message entre dans le registre et fait évoluer x_t en x_{t+1}, on note ce bit f_t et il peut représenter la paire d'états (x_t, x_{t+1}). Ainsi, il y a une bijection entre les séquences d'états x et les séquences de transitions f. Chaque bit r_t du mot reçu r dépend uniquement des transitions f_t au temps t :

$$P(r|x) = P(r|f) = \prod_{t=0}^{K-1} P(r_t|f_t)$$

Le mot de code envoyé c est une fonction déterministe des états de la mémoire x donc :

$$P(r|c) = P(r|x) = P(r|f) = \prod_{t=0}^{K-1} P(r_t|f_t)$$

Pour estimer cette probabilité, on introduit une description du codage sous forme d'un treillis. Chaque nœud correspond à un état distinct x_t à un temps donné et chaque branche représente la transition f_t d'un état à un autre état au temps suivant. Le treillis commence et finit aux états connus x_0 et x_K. La propriété importante est qu'à chaque séquence d'états possibles x correspond un unique chemin à travers le treillis. Maximiser la probabilité $P(r|c)$ revient à minimiser l'opposé du logarithme de cette probabilité et on a ainsi :

$$-\ln P(r,c) = \sum_{t=0}^{K-1} -\ln(P(r_t|f_t))$$

Si on assigne à chaque branche, la longueur suivante :

$$\lambda(f_t) = -\ln(P(r_t|f_t))$$

Alors la longueur totale du chemin correspondant à un certain x est :

$$-\ln P(r,c) = \sum_{t=0}^{K-1} \lambda(f_t)$$

Le problème d'estimation de la probabilité est donc identique au problème de trouver le plus court chemin dans un graphe.

L'algorithme de Viterbi va donc permettre de trouver efficacement le plus court chemin dans ce graphe. Pour cela, on introduit les notations suivantes. On note x_0^t une séquence d'états $(x_0, x_1, \ldots, x_t)$. Dans le treillis, x_0^t correspond à un chemin commençant au nœud x_0 et terminant au nœud x_t. Pour n'importe quel nœud x_t, il y a généralement plusieurs chemins, chacun de longueur

$$\lambda(x_0^t) = \sum_{i=0}^{t-1} \lambda(f_i)$$

Le plus court de ces chemins est appelé le survivant du nœud x_t et est noté $\widehat{x}(x_t)$. A chaque instant $t > 0$, il y a M survivants, un pour chaque x_t. L'observation est la suivante : le chemin survivant complet $\widehat{x}$ doit commencer par un de ces survivants (si ce n'est pas le cas et qu'il est quand même passé par l'état x_t au temps t alors on peut remplacer ce segment initial par $\widehat{x}(x_t)$ pour avoir un chemin plus court, or le chemin survivant complet est déjà le plus court donc on aboutit à une contradiction). Donc à chaque instant t, on a besoin de se rappeler uniquement des M survivants $\widehat{x}(x_t)$ et de leur longueur $\lambda(\widehat{x}(x_t))$. Pour passer au temps $t+1$, on a seulement besoin d'étendre tous les survivants du temps t par une unité de temps, calculer la longueur de ces chemins étendus et, pour chaque nœud x_{t+1}, sélectionner le plus petit chemin étendu terminant en x_{t+1} comme le survivant au temps $t+1$. La récursion se poursuit indéfiniment sans que le nombre de survivants ne soit supérieur à M.

La valeur $\lambda(f_t)$ assignée à chaque branche dépend du canal sans mémoire utilisé. Plaçons-nous dans le cas le plus courant, le canal binaire symétrique. La probabilité d'erreur est de $p < \frac{1}{2}$. On a reçu un mot r, soit le bit r_t diffère de c_t et on note ainsi $d_t = 1$ soit le bit $r_t = c_t$ et on note $d_t = 0$. d_t peut être vu comme la distance de Hamming entre r_t et c_t. Donc la probabilité que c_t se transforme en un tel r_t est :

$$P(r_t|c_t) = p^{d_t}(1-p)^{1-d_t}$$

Ainsi :

$$-\ln P(r_t|c_t) = d_t \ln\left(\frac{1-p}{p}\right) - \ln(1-p)$$

Il est clair que le second terme est constant pour chaque t. De plus, comme $p < \frac{1}{2}$, on peut exprimer cette formule comme :

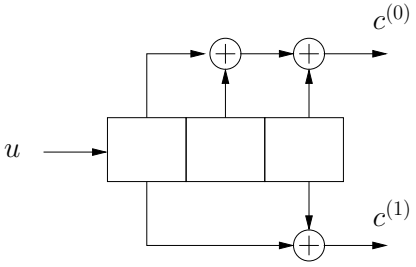
$$-\ln P(r_t|c_t) = \alpha d_t + \beta$$

où α et β sont des constantes positives et d_t la distance de Hamming (positive). Ainsi, dans ce cas, on peut prendre $\lambda(f_t) = d_t$ et donc :

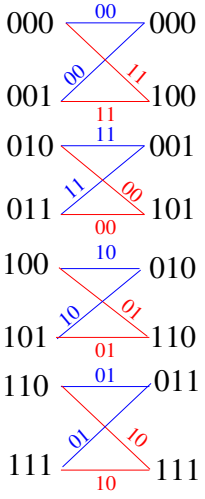
$$-\ln P(r,c) = \sum_{t=0}^{K-1} d_t$$

Donc minimiser $-\ln P(r,c)$ revient à minimiser la somme des distances entre chaque bit qui est exactement la définition de la distance de Hamming entre c et r.

Exemple 1.5. *Reprenons l'exemple précédent. On a le circuit suivant.*

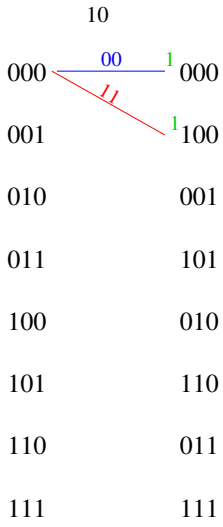


Avec ce circuit, on peut construire le treillis du code :

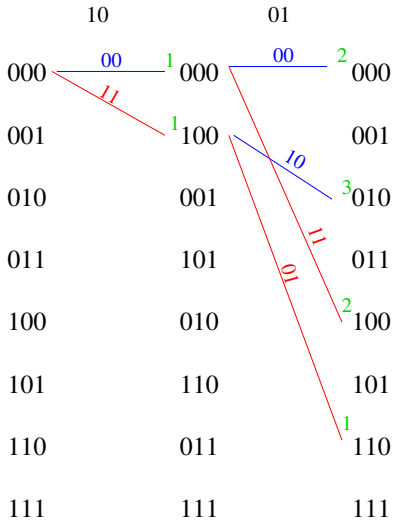


Les bits f_t de transition correspondent à la sortie de l'encodeur avec la mémoire remplie par les bits inscrits à droite du treillis. En bleu, c'est la transition de la mémoire lorsque qu'un 0 entre dans le registre et en rouge, c'est la transition lorsqu'un 1 entre dans le registre.

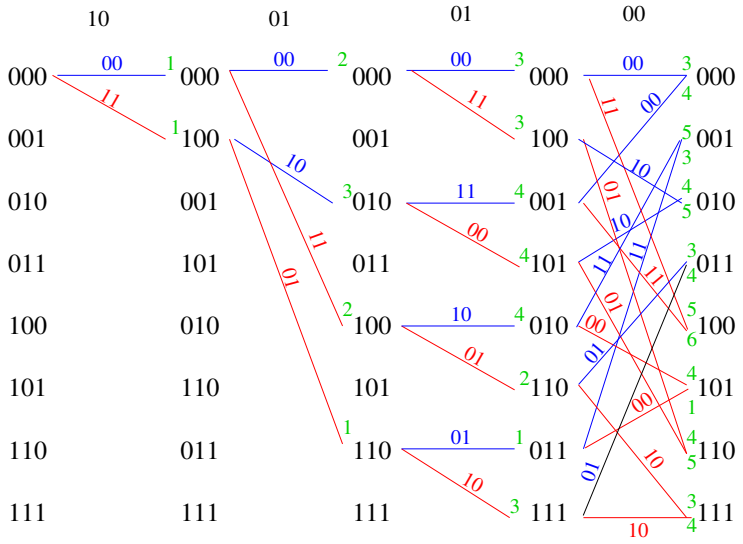
Supposons que le destinataire ait reçu le mot 1001010010. *Le codage commençant avec une mémoire nulle, le décodage commence avec l'état* 000. *On déroule le treillis et on calcule en vert la distance de Hamming entre les bits de sortie possibles du circuit et les bits reçus (écrits en noir au dessus du treillis).*



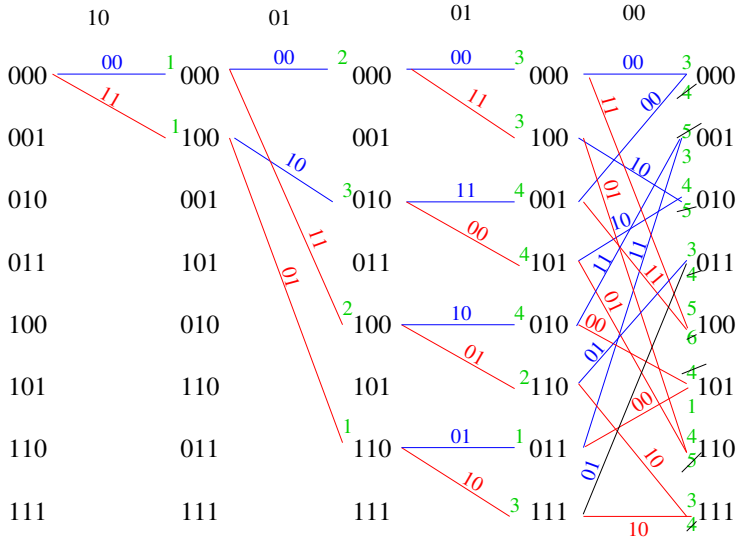
A la deuxième étape, on ajoute la distance de Hamming entre les bits reçus et les bits de sortie possibles du circuit à la distance de Hamming calculée à l'étape précédente.



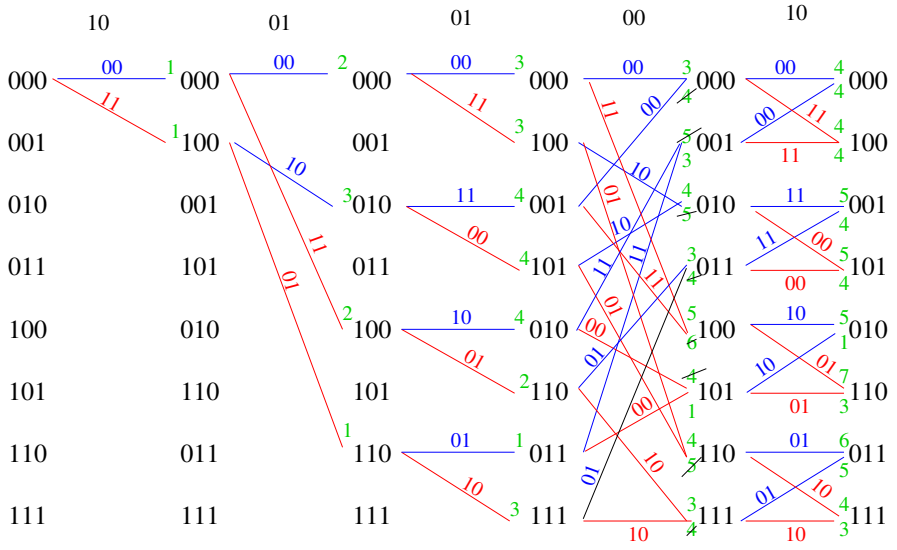
On continue le même processus à la troisième et quatrième étape.



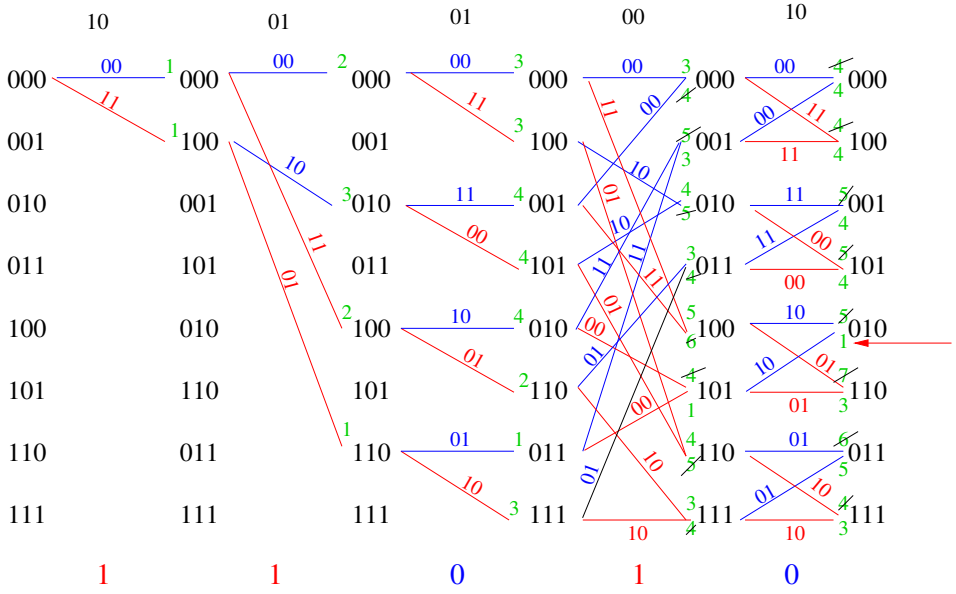
A partie de cette quatrième étape, on a deux chemins pour chaque état de la mémoire. On conserve les survivants, c'est-à-dire ceux avec la distance de Hamming la plus faible pour chaque état.



On poursuit l'algorithme avec ces survivants.



On obtient également deux chemins pour chaque état de la mémoire donc on conserve les survivants pour chaque état. On est arrivé à la fin du mot reçu donc on sélectionne le chemin qui a la plus courte distance de Hamming égale à 1 (flèche rouge). Puis on remonte ce chemin et on fait correspondre la couleur bleu à un bit de message égal à 0 et la couleur rouge à un bit de message égal à 1. On retrouve ainsi le message envoyé soit 11010.



1.3.3 Propriétés

En 1970, Forney [For70] démontre les propriétés des bons codes convolutifs, ayant un fort pouvoir de correction. Le message et la matrice génératrice sous forme polynomiale sont des éléments de $\mathbb{F}_2[X]$, l'anneau des polynômes à coefficients dans $\mathbb{F}_2$. Cet anneau est un anneau principal, qui a, entre autres, la propriété de n'avoir aucun diviseur de zéro.

Définition 1.11. *Soit $a \neq 0 \in A$, un anneau. On dit que a est un* ***diviseur de zéro à gauche*** *dans A s'il existe $b \neq 0 \in A$ tel que $ab = 0$. On dit que a est un* ***diviseur de zéro à droite*** *dans A s'il existe $b \neq 0 \in A$ tel que $ba = 0$. On dit que a est un diviseur de zéro dans A si a est un diviseur de zéro à gauche ou à droite.*

Cette propriété assure que tous les polynômes non nuls de $\mathbb{F}_2[X]$ peuvent être utilisés comme fonction de transfert du code. En effet, pour qu'un code soit bien défini, il est nécessaire qu'à chaque message corresponde un unique mot de code. S'il existe un polynôme $\tau \neq 0$, tel qu'il existe un $u \neq 0$ avec $u\tau = 0$, alors le message 0 et ce message u ont le même mot de code. Si on reçoit ce mot de code, on ne peut pas savoir de quel message il provient.

Certaines matrices génératrices de code de rendement $\frac{1}{n}$ sont des matrices sous forme systématique qui permettent, comme pour les codes en bloc, de répéter les bits du message. De telles matrices, pour un code de rendement $\frac{1}{n}$, sont de la forme

$$G_T(X) = \begin{pmatrix} 1 & g^{(1)}(X) & g^{(2)}(X) & \dots & g^{(n-1)}(X) \end{pmatrix}$$

Comme pour les codes en bloc, on va avoir une notion de distance afin de déterminer les codes optimaux, c'est-à-dire qui corrigeront le plus d'erreurs.

Définition 1.12. *On appelle* ***distance libre*** *d'un* (n, k) *code convolutif* C*, l'entier* $d_l(C)$ *défini par :*

$$d_l(C) = \min_{c \neq c'}(d_H(c, c'))$$

avec c *et* c' *deux mots du code* C *et* d_H *la distance de Hamming.*

1.3.4 Reconnaissance de codes

Les codes correcteurs d'erreurs, en bloc ou convolutifs, n'empêchent pas, en théorie, une tierce personne de retrouver le message à partir du mot intercepté. Cependant, en contexte non-coopératif, c'est-à-dire ne connaissant pas le code utilisé, remonter au message qui a été encodé par l'expéditeur n'est pas quelque chose de facile à réaliser. Des recherches ont donc été menées pour étudier la reconnaissance des paramètres d'un code correcteur d'erreurs dans ce contexte, ce que nous allons voir dans la suite.

1.3.4.1 Codes en bloc

Nous nous intéressons à la reconnaissance de codes en bloc dans l'hypothèse où le train de bits est non-bruité. Tout d'abord, il faut détecter si le train de bits en question est encodé ou non. Dans [Val00, BG03], le critère du rang est utilisé. Le train de bits est réorganisé sous forme de matrices M de taille $R \times l$ dont le rang est calculé. Le rang d'une matrice correspond au nombre de colonnes (ou de lignes) qui sont linéairement indépendantes. Or si le train de bits est le résultat d'un encodage alors il y a une dépendance entre les bits de message et les bits de redondance. Donc, il existera des matrices qui auront un rang non maximal.

De plus, les matrices dont le rang ne sera pas maximal seront celles avec un certain nombre de colonnes l. Pour l multiple de n (la longueur d'un mot de code) les matrices ne seront pas de rang plein. Cette méthode engendre une formule qui permet de trouver le rendement $\frac{k}{n}$ du code.

Si on fait l'hypothèse que le train de bits est bruité, le critère du rang donne de moins bons résultats à cause des erreurs qui peuvent rendre maximal le rang des matrices de longueur n. Cluzeau et Finiasz proposent dans [CF09] une méthode qui permet d'estimer la longueur et la synchronisation du code. Cette méthode utilise la matrice M construite par le même processus que pour la méthode du critère du rang, sauf qu'ils ne l'utilisent pas pour calculer le rang mais pour chercher un mot h du code dual de longueur n tel que $M \times h$ soit de petit poids. Pour trouver la bonne longueur et la bonne synchronisation, ils utilisent le fait que le poids du produit $M \times h$ ne suit pas la même distribution de probabilité selon que la longueur et la synchronisation sont les bonnes ou non.

Une fois la longueur et la synchronisation estimées, Valembois [Val01] montre que le problème d'identification du code linéaire est NP-complet. Cependant, Cluzeau [Clu04, Clu06] propose une méthode pour trouver les matrices génératrices de codes linéaires en bloc de petites tailles ou lorsque le taux d'erreur n'est pas trop

grand. Cette méthode utilise un test statistique couplé à l'algorithme de recherche de mots du code dual présenté dans [CF09]. La reconnaissance de certains codes en bloc, comme les codes BCH, les codes de Reed-Solomon ou les codes cycliques sont étudiés plus spécifiquement dans, par exemple, [ZHL+13], [LLWC13] et [ZHSY13] respectivement.

1.3.4.2 Codes convolutifs

Pour les codes convolutifs, les premiers travaux avec l'hypothèse d'un train de bits non-bruité sont réalisés en 1995 par Rice [Ric95] pour les codes de rendement $\frac{1}{2}$. Filiol [Fil97] étend ces travaux aux codes de rendement $\frac{1}{n}$, puis Barbier [Bar05] réalise la reconnaissance des codes convolutifs de rendement $\frac{k}{n}$. La méthode utilisée suppose que n et k sont connus et propose de résoudre le système linéaire $M^t h = 0$ avec M la matrice ayant pour lignes les mots de code interceptés. Pour trouver n et k, Mélanie Marazin explique dans ses travaux de thèse [Mar09] que les matrices M de taille $R \times l$, définies à partir des mots interceptés, auront, pour certaines, une chute de rang. La principale différence avec les codes en bloc est que la première matrice de rang déficient n'est pas de taille n mais de taille αn, $\alpha \geqslant 1$. La connaissance de ce αn permet de déduire n, k et la mémoire m du code.

Sous l'hypothèse que le train de bits est bruité, Barbier et al. [BSH06] ont proposé en 2006 un algorithme qui estime une base du code dual d'un code convolutif de rendement $\frac{1}{n}$. Puis Dingel et al. [DH07] ont présenté une approche différente basée sur l'algorithme Expectation-Maximization, qui permet d'identifier un code de rendement $\frac{1}{n}$. Une méthode basée sur l'algorithme d'Euclide a été proposée par Wang et al. [WHZ07], qui permet, dans un contexte bruité, d'identifier un code de rendement $\frac{1}{2}$. Dans [CS09], une amélioration de la méthode de Filiol [Fil97] utilisant l'algorithme de Valembois [Val01] est proposée par Côte et Sendrier pour les longueurs $n > 8$. Une reconstruction complète des codes convolutifs de rendement $\frac{k}{n}$ est effectuée par Marazin et al. dans [MGB11].

Dans cette thèse, on se place dans le contexte d'un attaquant qui intercepte un train de bits, qui ne connaît aucun des paramètres et qui tente de reconnaître en aveugle le code utilisé. L'objectif des codes que l'on cherche à construire est d'empêcher totalement la reconnaissance en aveugle des codes, grâce au caractère chiffrant de l'encodage.

1.3.5 Aspects cryptographiques

Comme nous l'avons précisé au début de ce chapitre, les codes correcteurs d'erreurs sont utiles pour corriger les erreurs de transmission, mais ne protègent pas les données contre une tierce personne les interceptant. Pour protéger son message, on utilise des algorithmes de chiffrement. Ces algorithmes utilisent une clé, qui est la seule garantie de la sécurité. Il existe deux types d'algorithmes, ceux à clé privée et ceux à clé publique.

Les algorithmes à clé privée utilisent une seule clé symétrique, ce qui signifie que l'expéditeur et le destinataire possèdent une clé commune identique, connue d'eux seuls. Le message est chiffré par un algorithme de chiffrement. Le destinataire

déchiffre avec un algorithme de déchiffrement qui est l'inverse de l'algorithme de chiffrement.

Les algorithmes à clé publique fonctionnent avec deux clés, l'une publique, l'autre privée. L'expéditeur du message demande au destinataire sa clé publique pour chiffrer le message. Le destinataire déchiffre le message avec sa clé privée. La connaissance de la clé publique n'apporte aucun avantage pour trouver la clé privée.

Donc pour protéger son message contre les erreurs de transmission et les tiers malveillants, l'expéditeur doit tout d'abord chiffrer son message avec un système de chiffrement, puis encoder le chiffré avec un code correcteur d'erreurs. Le destinataire décode le train de bit reçu et récupère le message chiffré qu'il déchiffre avec l'algorithme de déchiffrement adapté. On peut trouver plus d'informations sur les algorithmes de cryptographie dans [Sta11].

Certains systèmes cryptographiques sont basés sur des codes correcteurs d'erreurs. C'est le cas des cryptosystèmes de McEliece [McE78] et de Niederreiter [Nie86] qui sont des schémas de chiffrement à clé publique. Ils ont été prouvés équivalents en 1994 dans [LDW94]. Le message est chiffré en lui ajoutant autant d'erreurs que possible tout en gardant la possibilité de corriger celles-ci. La clé secrète est la méthode de correction de ces erreurs et la clé publique est la méthode d'encodage. Le cryptosystème de McEliece utilise les codes de Goppa [Gop70] qui sont faciles à décoder, mais s'ils sont masqués par une permutation, ils sont difficiles à distinguer d'un code linéaire. Et il est difficile de décoder un code linéaire aléatoire.

Dans ce cryptosystème le code est détourné de sa fonction initiale de correction d'erreur et est utilisé pour chiffrer le message. Il faut utiliser un autre code pour encoder le message chiffré.

Cette utilisation des codes pour faire de la cryptographie n'est pas ce qui va être abordé dans cette thèse. Nous allons nous intéresser à des codes correcteurs d'erreurs, dont nous aimerions que l'opération d'encodage soit une opération de chiffrement symétrique, et le décodage une opération de déchiffrement. Nous voulons, en d'autres mots, unifier les deux phases cryptographie + codes correcteurs d'erreurs en un seul algorithme.

Chapitre 2

Introduction aux objets mathématiques

Dans ce chapitre, nous allons faire un rappel sur la théorie des groupes. Nous verrons les actions de groupe, l'algèbre d'un groupe et la définition d'un domaine fondamental. Si vous êtes familiers avec ces notions, vous pouvez passer directement à la partie 2.5, où nous démontrerons un critère pour décider si un élément de l'algèbre d'un groupe fini est un diviseur de zéro ou non.

2.1 Théorie des groupes

Les définitions de ce chapitre sont tirées en partie de [Ulm12].

Définition 2.1. *Un* ***groupe*** $(G, \circ)$ *est la donnée d'un ensemble* G *et d'une loi de composition interne* $\circ$ *tels que :*

1. $\forall a, b \in G,\ a \circ b \in G$
2. $\forall a, b, c \in G,\ (a \circ b) \circ c = a \circ (b \circ c)$
3. $\exists e \in G,\ \forall a \in G,\ e \circ a = a \circ e = a$. *L'élément* e *est appelé l'élément neutre de* G.
4. $\forall a \in G,\ \exists b \in G, a \circ b = b \circ a = e$ *avec* e *l'élément neutre de* G. *L'élément* b *est appelé le symétrique de* a.

Notation :
En notation additive, la loi de composition interne est notée $+$, l'élément neutre est 0 et le symétrique de a est noté $-a$. En notation multiplicative, la loi de composition interne est notée $\times$ ou $\cdot$, l'élément neutre est 1 et le symétrique de a est noté a^{-1}.

Définition 2.2. *Soit* $(G, \circ)$ *un groupe. Si :*

$$\forall a, b \in G,\ a \circ b = b \circ a$$

alors le groupe est dit ***commutatif*** *ou abélien. Sinon le groupe est dit non-commutatif.*

Définition 2.3. ***L'ordre d'un groupe*** G, *noté* $|G|$ *est le cardinal de l'ensemble sous-jacent* G. *Si l'ensemble* G *contient un nombre fini* $n \in \mathbb{N}$ *d'éléments, on dit que le groupe* G *est d'ordre* n *; sinon il est dit d'ordre infini.*

Définition 2.4. *Soit $(G, \circ)$ un groupe. Un sous-ensemble H de G est un* ***sous-groupe*** *de G si :*

1. $e \in H$
2. $\forall h, h' \in H, h \circ h' \in H$,
3. $\forall h \in H, h^{-1} \in H$ *avec* h^{-1} *le symétrique de* h.

Définition 2.5. *Soit H un sous-groupe d'un groupe G. On dit que H est* ***distingué*** *(ou normal) dans G s'il est stable par conjugaison, c'est-à-dire si :*

$$\forall h \in H,\ \forall g \in G,\ ghg^{-1} \in H$$

On note alors $H \trianglelefteq G$.

Définition 2.6. *Étant donnés un groupe G et un sous-ensemble X de G, le* ***sous-groupe de G engendré par X*** *est le plus petit sous-groupe de G contenant X. On le note $\langle X \rangle_G$ ou simplement $\langle X \rangle$. Pour un sous-ensemble fini $\{g_0, g_1, \ldots, g_{n-1}\}$ de G on note $\langle g_0, g_1, \ldots, g_{n-1} \rangle$.*

Définition 2.7. *Soient G un groupe et X un sous-ensemble de G. L'ensemble des sous-groupes distingués de G contenant X n'est pas vide car il comprend au moins G. L'intersection des sous-groupes distingués de G contenant X est un sous-groupe distingué de G. C'est le plus petit sous-groupe distingué de G contenant X. On l'appelle le* ***sous-groupe distingué de G engendré par X****.*

Par minimalité de $\langle X \rangle$, il est clair que le sous-groupe $\langle X \rangle$ engendré par X est contenu dans le sous-groupe distingué de G engendré par X.

Définition 2.8. *On appelle* ***ordre d'un élément*** *g de G l'ordre du sous-groupe $\langle g \rangle$ engendré par g.*

Un groupe engendré par $X = \{x_j \mid j \in J\}$ contient également l'inverse des générateurs x_j. En notation multiplicative, $\langle X \rangle$ contient l'ensemble de tous les mots de longueur finie en les $x_j \in X$ et leurs inverses x_j^{-1}.

Théorème 2.1. *Soient G un groupe et $g \in G$ un élément d'ordre fini $n \in \mathbb{N}$. Alors, n est le plus petit entier strictement positif ayant la propriété $g^n = e$ et on a $\langle g \rangle = \{g, g^2, \ldots, g^n = e\}$. Pour $k \in \mathbb{Z}$ on a $g^k = e$ si et seulement si n divise k.*

Démonstration. Preuve réalisée page 7 de [Ulm12]. □

Nous allons maintenant définir des notions liées aux applications entre ensembles et entre groupes.

Définition 2.9. *Une application f d'un ensemble E vers un ensemble F est dite* ***injective*** *si :*

$$\forall x, y \in E,\ f(x) = f(y) \Rightarrow x = y$$

Une application f d'un ensemble E vers un ensemble F est dite ***surjective*** *si :*

$$\forall y \in F,\ \exists x \in E,\ y = f(x)$$

Une application d'un ensemble E vers un ensemble F est dite ***bijective*** *si elle est injective et surjective.*

Définition 2.10. *Soient $(G, \circ)$ et $(\Gamma, \bullet)$, deux groupes dont les éléments neutres sont respectivement e_G et e_Γ. On appelle :*

- ***morphisme de groupes*** *de G vers Γ toute application $\phi : G \to \Gamma$ qui vérifie la relation $\phi(g \circ h) = \phi(g) \bullet \phi(h)$ pour tout $g, h \in G$.*
- *$Hom(G, \Gamma)$ est l'ensemble des morphismes (on dit aussi homomorphismes) de G vers Γ.*
- ***noyau*** *de ϕ qu'on note $Ker(\phi)$ l'ensemble*

$$Ker(\phi) = \{g \in G \mid \phi(g) = e_\Gamma\}$$

  des éléments de G dont l'image par ϕ est l'élément neutre de Γ.
- ***image*** *de ϕ qu'on note $Im(\phi)$, l'ensemble*

$$Im(\phi) = \{\gamma \in \Gamma \mid \exists g \in G, \phi(g) = \gamma\}$$

  des éléments de Γ de la forme $\phi(g)$ pour $g \in G$.
- ***isomorphisme*** *de G dans Γ tout morphisme de G bijectif.*
- ***endomorphisme*** *de G tout morphisme de G dans lui-même.*
- *$End(G)$ l'ensemble des endomorphismes du groupe G.*
- ***automorphisme*** *de G tout endomorphisme bijectif de G.*
- *$Aut(G)$ l'ensemble des automorphismes du groupe G. Cet ensemble, muni de la loi de composition est un groupe.*
- ***anti-morphisme*** *de G vers Γ toute application $\xi : G \to \Gamma$ du groupe G vers le groupe Γ qui vérifie la relation $\xi(g \circ h) = \xi(h) \bullet \xi(g)$ pour tous $g, h \in G$.*

Nous présentons maintenant la définition des groupes quotients.

Définition 2.11. *Une **relation d'équivalence** $\sim$ sur un ensemble X est une relation $\sim$ sur X telle que :*

1. *$\sim$ est réflexive : $\forall x \in X,\ x \sim x$*
2. *$\sim$ est symétrique : $\forall x, y \in X,\ x \sim y \Rightarrow y \sim x$*
3. *$\sim$ est transitive : $\forall x, y, z \in X,\ x \sim y$ et $y \sim z \Rightarrow x \sim z$*

Définition 2.12. *Soient x un ensemble, $\sim$ une relation d'équivalence sur X et $x \in X$. L'ensemble $\{y \in X \mid x \sim y\}$ est appelé **classe d'équivalence** de x. Il est noté $\overline{x}$.*

Définition 2.13. *Soient G un groupe et H un sous-groupe de G. La relation « $g_1 \sim_H g_2$ s'il existe $h \in H$ tel que $g_1 = g_2 h$ » définit une relation d'équivalence sur G. Les classes d'équivalences sont les sous-ensembles $gH = \{gh \mid h \in H\}$ de G. Elles sont appelées les **classes à gauche de** G **modulo** H.*

Remarque : On peut définir de la même façon les classes à droite $Hg = \{hg \mid h \in H\}$

Définition 2.14. *Soient G un groupe et H un sous-groupe de G. On appelle **ensemble quotient** de G par la relation d'équivalence $\sim_H$ et on note G/H l'ensemble $\{gH \mid g \in G\}$ des classes à gauche de G modulo H.*

Définition 2.15. *Soit $\sim$ une relation d'équivalence sur un ensemble X, l'ensemble $X/\sim = \{\overline{x} \mid x \in X\}$ des classes d'équivalence est appelé* ***l'ensemble quotient*** *de X par $\sim$. L'application*

$$\begin{aligned} \pi : X &\to X/\sim \\ x &\mapsto \overline{x} \end{aligned}$$

est appelée application canonique.

Proposition 2.1. *Soit H un sous-groupe d'un groupe G. Alors les conditions suivantes sont équivalentes :*

1. *H est distingué dans G.*
2. *$gHg^{-1} = H$ pour tout $g \in G$.*
3. *$gH = Hg$ pour tout $g \in G$.*

Démonstration. H est distingué dans G signifie qu'il est stable par conjugaison et donc (1) $\Leftrightarrow$ (2). Pour passer de (2) à (3) ou l'inverse, il suffit de multiplier à droite par g ou g^{-1}. □

Théorème 2.2. *Soit G un groupe. Un sous-groupe H de G est distingué dans G si et seulement si la loi $g_1H \circ g_2H = (g_1g_2)H$ est une loi de groupe $\circ$ sur l'ensemble quotient G/H. Le groupe $(G/H, \circ)$ est appelé le* ***groupe quotient*** *de G par H.*

Démonstration. Vérifions que la loi $\circ$ est bien définie. Soient $g_1' \sim g_1$ et $g_2' \sim g_2$. On peut écrire, d'après la proposition 2.1, $g_1' = h_1g_1$ et $g_2' = g_2h_2$, avec $h_1, h_2 \in H$, d'où $g_1'g_2' = h_1(g_1g_2)h_2$. Ainsi $g_1'g_2' \in H(g_1g_2h_2) = (g_1g_2h_2)H$ d'après la proposition 2.1, mais ce dernier ensemble n'est autre que $(g_1g_2)H$ car $h_2 \in H$. Finalement $g_1'g_2' \sim g_1g_2$. □

Définition 2.16. *Soit $\mathcal{M}(\mathcal{A})$ l'ensemble des mots de longueur finie sur un alphabet $\mathcal{A}$ en les $a_i \in \mathcal{A}$ et leurs inverses a_i^{-1}. La loi de composition est la concaténation des mots avec le mot vide comme élément neutre. Deux mots m et m' sont dit équivalents, et on note $m \sim m'$, si l'on peut aller de l'un à l'autre en ajoutant ou en enlevant des termes de la forme $a_ia_i^{-1}$ ou $a_i^{-1}a_i$. La relation $\sim$ est une relation d'équivalence et nous noterons $F(\mathcal{A})$ l'ensemble quotient $\mathcal{M}(\mathcal{A})/\sim$.*

On appelle ***groupe libre*** *sur l'alphabet $\mathcal{A}$ et on note $F(\mathcal{A})$ le groupe dont l'ensemble sous-jacent est $\mathcal{M}(\mathcal{A})/\sim$ et dont la loi est la concaténation des représentants de classes de mots.*

Proposition 2.2. *Soient $\mathcal{A}$ un ensemble et G un groupe. Toute application $f : \mathcal{A} \to G$ peut être étendue de manière unique en un morphisme $\varphi_f : F(\mathcal{A}) \to G$.*

Définition 2.17. *Soient $\mathcal{A}$ un ensemble, G un groupe et $\varphi_f : F(\mathcal{A}) \to G$ un morphisme surjectif. Un élément de $Ker(\varphi_f)$ est appelé une* ***relation*** *entre les générateurs $\{f(a) \mid a \in \mathcal{A}\}$ de G. Soit un ensemble R tel que $Ker(\varphi_f)$ soit le sous-groupe distingué engendré par R alors on appelle $\mathcal{A}$ et R une* ***présentation par générateurs et relations*** *de G et on note $G = \langle \mathcal{A} \mid R \rangle$.*

Un groupe G est dit ***de type fini*** *s'il est engendré par une partie $\mathcal{A}$ finie.*

Définition 2.18. *Soient G_1 et G_2 deux groupes. Désignons par $G_1 \times G_2$ leur produit cartésien. Il est naturel de définir sur $G_1 \times G_2$ une loi de composition $\star$ composante par composante :*

$$(x_1, x_2) \star (y_1, y_2) = (x_1 y_1, x_2 y_2)$$

le produit $x_1 y_1$ apparaissant dans le second membre étant calculé dans G_1 et le produit $x_2 y_2$ dans G_2. On vérifie facilement que cette loi de composition munit $G_1 \times G_2$ d'une structure de groupe. Ce groupe est appelé ***produit direct*** *des groupes G_1 et G_2 et noté $G_1 \times G_2$. Si e_1 et e_2 désignent respectivement les neutres de G_1 et de G_2, le neutre de $G_1 \times G_2$ est (e_1, e_2). Le symétrique d'un élément (x_1, x_2) de $G_1 \times G_2$ est l'élément (x_1^{-1}, x_2^{-1}).*

Définition 2.19. *Soient A et H deux groupes, et ϕ un morphisme de H dans $Aut(A)$ le groupe des automorphismes de A, c'est-à-dire :*

$$\begin{aligned} \phi : H &\to Aut(A) \\ h &\mapsto \phi(h) = (\phi_h : a \mapsto \phi_h(a)) \end{aligned}$$

Le ***produit semi-direct externe*** *G de A et H suivant ϕ est défini comme le produit cartésien de A et H muni de la loi de groupe :*

$$(a_1, h_1)(a_2, h_2) = (a_1 \phi_{h_1}(a_2), h_1 h_2)$$

On note $G = A \rtimes_\phi H$ ou tout simplement $G = A \rtimes H$.

Définition 2.20. *Soit X un ensemble. Alors l'ensemble $S(X)$ des bijections de X dans X, muni de la composition des applications est un groupe qu'on appelle le* ***groupe symétrique*** *de X.*

Pour $n \in \mathbb{N}$, on note simplement S_n le groupe $S(\{1, 2, \ldots, n\})$. C'est un groupe fini d'ordre $n! = n(n-1)\ldots 1$.

Nous notons les éléments sous la forme $\begin{pmatrix} 1 & 2 & \ldots & n \\ \sigma(1) & \sigma(2) & \ldots & \sigma(n) \end{pmatrix}$ avec $\sigma(i)$ l'image par la permutation σ de l'élément $i \in \{1, 2, \ldots, n\}$.

Exemple 2.1. *Considérons les éléments $\sigma = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}$ et $\rho = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}$ du groupe S_3. La composition des permutations dans S_n correspond à la formule $\sigma\rho(i) = \sigma(\rho(i))$ pour tout $i \in \{1, \ldots, n\}$. Ainsi, $\sigma\rho = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$ et $\rho\sigma = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$.*

Définition 2.21. *Soient $1 \leqslant l \in \mathbb{N}$ et $i_1, i_2, \ldots, i_l$ des éléments distincts de $\{1, \ldots, n\}$. La permutation $\gamma \in S_n$ définie par*

$$\gamma(j) = \begin{cases} j & si\ j \notin \{i_1, i_2, \ldots, i_l\} \\ i_{k+1} & si\ j = i_k\ avec\ k < l \\ i_1 & si\ j = i_l \end{cases}$$

et notée $(i_1, i_2, \ldots, i_l)$ est appelée ***cycle*** *de longueur l.*

Exemple 2.2. $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 5 & 3 & 2 & 1 \end{pmatrix}$ *s'écrit sous forme de cycle comme $(1\ 4\ 2\ 5) \in S_5$.*

2.2 Actions de groupe

Soient X un ensemble et G un groupe.

Définition 2.22. *Une* ***action à gauche*** *de G sur X est la donnée de :*

$$\begin{aligned} G \times X &\to X \\ (g, x) &\mapsto g \cdot x \end{aligned}$$

vérifiant les propriétés suivantes :

$$\begin{aligned} &\forall x \in X,\ e \cdot x = x \\ &\forall (g, g') \in G^2,\ \forall x \in X,\ g' \cdot (g \cdot x) = (g'g) \cdot x \end{aligned}$$

On dit que X est un G-ensemble à gauche.

Définition 2.23. *Une* ***action à droite*** *de G sur X est la donnée de :*

$$\begin{aligned} X \times G &\to X \\ (x, g) &\mapsto x \cdot g \end{aligned}$$

vérifiant les propriétés suivantes :

$$\begin{aligned} &\forall x \in X,\ x \cdot e = x \\ &\forall (g, g') \in G^2,\ \forall x \in X,\ (x \cdot g) \cdot g' = x \cdot (gg') \end{aligned}$$

On dit que X est un G-ensemble à droite.

Proposition 2.3. *Il existe une correspondance bijective entre les actions à gauche de G sur X et les morphismes $\gamma : G \to S(X)$ définie de la manière suivante : à une action à gauche d'un groupe G sur un ensemble X correspond le morphisme :*

$$\begin{aligned} \varphi : G &\to S(X) \\ g &\mapsto \sigma_g \text{ avec } \sigma_g(x) = g.x \end{aligned}$$

Définition 2.24. *Soient G un groupe agissant à gauche sur un ensemble X et $x \in X$.*
On appelle ***orbite*** *de x sous G le sous-ensemble $G.x = \{g.x \mid g \in G\} \subseteq X$.*
On appelle ***stabilisateur*** *de x dans G le sous-groupe $G_x = \{g \in G \mid g.x = x\} \subseteq G$.*
On dit que G agit ***librement*** *sur X si tous les stabilisateurs sont triviaux, c'est-à-dire si $G_x = \{e\}$ pour tout $x \in X$.*
On dit que G agit ***transitivement*** *sur X s'il existe exactement une orbite dans X.*

Proposition 2.4. *Soit G un groupe agissant à gauche sur un ensemble X. La relation $\sim$ définie sur X par « $x \sim y$ si x appartient à l'orbite de y » est une relation d'équivalence sur X dont les classes d'équivalences sont les orbites des éléments de X sous l'action de G. Autrement dit, les orbites des éléments de X sous l'action de G forment une partition de X. On note G/X l'ensemble quotient $X/\sim$.*

Démonstration. Preuve réalisée page 30 de [Ulm12]. □

On va maintenant voir comment les actions agissent sur les applications entre deux groupes.

Définition 2.25. *Soit Y un ensemble et X un G-ensemble à gauche, et considérons X^Y l'ensemble des applications de Y dans X. On définit :*

$$\forall f \in X^Y,\ g \in G,\ y \in Y,\ (g \cdot f)(y) = g \cdot (f(y)) \tag{2.1}$$

Théorème 2.3. *X^Y muni de l'action (2.1) est un G-ensemble à gauche.*

Démonstration. Soit $f \in X^Y$, $g, g' \in G^2$, $y \in Y$.

$$\begin{aligned}((gg') \cdot f)(y) &= (gg') \cdot (f(y)) \\ &= g \cdot (g' \cdot f(y)) \\ &= g \cdot ((g' \cdot f)(y)) \\ &= (g \cdot (g' \cdot f))(y)\end{aligned}$$

Donc $\forall f \in X^Y, g, g' \in G^2, (gg') \cdot f = g \cdot (g' \cdot f)$ et on a bien la propriété d'associativité. De plus,

$$(e \cdot f)(y) = e \cdot (f(y)) = f(y)$$

Donc X^Y est bien un G-ensemble à gauche. □

Définition 2.26. *Soit Y un ensemble et X un G-ensemble à droite, et considérons X^Y l'ensemble des applications de Y dans X. On définit :*

$$\forall f \in X^Y,\ g \in G,\ y \in Y,\ (f \cdot g)(y) = f(y) \cdot g \tag{2.2}$$

Théorème 2.4. *X^Y muni de l'action (2.2) est un G-ensemble à droite.*

Démonstration. Soit $f \in X^Y$, $g, g' \in G^2$, $y \in Y$.

$$\begin{aligned}(f \cdot (gg'))(y) &= f(y) \cdot (gg') \\ &= (f(y) \cdot g) \cdot g' \\ &= ((f \cdot g)(y)) \cdot g' \\ &= ((f \cdot g) \cdot g')(y)\end{aligned}$$

Donc $\forall f \in X^Y, g, g' \in G^2, f \cdot (gg') = (f \cdot g) \cdot g'$ et on a bien la propriété d'associativité. De plus,

$$(f \cdot e)(y) = f(y) \cdot e = f(y)$$

Donc X^Y est bien un G-ensemble à droite. □

Définition 2.27. *Soit Y un ensemble et X un G-ensemble à gauche, et considérons Y^X l'ensemble des applications de X dans Y. On définit :*

$$\forall f \in Y^X,\ g \in G,\ x \in X,\ (f \cdot g)(x) = f(g \cdot x) \tag{2.3}$$

Théorème 2.5. *Y^X muni de l'action (2.3) est un G-ensemble à droite.*

Démonstration. Soit $f \in Y^X$, $g, g' \in G^2$, $x \in X$.

$$\begin{aligned}(f \cdot (gg'))(x) &= f((gg') \cdot x) \\ &= f(g \cdot (g' \cdot x)) \\ &= (f \cdot g)(g' \cdot x) \\ &= ((f \cdot g) \cdot g')(x)\end{aligned}$$

Donc $\forall f \in Y^X, g, g' \in G^2, f \cdot (gg') = (f \cdot g) \cdot g'$ et on a bien la propriété d'associativité. De plus,

$$(f \cdot e)(x) = f(e \cdot x) = f(x)$$

Donc Y^X est bien un G-ensemble à droite. □

Définition 2.28. *Soit Y un ensemble et X un G-ensemble à droite, et considérons Y^X l'ensemble des applications de X dans Y. On définit :*

$$\forall f \in Y^X,\ g \in G,\ x \in X,\ (g \cdot f)(x) = f(x \cdot g) \tag{2.4}$$

Théorème 2.6. *Y^X muni de l'action (2.4) est un G-ensemble à gauche.*

Démonstration. Soit $f \in Y^X$, $g, g' \in G^2$, $x \in X$.

$$\begin{aligned}((gg') \cdot f)(x) &= f(x \cdot (gg')) \\ &= f((x \cdot g) \cdot g') \\ &= (g' \cdot f)(x \cdot g) \\ &= (g \cdot (g' \cdot f))(x)\end{aligned}$$

Donc $\forall f \in Y^X, g, g' \in G^2, (gg') \cdot f = g \cdot (g' \cdot f)$ et on a bien la propriété d'associativité. De plus,

$$(e \cdot f)(x) = f(x \cdot e) = f(x)$$

Donc Y^X est bien un G-ensemble à gauche. □

En résumé, si Y est un ensemble, $\forall x \in X, y \in Y, g, g' \in G$, on a :

	X^Y	Y^X
X G-ensemble à gauche	G-ensemble à gauche $(g \cdot f)(y) = g \cdot f(y)$	G-ensemble à droite $(f \cdot g)(x) = f(g \cdot x)$
X G-ensemble à droite	G-ensemble à droite $(f \cdot g)(y) = f(y) \cdot g$	G-ensemble à gauche $(g \cdot f)(x) = f(x \cdot g)$

Ce tableau, on peut aussi le voir comme ceci :

	X^Y
Y ensemble X G-ensemble à gauche	G-ensemble à gauche $(g \cdot f)(y) = g \cdot f(y)$
Y ensemble X G-ensemble à droite	G-ensemble à droite $(f \cdot g)(y) = f(y) \cdot g$
X ensemble Y G-ensemble à gauche	G-ensemble à droite $(f \cdot g)(y) = f(g \cdot y)$
X ensemble Y G-ensemble à droite	G-ensemble à gauche $(g \cdot f)(y) = f(y \cdot g)$

Soit X un ensemble pointé, c'est-à-dire qu'il admet un élément $x_0 \in X$ appelé point de base. Soit G un groupe que l'on considère comme un G-ensemble à droite en faisant agir G sur lui-même par translations à droite, on a donc que X^G est un G-ensemble à gauche. Soit $E \subseteq G$. Une application $f \in X^E$ est étendue à G par le point de base x_0 et ainsi on a $X^E \subseteq X^G$.

Théorème 2.7. *Pour tout ensemble $E \subseteq G$, pour tout $g \in G$ et pour tout ensemble X, on a l'égalité :*

$$g \cdot X^E = X^{E \cdot g^{-1}}$$

Démonstration. Soient $f \in X^E$ et $e \in E$, on a $(g \cdot f)(e) = f(e \cdot g) = x_0$ lorsque $e \cdot g \notin E$ c'est-à-dire lorsque $e \notin E \cdot g^{-1}$. Cela montre bien que $g \cdot X^E = X^{E \cdot g^{-1}}$. □

On considère maintenant G comme un G-ensemble à gauche, donc X^G est un G-ensemble à droite.

Théorème 2.8. *Pour tout ensemble $E \subseteq G$, pour tout $g \in G$ et pour tout ensemble X, on a l'égalité :*

$$X^E \cdot g = X^{g^{-1} \cdot E}$$

Démonstration. Soient $f \in X^E$ et $e \in E$, on a $(f \cdot g)(e) = f(g \cdot e) = x_0$ lorsque $g \cdot e \notin E$ c'est-à-dire lorsque $e \notin g^{-1} \cdot E$. Cela montre bien que $X^E \cdot g = X^{g^{-1} \cdot E}$. □

2.3 Algèbre de groupe

On va commencer par définir ce qu'est un anneau.

Définition 2.29. *Un **anneau** unitaire associatif (on considérera dans la suite uniquement des anneaux de ce type) $(A, +, .)$ est un ensemble A muni de deux lois de composition $+$ et $.$ (addition et multiplication) qui satisfont aux axiomes suivants :*

1. *$(A, +)$ est un groupe commutatif ; nous notons son élément neutre 0.*
2. *La loi $.$ est associative : $(a.b).c = a.(b.c)$ pour tout $a, b, c \in A$.*
3. *La multiplication est distributive par rapport à l'addition :*
   $(a + b).c = (a.c) + (b.c)$ et $c.(a + b) = (c.a) + (c.b)$ pour tout $a, b, c \in A$.
4. *Il existe un élément neutre 1 pour la multiplication, appelé élément unité de A : $1.a = a.1 = a$ pour tout a dans A.*

Un anneau A est dit commutatif si $a.b = b.a$ pour tout $a, b \in A$.

Définition 2.30. *Soit A un anneau. Un sous-ensemble $I \subseteq A$ est un **idéal bilatère** de A si :*

1. *$(I, +)$ est un sous-groupe de $(A, +)$*
2. *pour tout $a \in A$ et pour tout $b \in I$, les produits ab et ba appartiennent à I.*

Dans un anneau, les idéaux bilatères jouent le rôle des sous-groupes distingués dans les groupes. Le quotient A/I possède une structure d'anneau.

Définition 2.31. *Soit A un anneau et $(M,+)$ un groupe commutatif. Si, de plus, M est muni d'une loi externe $\cdot$ de $A \times M$ dans M vérifiant, pour tous éléments a et b de A et x, y de M :*

- *$a \cdot (x+y) = a \cdot x + a \cdot y$ (distributivité de $\cdot$ par rapport à l'addition dans M)*
- *$(a+b) \cdot x = a \cdot x + b \cdot x$ (distributivité de $\cdot$ par rapport à l'addition dans A) Remarque : la loi $+$ du membre de gauche est celle de l'anneau A et la loi $+$ du membre de droite est celle du groupe M*
- *$(ab) \cdot x = a \cdot (b \cdot x)$*
- *$1 \cdot x = x$*

*alors $(M,+,\cdot)$ est un **A-module à gauche**.*
On peut définir un A-module à droite en plaçant les éléments de A à droite dans la loi externe.

Définition 2.32. *Soit E un A-module à gauche et M une partie de E. On dit que M est un **sous-module** (à gauche) si les conditions suivantes sont respectées :*

- *M est un sous-groupe de $(E,+)$*
- *Pour tout $a \in A$, $x \in M$, $a \cdot x \in M$*

Définition 2.33. *On appelle **module libre** un module M qui possède une base B, c'est-à-dire une partie B de M qui est à la fois :*

- *génératrice pour M, c'est-à-dire que tout élément de M est combinaison linéaire d'éléments de B*
- *libre, c'est-à-dire que pour toutes familles finies $(e_i)_{1 \leqslant i \leqslant n}$ d'éléments de B deux à deux distincts et $(a_i)_{1 \leqslant i \leqslant n}$ d'éléments de l'anneau sous-jacent telles que $a_1 e_1 + \ldots + a_n e_n = 0$, on a : $a_1 = \ldots = a_n = 0$.*

Définition 2.34. *Soient A un anneau commutatif et G un groupe. On note A^G le A-module des applications de G dans A. La A-algèbre de G, notée $A[G]$, est le sous-module de A^G constitué des applications de support fini (c'est-à-dire nulles sauf sur une partie finie de G), muni de l'opération de convolution définie par :*

$$(f * h)(g) = \sum_{t \in G} f(t) h(t^{-1} g).$$

Plus explicitement, un élément de $A[G]$ est noté

$$f = \sum_{g \in G} \alpha_g g, \quad \alpha_g \in A$$

où les éléments α_g sont presque tous nuls, et la convolution de deux tels éléments est donné par :

$$\left(\sum_{x \in G} \alpha_x x \right) * \left(\sum_{y \in G} \beta_y y \right) = \sum_{g \in G} \left(\sum_{x,y \in G, xy = g} \alpha_x \beta_y \right) g.$$

*$A[G]$ est appelée **l'algèbre du groupe** G.*

Définition 2.35. *Un **corps** commutatif est un anneau commutatif non nul dans lequel tout élément non nul est inversible.*

Définition 2.36. *Soit* 1 *l'unité d'un anneau* K*. S'il existe un entier naturel* n *non nul tel que* $n \times 1 = 1 + 1 + \ldots + 1$ *(additionné* n *fois) est nul, on appelle* ***caractéristique*** *de* K*, et on note* $car(K)$*, le plus petit entier positif non nul vérifiant cette propriété. S'il n'existe pas d'entier non nul vérifiant cette propriété, on dit que* K *est de caractéristique nulle.*

Définition 2.37. *Soit* K *un corps. Un* ***espace vectoriel*** *sur* K*, ou* K*-espace vectoriel, est un module sur l'anneau* K*.*

Nous aurons besoin dans la suite de la définition de l'indicatrice d'Euler qui est la suivante.

Définition 2.38. ***L'indicatrice d'Euler*** *est la fonction* φ*, de l'ensemble* $\mathbb{N}^*$ *des entiers strictement positifs dans lui-même, définie par :*

$$\begin{array}{rccl} \varphi & : & \mathbb{N}^* & \longrightarrow & \mathbb{N}^* \\ & & n & \longmapsto & Card(\{m \in \mathbb{N}^* \mid m \leqslant n \text{ et } m \text{ premier avec } n\}). \end{array}$$

2.4 Domaine fondamental et topologie algébrique

Définition 2.39. *Un* ***domaine fondamental*** *pour l'action d'un groupe sur un ensemble* E *est un sous-ensemble de* E *dont les images par l'action du groupe forment une partition de* E*. C'est donc un domaine contenant exactement un point par orbite du groupe. Autrement dit, soit* G *un groupe,* E *un ensemble sur lequel* G *agit. On note* $g.x$ *l'image d'un point* x *de* E *par l'action de l'élément* $g \in G$*. Un sous-ensemble* F *de* E *est appelé domaine fondamental pour l'action du groupe si :*

— $\bigcup_{g \in G} g(F) = E$

— $\forall g, g' \in G$ *tels que* $g \neq g', g(F) \cap g'(F) = \emptyset$

Dans la suite de la thèse, on va considérer des surfaces connexes sur lesquelles vont agir un groupe G. Définissons la notion de connexité. Nous nous plaçons sur un espace topologique dont la définition est la suivante.

Définition 2.40. *Un espace topologique* $(X, \mathcal{U})$ *est la donnée d'un ensemble* X *et d'une topologie* $\mathcal{U}$ *sur* X *ce qui signifie que* $\mathcal{U}$ *est une collection de sous-ensembles de* X *telle que :*

1. $\emptyset$ *et* X *appartiennent à* $\mathcal{U}$
2. *pour tous* $U_1, U_2 \in \mathcal{U}$*, on a* $U_1 \cap U_2 \in \mathcal{U}$
3. *pour toute famille* $(U_i)_{i \in I}$*,* $U_i \in \mathcal{U}$*, on a* $\bigcup_{i \in I} U_i \in \mathcal{U}$*.*

Les éléments U *de* $\mathcal{U}$ *sont appelés les ouverts de* X*.*

Définition 2.41. *Soit* E *un espace topologique. Un* ***fermé*** *est un sous-ensemble de* E *dont le complémentaire est un ouvert.*

Définition 2.42. *Un* ***espace topologique*** *non vide* E *est dit* ***connexe*** *s'il n'existe pas dans* E *de sous-ensemble à la fois ouvert et fermé distinct du vide et de* E*.*

La connexité formalise le concept d'« objet d'un seul tenant ».

Définition 2.43. *Un* ***homéomorphisme*** *est une application bijective continue, d'un espace topologique dans un autre, dont la bijection réciproque est continue. Dans ce cas, les deux espaces topologiques sont dits homéomorphes.*

La notion d'homéomorphisme est la bonne notion pour dire que deux espaces topologiques sont « le même » vu différemment.

Pour la définition du genre d'une surface, nous avons tout d'abord besoin de définir un voisinage d'un point.

Définition 2.44. *Si x est un point de X, on dit que W est un* ***voisinage*** *de x s'il existe un ouvert $U \subset X$ contenant x et inclus dans W.*

Définition 2.45. *Soit S une surface, c'est-à-dire un espace topologique dont tout point possède un voisinage homéomorphe au plan. Le* ***genre*** *de la surface connexe S est le nombre maximum de courbes fermées simples sans points communs pouvant être tracées à l'intérieur de cette surface sans la déconnecter.*

Exemple 2.3. *La sphère et un disque ont un genre 0. Le tore a un genre 1.*

La notion de genre formalise le « nombre de trous » de la surface.

Le genre g peut être défini à partir de la caractéristique d'Euler χ, ainsi, pour une surface orientable, on aura $\chi = 2 - 2g$. Dans cette thèse, nous allons nous intéresser uniquement à des surfaces orientables, pour plus d'informations sur cette notion, se référer à [Rat06], partie 9.1.

La caractéristique d'Euler est la formule suivante, avec S le nombre de sommets d'un polyèdre, A le nombre d'arêtes et F le nombre de faces :

$$\chi = S - A + F$$

2.5 Diviseurs de zéro dans une algèbre d'un groupe fini

Dans certaines algèbres de groupe, certains éléments sont des diviseurs de zéro. Ces éléments ne peuvent donc pas être choisi comme fonction de transfert d'un code convolutif, car à un message doit correspondre un unique mot de code. Nous démontrons ici un critère pour déterminer si un élément de $\mathbb{F}_{2^k}[G]$ est un diviseur de zéro ou non dans le cas où G est un groupe fini.

On rappelle que, dans un groupe fini G, l'ensemble des applications de G dans $\mathbb{F}_{2^k}$, noté $\mathbb{F}_{2^k}^G$ est également l'algèbre du groupe G sur $\mathbb{F}_{2^k}$, notée $\mathbb{F}_{2^k}[G]$. Soit h un élément de cette algèbre, on note h^n le produit de n copies de h.

On rappelle la définition d'un diviseur de zéro à droite.

Définition 2.46. *Soit $a \neq 0 \in A$, un anneau. On dit que a est un* ***diviseur de zéro à droite*** *dans A s'il existe $b \neq 0 \in A$ tel que $ba = 0$.*

Lorsqu'un élément n'est pas un diviseur de zéro à droite, on dit que c'est un élément régulier à droite.

Définition 2.47. *Soit A un anneau. Un élément a de A est un* ***élément régulier à droite*** *de A si :*

$$\forall b \in A,\ ba = 0 \Rightarrow b = 0$$

Définition 2.48. *Soit G un groupe et $\tau \in \mathbb{F}_{2^k}[G]$. L'élément τ est une fonction de transfert si τ est régulier à droite.*

Théorème 2.9. *Soit A un anneau fini. Soit $a \in A$. Alors, a est régulier à droite si et seulement si a est inversible.*

Démonstration. Supposons que a est régulier. Soit $m\colon A \to A$ l'application de multiplication à droite par a, c'est-à-dire, $m(b) = ba$. L'application m est un endomorphisme du groupe additif A. Comme a est régulier à droite, son noyau est trivial. L'application m est donc injective. Comme m est une application injective d'un ensemble fini dans lui-même, m est bijective. En particulier, m est surjective. Il existe donc $s \in A$ tel que $sa = 1$. Montrons qu'on a aussi $as = 1$. En effet, comme $sa = 1$, on a aussi $asa = a$, ou encore $(as - 1)a = 0$. Comme a est régulier à droite, on en déduit que $as - 1 = 0$, c'est-à-dire $as = 1$. Cela montre bien que a est inversible si a est régulier à droite. L'implication réciproque est évidente. □

Corollaire 2.1. *Soit G un groupe fini et $\tau \in \mathbb{F}_{2^k}[G]$. Alors les conditions suivantes sont équivalentes :*

1. *τ est une fonction de transfert,*
2. *τ est inversible dans $\mathbb{F}_{2^k}[G]$, et*
3. *il existe un entier naturel n non nul tel que $\tau^n = 1$ dans $\mathbb{F}_{2^k}[G]$.*

Démonstration. Supposons que τ est une fonction de transfert. D'après la définition, τ est régulier à droite dans $\mathbb{F}_{2^k}[G]$. Il suit de l'énoncé précédent que τ est inversible dans $\mathbb{F}_{2^k}[G]$. Cela montre l'implication $1 \Rightarrow 2$.

Supposons que τ est inversible dans $\mathbb{F}_{2^k}[G]$. Cela veut dire que τ appartient au groupe multiplicatif de l'anneau $\mathbb{F}_{2^k}[G]$. Ce dernier groupe étant fini, τ est d'ordre fini, c'est-à-dire qu'il existe un entier naturel n non nul tel que $\tau^n = 1$ dans $\mathbb{F}_{2^k}[G]$. Cela montre l'implication $2 \Rightarrow 3$.

Supposons qu'il existe un entier naturel n non nul tel que $\tau^n = 1$ dans $\mathbb{F}_{2^k}[G]$. Montrons que τ est régulier à droite dans $\mathbb{F}_{2^k}[G]$. Soit $a \in \mathbb{F}_{2^k}[G]$ avec $a\tau = 0$. On en déduit que $0 = a\tau \cdot \tau^{n-1} = a\tau^n = a$, et τ est régulier à droite dans $\mathbb{F}_{2^k}[G]$. Cela montre l'implication $3 \Rightarrow 1$. □

A l'aide du théorème de Cayley-Hamilton, nous pouvons en déduire un critère basé sur le déterminant d'une matrice.

Soit G un groupe de cardinal n. Soit $L(\mathbb{F}_{2^k}[G])$ l'ensemble des applications $\mathbb{F}_{2^k}$-linéaires de $\mathbb{F}_{2^k}[G]$ dans $\mathbb{F}_{2^k}[G]$. Soit $M_n(\mathbb{F}_{2^k})$ l'ensemble des matrices carrées de taille n sur $\mathbb{F}_{2^k}$. Soit $\tau \in \mathbb{F}_{2^k}[G]$ et soit m_τ l'application de multiplication à droite par τ, c'est-à-dire :

$$\begin{aligned} m_\tau : \mathbb{F}_{2^k}[G] &\to \mathbb{F}_{2^k}[G] \\ \sigma &\mapsto \sigma\tau \end{aligned}$$

m_τ est $\mathbb{F}_{2^k}$-linéaire. Soit :

$$\begin{aligned}\phi : \mathbb{F}_{2^k}[G] &\to L(\mathbb{F}_{2^k}[G]) \\ \tau &\mapsto m_\tau\end{aligned}$$

Soit $\chi : L(\mathbb{F}_{2^k}[G]) \to M_n(\mathbb{F}_{2^k})$ l'application qui associe à une application linéaire sa matrice dans la base G de $\mathbb{F}_{2^k}[G]$. Soit $\varphi : \mathbb{F}_{2^k}[G] \to M_n(\mathbb{F}_{2^k})$ telle que $\varphi = \chi \circ \phi$. L'application φ est un morphisme d'anneau injectif. On a :

$$\mathbb{F}_{2^k}[G] \cong Im(\varphi) \subseteq M_n(\mathbb{F}_{2^k})$$

Ainsi, on peut identifier $\tau \in \mathbb{F}_{2^k}[G]$ avec $\varphi(\tau) \in M_n(\mathbb{F}_{2^k})$.

Théorème 2.10. *Soit $\tau \in \mathbb{F}_{2^k}[G]$.*
Si τ est inversible dans $M_n(\mathbb{F}_{2^k})$ alors τ est inversible dans $\mathbb{F}_{2^k}[G]$.

Démonstration. Soit $P_C(X) = det(XI_n - \tau) \in \mathbb{F}_{2^k}[X]$ le polynôme caractéristique de τ. En utilisant le théorème de Cayley-Hamilton, on a :

$$P_C(\tau) = 0$$

Or :

$$\begin{aligned}P_C(\tau) = \tau^n + a_{n-1}\tau^{n-1} + \ldots + a_1\tau + \det(\tau)\tau^0 &= 0 \\ \Rightarrow \tau^n + a_{n-1}\tau^{n-1} + \ldots + a_1\tau &= \det(\tau)\tau^0 \\ \Rightarrow (\tau^{n-1} + a_{n-1}\tau^{n-2} + \ldots + a_1)\tau &= \det(\tau)\tau^0 \\ \Rightarrow \tau^{-1} = \frac{1}{\det(\tau)}.(\tau^{n-1} + a_{n-1}\tau^{n-2} + \ldots + a_1)&\end{aligned}$$

Or toutes les puissances de τ sont dans $\mathbb{F}_{2^k}[G]$ donc $\tau^{-1} \in \mathbb{F}_{2^k}[G]$. Donc τ est inversible dans $\mathbb{F}_{2^k}[G]$. □

Le critère consiste donc, pour une fonction $\tau \in \mathbb{F}_{2^k}[G]$, à calculer sa matrice associée dans $M_n(\mathbb{F}_{2^k})$ et à tester si son déterminant est non nul. S'il est non nul, alors τ peut être utilisée comme une fonction de transfert pour un code convolutif sur le groupe G considéré.

Chapitre 3

Codes convolutifs définis sur un groupe non-commutatif

Dans ce chapitre, nous allons définir théoriquement les codes convolutifs sur des groupes non-commutatifs. Nous avons vu dans le chapitre 1 que les codes convolutifs de rendement $\frac{1}{n}$ sont définis par la convolution d'un message u par une fonction de transfert τ dans le groupe des entiers relatifs $\mathbb{Z}$.

Pour tenter de construire des codes avec des propriétés cryptographiques, nous allons remplacer ce groupe $\mathbb{Z}$ par un groupe non-commutatif, fini ou infini. Ces codes seront une généralisation des codes convolutifs classiques de rendement $\frac{1}{n}$.

3.1 Codes convolutifs sur un groupe non-commutatif infini

Dans le chapitre 4, nous allons remplacer le groupe $\mathbb{Z}$ par le groupe diédral infini D_∞. La convolution sera donc définie sur ce groupe non-commutatif. Dans le chapitre 1, l'encodage est aussi le produit de deux éléments non nuls de $\mathbb{F}_2[X]$ qui est l'algèbre du groupe $\mathbb{Z}$. Sur des groupes G non-commutatifs infinis, l'encodage sera, de la même façon, le produit de deux éléments non nuls de $\mathbb{F}_2[G]$, l'algèbre du groupe G. Cependant, dans une algèbre de groupe quelconque, le produit de deux éléments non nuls de cette algèbre peut être nul. Les éléments d'un tel produit sont des diviseurs de zéro, comme nous l'avons vu au chapitre 2. Encoder un message avec un tel diviseur de zéro poserait des problèmes de décodage, car, à un mot de code, pourrait correspondre plusieurs messages. Un élément qui n'est jamais diviseur de zéro est un élément régulier de l'algèbre de groupe $\mathbb{F}_2[G]$. Nous avons donc la définition suivante.

Définition 3.1. *Une* ***fonction de transfert*** *d'un code sur un groupe G est un élément régulier à droite de l'algèbre $\mathbb{F}_2[G]$ du groupe G.*

L'encodage est simplement le produit du message et de la fonction de transfert dans l'algèbre de groupe.

Pour avoir un code convolutif avec un rendement au moins aussi bon que celui des codes convolutifs classiques, il faut trouver des éléments $g_i \in G$ du groupe et

un ordre sur ces éléments (correspondant à la numérotation par l'indice i), de sorte qu'on puisse envoyer le message $u \in \mathbb{F}_2^k$ dans $\mathbb{F}_2[G]$ de la façon suivante :

$$u = \sum_{i=0}^{k-1} u_i g_i$$

De plus, il est nécessaire que la fonction de transfert τ soit de support fini dans $\mathbb{F}_2[G]$. Enfin, pour avoir un rendement aussi bon que celui des codes convolutifs classiques, on aimerait que les indices i des bits c_i non nuls du mot de code $c \in \mathbb{F}_2[G]$ soient dans l'ensemble $\{0, \ldots, kn-1\}$.

L'encodage serait donc ainsi réalisé par le schéma 3.1.

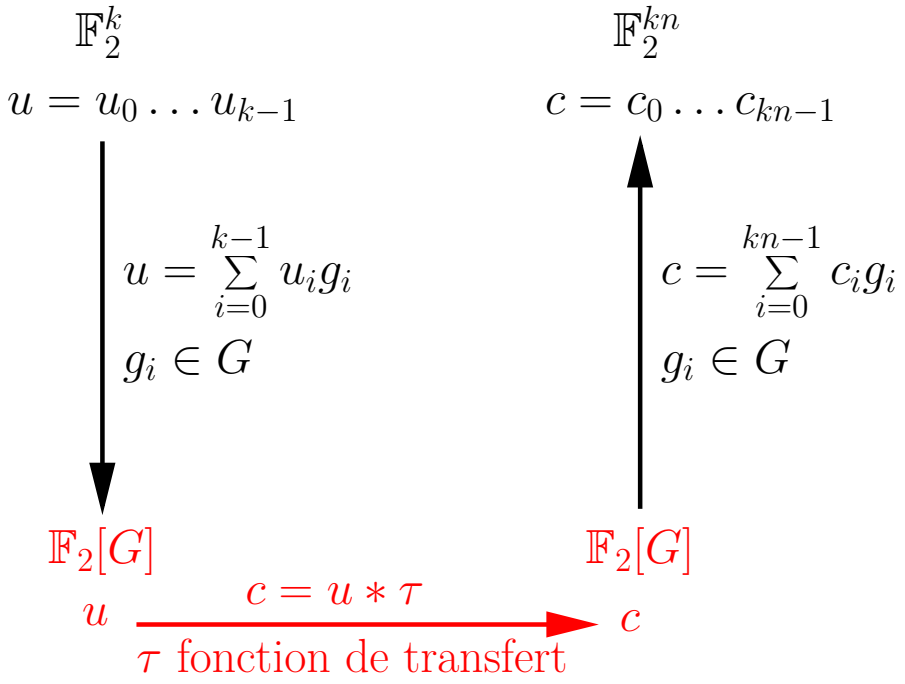


FIGURE 3.1 – Encodage sur un groupe non-commutatif infini

3.2 Codes convolutifs sur un groupe non-commutatif fini

Prenons maintenant un groupe non-commutatif fini de cardinal n. Nous allons donc utiliser un code convolutif en bloc, c'est-à-dire que chaque message U de longueur l sera divisé en blocs u de longueur $k < n$. Nous allons nous intéresser ici au codage d'un seul bloc u de longueur k.

La première étape consiste à envoyer le message $u \in \mathbb{F}_2^k$ dans l'algèbre $\mathbb{F}_2[G]$ du groupe G. Pour cela, nous allons utiliser des sous-ensembles $E = \{e_0, \ldots, e_{k-1}\}$ de longueur k du groupe G appelés **intervalles d'encodage**. Ainsi, le message u est défini sur l'algèbre du groupe, $\mathbb{F}_2[G]$ par :

$$u = \sum_{i=0}^{k-1} u_i e_i, \ e_i \in E$$

Pour espérer avoir une opération d'encodage qui soit une opération de chiffrement, nous allons faire varier ces intervalles E dans le temps, de sorte que chaque message u soit encodé sur un sous-ensemble E différent à chaque encodage.

Définition 3.2. *Un code convolutif sur un groupe fini est dit* ***variable*** *si chaque message u est envoyé sur un sous-ensemble E du groupe G différent à chaque encodage.*

Afin d'avoir un système de chiffrement à clé symétrique, nous allons générer les intervalles d'encodage E de manière chaotique, en parcourant les éléments, à partir d'un état initial d'un système dynamique, qui sera la clé symétrique du système cryptographique. Nous allons tout d'abord définir une famille chaotique de parcours des éléments d'un groupe fini G.

Définition 3.3. *Une famille de parcours d'éléments d'un groupe fini G est dite* ***chaotique*** *si elle satisfait les deux conditions suivantes :*

1. *Une grande sensibilité aux conditions initiales, c'est-à-dire, soient a, b deux points initiaux d'un système dynamique, avec $a \neq b$ et soit f la fonction de parcours des éléments du groupe G, telle que :*

$$\begin{aligned} f : \mathbb{N} \times A &\to G \\ (i, a) &\mapsto f_i(a) \end{aligned}$$

   avec A l'ensemble des points sur lequel est défini le système dynamique. Donc $f_i(a) \in G$ est le i-ème élément calculé à partir de $a \in A$ et l'ensemble des $f_i(a)$ est donc la famille de parcours des éléments de G considérée. On définit la fonction $\phi_i(a, b)$ suivante :

$$\forall a, b,\ \phi_i(a, b) = \begin{cases} 1 & si\ f_i(a) = f_i(b) \\ 0 & sinon \end{cases}$$

   Alors on a :

$$\lim_{l \to \infty} \frac{1}{l} \sum_{i=0}^{l} \phi_i(a, b) = \frac{1}{n}$$

   avec $n = Card(G)$.

2. *Une uniforme répartition des éléments, c'est-à-dire que si on considère un parcours fini $f_i(a)$ obtenu à partir d'un $a \in A$ quelconque, alors, si on note o_g le nombre d'occurrences d'un élément g dans le parcours $f_i(a)$ des éléments et L le nombre total d'éléments parcourus, nous avons :*

$$\forall g \in G,\ \lim_{L \to \infty} \frac{o_g}{L} = \frac{1}{n}$$

   avec $n = Card(G)$.

Pour avoir des intervalles d'encodage dit chaotiques, on génère un parcours issu d'une famille de parcours chaotique, que l'on divise en intervalles sans répétitions d'éléments, en supprimant des éléments ou non. Nous verrons la génération chaotique

des intervalles d'encodage dans deux exemples de groupes finis non-commutatifs aux chapitres 5 et 7.

Nous allons ensuite encoder ce message u de $\mathbb{F}_2[G]$ en effectuant une convolution avec une fonction de transfert, définie comme dans la définition 3.1. Pour savoir si cette fonction est régulière, nous utilisons le critère 2.5 du chapitre 2. La fonction de transfert τ est donc également définie sur $\mathbb{F}_2[G]$ comme suit.

$$\tau = \sum_{i=0}^{n-1} \tau_i g_i, \ g_i \in G$$

Nous convoluons ces deux éléments de $\mathbb{F}_2[G]$ pour obtenir $c \in \mathbb{F}_2[G]$ tel que :

$$c = \sum_{i=0}^{n-1} c_i g_i, \ g_i \in G$$

Les bits $c_i \in \mathbb{F}_2$ forment le mot de code qui est envoyé sur le canal. En résumé, l'encodage est effectué par le schéma 3.2.

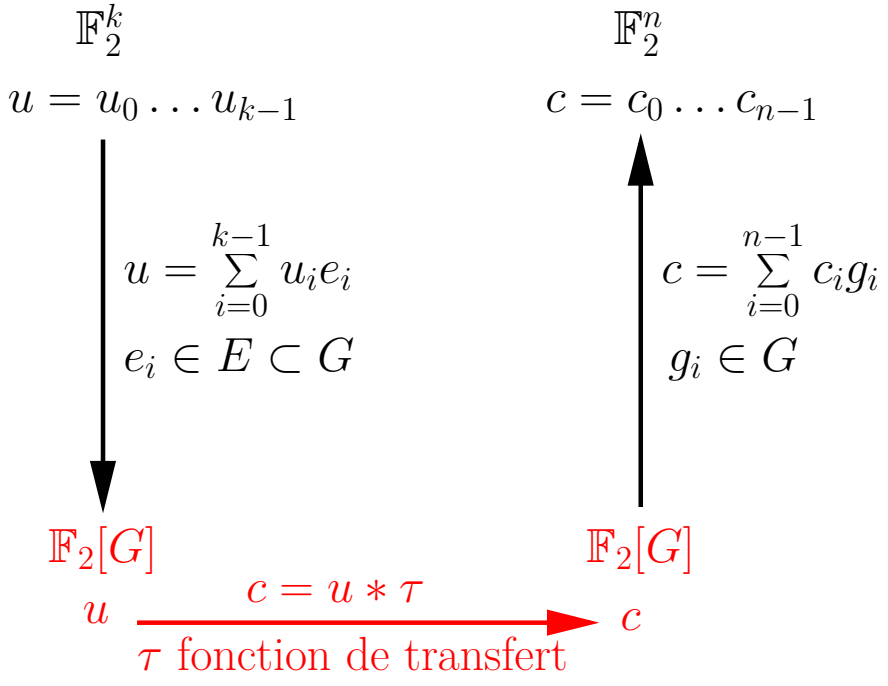


FIGURE 3.2 – Encodage sur un groupe non-commutatif fini

Le décodage de ces codes convolutifs sur un groupe non-commutatif fini est réalisé par « l'inverse » de l'encodage. C'est-à-dire qu'il y a tout d'abord une étape de correction des erreurs, qui permet, à partir du mot reçu r, de retrouver le mot de code $c \in \mathbb{F}_2^n$. Puis ce mot de code est envoyé dans l'algèbre du groupe G en utilisant l'égalité :

$$c = \sum_{i=0}^{n-1} c_i g_i, \ g_i \in G$$

Ensuite, on effectue la convolution de ce mot de code avec τ^{-1}, l'inverse de la fonction de transfert dans l'algèbre de groupe. On retrouve ainsi le message $u \in \mathbb{F}_2[G]$. Enfin, on utilise les intervalles d'encodage E pour retrouver le message $u \in \mathbb{F}_2^k$, en utilisant le fait que :

$$u = \sum_{i=0}^{k-1} u_i e_i, \ e_i \in E$$

Par conséquent, le décodage peut se résumer au schéma 3.3.

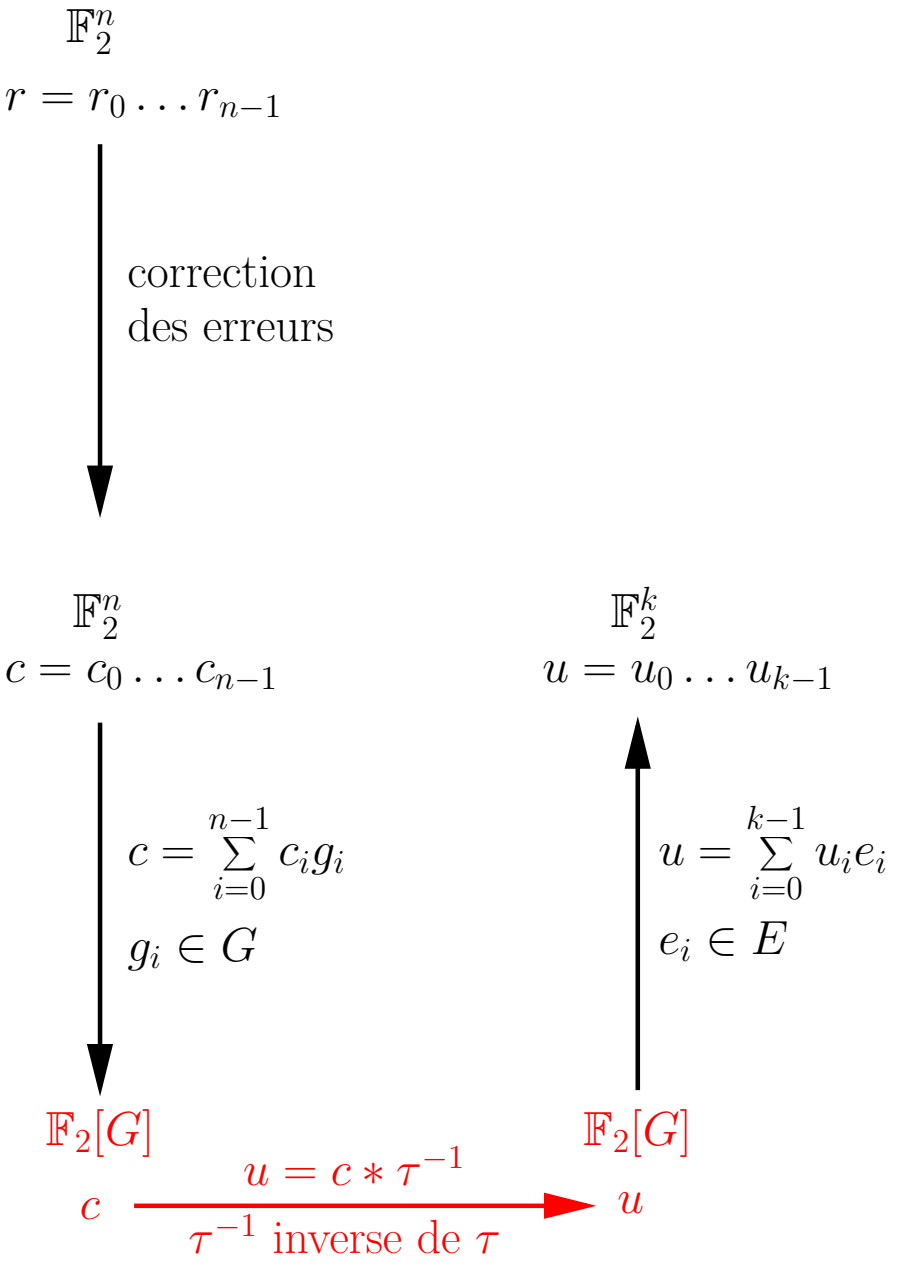


FIGURE 3.3 – Décodage sur un groupe non-commutatif fini

Chapitre 4

Codes convolutifs sur le groupe diédral infini

Nous avons vu dans le chapitre 1 que les codes convolutifs classiques de rendement $\frac{1}{n}$ pouvaient se ramener à la convolution du message et d'une fonction de transfert sur le groupe des entiers relatifs $\mathbb{Z}$. Nous allons, dans ce chapitre, remplacer $\mathbb{Z}$ par le groupe diédral infini D_∞.

4.1 Groupe diédral infini

Tout d'abord, rappelons la définition du groupe diédral fini D_n.

Définition 4.1. *Le* ***groupe diédral fini*** D_n *d'ordre* $2n$ *a pour présentation :*

$$D_n = \langle r, s \mid r^n = 1, s^2 = 1, srs = r^{-1} \rangle$$

pour tout entier naturel n *non nul.*

Définition 4.2. *Le* ***groupe diédral infini*** D_∞ *est donc naturellement défini par la présentation suivante :*

$$D_\infty = \langle r, s \mid s^2 = 1, srs = r^{-1} \rangle$$

D_∞ peut également être vu comme le produit semi-direct $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$. En effet, un élément de D_∞ peut s'écrire de façon unique comme $r^a s^\varepsilon$, avec $a \in \mathbb{Z}$ et $\varepsilon \in \{0, 1\} = \mathbb{Z}/2\mathbb{Z}$. Nous en déduisons que D_∞ peut s'identifier avec l'ensemble $\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$. La loi induite sur cet ensemble est donnée par :

$$\begin{aligned}
(a, 0) \circ (b, 0) &= (a + b, 0) \\
(a, 0) \circ (b, 1) &= (a + b, 1) \\
(a, 1) \circ (b, 0) &= (a - b, 1) \\
(a, 1) \circ (b, 1) &= (a - b, 0),
\end{aligned}$$

avec $a, b \in \mathbb{Z}$. C'est exactement la loi du produit semi-direct $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$ sur l'ensemble $\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$. On peut voir, de plus, que le groupe $\mathbb{Z} = \mathbb{Z} \times \{0\}$ est un sous-groupe distingué d'indice 2 dans $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$.

Si on revient maintenant à la présentation initiale de D_∞, on peut écrire $x = rs$ et $y = s$ et on a la présentation équivalente suivante :

$$D_\infty = \langle x, y \mid x^2 = y^2 = 1 \rangle$$

Ainsi, cela montre que D_∞ peut aussi s'identifier au produit libre $\mathbb{Z}/2\mathbb{Z} * \mathbb{Z}/2\mathbb{Z}$ du groupe $\mathbb{Z}/2\mathbb{Z}$ avec lui-même.

4.2 Algèbre de groupe

Soit $\mathbb{F}_2[D_\infty]$ la $\mathbb{F}_2$-algèbre du groupe D_∞, c'est-à-dire que $\mathbb{F}_2[D_\infty]$ est le $\mathbb{F}_2$-espace vectoriel ayant pour base D_∞. Nous constatons que $\mathbb{F}_2[D_\infty]$ n'est rien d'autre que l'ensemble des fonctions à support fini de D_∞ à valeurs dans $\mathbb{F}_2$. Le produit dans $\mathbb{F}_2[D_\infty]$ coïncide avec la convolution de fonctions.

Comme on veut voir $\mathbb{F}_2[D_\infty]$ comme une algèbre polynomiale non-commutative, nous préférons écrire X au lieu de x et Y au lieu de y. Alors :

$$\mathbb{F}_2[D_\infty] = \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$$

avec $\mathbb{F}_2\{X, Y\}$ la $\mathbb{F}_2$-algèbre des polynômes non-commutatifs en les variables X et Y et $(X^2 - 1, Y^2 - 1)$ est l'idéal bilatère de $\mathbb{F}_2\{X, Y\}$ généré par $X^2 - 1$ et $Y^2 - 1$.

Comme $r = xy$ et $s = y$ dans D_∞, tout élément $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ s'écrit de façon unique comme :

$$P = \sum_{(i,j) \in \mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}} c_{i,j} (XY)^i Y^j$$

avec $c_{i,j} \in \mathbb{F}_2$.
Ou, par symétrie et le fait que $yx = (xy)^{-1}$, tout élément $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ s'écrit de façon unique comme

$$P = \sum_{(i,j) \in \mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}} c_{i,j} (XY)^i X^j \tag{4.1}$$

avec $c_{i,j} \in \mathbb{F}_2$.
En utilisant le fait que $rs = sr^{-1}$ et $r^{-1}s = sr$, on peut aussi en déduire que tout élément $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ s'écrit de façon unique comme :

$$P = \sum_{(i,j) \in \mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}} c_{i,j} Y^j (XY)^i$$

avec $c_{i,j} \in \mathbb{F}_2$.
Et alors de nouveau, tout élément $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ s'écrit de façon unique comme :

$$P = \sum_{(i,j) \in \mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}} c_{i,j} X^j (XY)^i$$

avec $c_{i,j} \in \mathbb{F}_2$.

On peut voir que la sous-algèbre $\mathbb{F}_2[XY, YX]$ de $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ générée par XY et YX est commutative et isomorphe à l'algèbre $\mathbb{F}_2[T, T^{-1}]$ qui n'est rien d'autre que la $\mathbb{F}_2$-algèbre $\mathbb{F}_2[\mathbb{Z}]$ du groupe $\mathbb{Z}$.

Théorème 4.1. $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ *est un* $\mathbb{F}_2[XY,YX]$*-module à gauche libre de base* $1, X$.

Démonstration. Ce théorème est la conséquence directe de l'unicité de l'écriture de l'équation 4.1. □

Cela signifie explicitement que si on a $P \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$, alors il existe $A, B \in \mathbb{F}_2[XY,YX]$ tels que :

$$P = A + BX$$

dans $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. De plus, A et B sont déterminés de façon unique par P.

4.3 Injectivité du code

Pour avoir un code correcteur d'erreurs sur le groupe non-commutatif D_∞ défini par la convolution, chaque mot de code doit être le résultat d'un et d'un seul message par la convolution, sinon il serait impossible de décoder chaque message de manière unique. Donc la convolution par la fonction de transfert doit être injective. Ce qui signifie que la fonction de transfert ne doit pas être un diviseur de zéro à droite dans l'anneau $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$, c'est-à-dire qu'elle doit être un élément régulier de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$:

Définition 4.3. *Soit R un anneau. Un élément P de R est un* ***élément régulier à droite*** *de R si :*

$$\forall C \in R,\ CP = 0 \Rightarrow C = 0$$

Donc un élément P d'un anneau R est un élément régulier à droite si et seulement si P n'est pas un diviseur de zéro à droite.

L'anneau $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ contient beaucoup de diviseurs de zéro. Les cas les plus simples sont $(1+X)$ et $(1+Y)$ car $(1+X)^2 = 0$ et $(1+Y)^2 = 0$. Donc tous les multiples à gauche ou à droite $P(1+X)$, $P(1+Y)$, $(1+X)P$ et $(1+Y)P$, pour tout P non nul dans $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$, sont des diviseurs de zéro.

Définissons l'anti-automorphisme :

$$\begin{aligned}\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1) &\to \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)\\ P &\mapsto \overline{P}\end{aligned}$$

par $\overline{X} = X$ et $\overline{Y} = Y$. Cela signifie que $\overline{PQ} = \overline{Q}.\overline{P}$ pour tous $P, Q \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. En particulier, on a $\overline{(XY)^n} = (YX)^n$ et donc $\overline{(XY)^n X} = \overline{X}.\overline{(XY)^n} = X(YX)^n = (XY)^n X$ pour tout $n \in \mathbb{N}$. On en déduit que :

$$\overline{A + BX} = \overline{A} + BX$$

pour tout $A, B \in \mathbb{F}_2[XY,YX]$. Comme $P \mapsto \overline{\overline{P}}$ est un automorphisme de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ fixant X et Y, c'est l'identité. Donc, on a :

$$\overline{\overline{P}} = P$$

pour tout $P \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. L'anti-automorphisme $P \mapsto \overline{P}$ est une anti-involution.

Proposition 4.1. *Pour tout $A \in \mathbb{F}_2[XY, YX]$, on a :*

$$AX = X\overline{A}$$

Démonstration. Il suffit de vérifier la propriété pour $A = (XY)^n$ et pour $A = (YX)^n$, où n est un entier naturel.
Si $A = (XY)^n$ alors :

$$\begin{aligned}(XY)^n X &= X(YX)^n \\ &= X\overline{(XY)^n}\end{aligned}$$

Si $A = (YX)^n$ avec $n \neq 0$, on a :

$$\begin{aligned}(YX)^n X &= Y(XY)^{n-1} \\ &= XXY(XY)^{n-1} \\ &= X(XY)^n \\ &= X\overline{(YX)^n}\end{aligned}$$

□

Théorème 4.2. *Tout élément non nul de la sous-algèbre $\mathbb{F}_2[XY, YX]$ de $\mathbb{F}_2\{X, Y\}$-$/(X^2 - 1, Y^2 - 1)$ est un élément régulier à droite de $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$.*

Démonstration. Soit $P \in \mathbb{F}_2[XY, YX]$ non nul. Nous voulons montrer que P est un élément régulier à droite de $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$. Soit $C \in \mathbb{F}_2\{X, Y\}$-$/(X^2 - 1, Y^2 - 1)$ tel que $CP = 0$. Comme nous l'avons montré précédemment, il existe $A, B \in \mathbb{F}_2[XY, YX]$ tels que $C = A + BX$. On a :

$$0 + 0X = CP = (A + BX)P = AP + BXP = AP + B\overline{P}X$$

Comme AP et $B\overline{P}$ sont des éléments de $\mathbb{F}_2[XY, YX]$, on en déduit que $AP = 0$ et $B\overline{P} = 0$, Or, l'algèbre $\mathbb{F}_2[XY, YX]$ est isomorphe à l'algèbre $\mathbb{F}_2[T, T^{-1}]$ et est donc intègre. Comme $P \neq 0$ et, par suite, $\overline{P} \neq 0$, on a nécessairement $A = 0$ et $B = 0$, c'est-à-dire $C = 0$. □

Ce théorème 4.2 est une conséquence du fait que $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ est un module libre à gauche sous l'anneau $\mathbb{F}_2[XY, YX]$ démontré au théorème 4.1.

Pour tout $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$, on définit $N(P) \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ par :

$$N(P) = P\overline{P}$$

La proposition 4.1 ci-dessus est utilisée pour démontrer le théorème suivant :

Théorème 4.3. *Soit $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ noté $P = A + BX$ avec $A, B \in \mathbb{F}_2[XY, YX]$. Alors :*

$$N(P) = A\overline{A} + B\overline{B}$$

En particulier, $N(P) \in \mathbb{F}_2[XY, YX]$ et $N(P) = N(\overline{P})$.

Démonstration. On a :

$$\begin{aligned}
P\overline{P} &= (A+BX)(\overline{A+BX}) \\
&= (A+BX)(\overline{A}+BX) \\
&= A\overline{A}+BX\overline{A}+ABX+BXBX \\
&= A\overline{A}+BAX+ABX+BX\overline{X}\overline{B} \text{ (proposition 4.1)} \\
&= A\overline{A}+BAX+BAX+B\overline{B} \text{ car } \mathbb{F}_2[XY,YX] \text{ est commutatif et } X^2=1 \\
&= A\overline{A}+B\overline{B}.
\end{aligned}$$

Cela prouve que $N(P) = A\overline{A}+B\overline{B} \in \mathbb{F}_2[XY,YX]$. Comme $\overline{P} = \overline{A}+BX$ on a aussi $N(\overline{P}) = N(P)$. □

Le théorème précédent suggère que N est une norme pour l'extension d'anneau non-commutative de degré 2 de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ sur $\mathbb{F}_2[XY,YX]$. Le théorème suivant nous permet de décider si un élément P de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ est un diviseur de zéro à droite ou non :

Théorème 4.4. *Soit P un élément de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. Alors P est un élément régulier à droite si et seulement si $N(P) \neq 0$.*

Démonstration. Soit P un élément de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. On suppose que $P \neq 0$.

1. Si $N(P) = 0$ alors $N(\overline{P}) = 0$ par le théorème précédent. On en déduit que $\overline{P}P = 0$ c'est-à-dire que P n'est pas un élément régulier à droite de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$.
2. Si $N(P) \neq 0$ alors, comme tous les éléments de $\mathbb{F}_2[XY,YX]$ sont des éléments réguliers à droite de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$, on a :

   $$\forall C \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1), C(P\overline{P}) = 0 \Rightarrow C = 0$$

   Donc :

   $$CP = 0 \Rightarrow (CP)\overline{P} = 0 \Rightarrow C(P\overline{P}) = 0 \Rightarrow C = 0$$

   Donc P est un élément régulier à droite de $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$.

□

Le théorème précédent nous permet de déterminer si un élément $\tau \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ peut être utilisé comme fonction de transfert pour un code convolutif sur le groupe diédral infini. En effet, si $N(\tau) \neq 0$ alors l'application multiplication-par-τ

$$\begin{aligned}
\times\tau : \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1) &\to \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1) \\
u &\mapsto u \times \tau
\end{aligned}$$

est injective, car τ est régulier à droite.

Exemple 4.1. *Soit* $\tau = YX + Y + 1 + X + XY + XYX$. *On a* $\overline{\tau} = XY + Y + 1 + X + YX + XYX$. *On calcule* $N(\tau)$:

$$\begin{aligned}
N(\tau) &= \tau\overline{\tau} \\
&= (YX + Y + 1 + X + XY + XYX)(XY + Y + 1 + X + YX + XYX) \\
&= 1 + YXY + YX + Y + YXYX + X \\
&+ YXY + 1 + Y + YX + X + YXYX \\
&+ XY + Y + 1 + X + YX + XYX \\
&+ XYXY + X + XY + XYX + 1 + XYXYX \\
&+ X + XYXY + XYX + XY + XYXYX + 1 \\
&= 0
\end{aligned}$$

$N(\tau) = 0$ *donc* τ *ne peut pas être utilisé comme une fonction de transfert.*

Exemple 4.2. *Soit* $\tau = Y + 1 + XY$. *On a* $\overline{\tau} = Y + 1 + YX$. *On calcule* $N(\tau)$:

$$\begin{aligned}
N(\tau) &= \tau\overline{\tau} \\
&= (Y + 1 + XY)(Y + 1 + YX) \\
&= 1 + Y + X \\
&+ Y + 1 + YX \\
&+ X + XY + 1 \\
&= YX + 1 + XY \neq 0
\end{aligned}$$

$N(\tau) \neq 0$ *donc* τ *peut être utilisé comme fonction de transfert.*

4.4 Codage par un registre

Nous avons vu dans le chapitre 1 que les codes convolutifs classiques sont définis sur l'anneau des polynômes sur $\mathbb{F}_2[X]$ et ont la propriété que le produit de deux polynômes avec des puissances positives de X donne un polynôme composé de puissances positives de X. Dans notre cas, si nous utilisons l'ordre donné dans la table 4.1 avec les éléments "positifs" à droite du monôme 1 et les éléments "négatifs" à gauche du monôme 1, nous voyons que $X(XY) = Y$ et que le produit de deux éléments "positifs" peut donner un élément "négatif". Pour éviter ce problème, nous

...	YXYX	YXY	YX	Y	1	X	XY	XYX	XYXY	...

TABLE 4.1 – Ordre naturel des monômes de $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$

allons choisir une autre façon de placer les bits du message. Nous pouvons voir que si nous choisissons une fonction de transfert τ avec une longueur impaire m centrée en 1 et un message u avec aussi une longueur impaire k, centré en 1, la convolution de u et de τ ajoute $m - 1$ éléments de redondance. Ce choix de longueurs k et m impaires est une convention, car ainsi le bit du milieu d'un train de bits est bien défini. Nous pourrions également choisir des longueurs k et m paires, en spécifiant

une convention pour définir pour le bit du milieu.

On peut voir un exemple dans la table 4.2 avec $\tau = YX + Y + 1 + X + XY$ et $u = u_{-2}YX + u_{-1}Y + u_0 + u_1X + u_2XY$. Nous vérifions que τ est une bonne fonction de transfert :

$$\begin{aligned} N(\tau) &= (YX + Y + 1 + X + XY)(XY + Y + 1 + X + YX) \\ &= YXYX + YX + 1 + XY + XYXY \neq 0 \end{aligned}$$

Les éléments en gras dans ce tableau sont les éléments de redondance du mot de code dans cet exemple. On peut voir que cette redondance est distribuée de manière égale de chaque côté des éléments sur lesquels le message est défini. Donc la redondance se place à gauche et à droite du message et pas seulement à gauche ou à droite.

τ \ u	YX	Y	1	X	XY
YX	**YXYX**	**YXY**	YX	Y	1
Y	X	1	Y	YX	**YXY**
1	YX	Y	1	X	XY
X	**XYX**	XY	X	1	Y
XY	1	X	XY	**XYX**	**XYXY**

TABLE 4.2 – La redondance (en gras) se place à gauche et à droite

Par conséquent, nous avons la convolution suivante, avec $G = \mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z} \simeq D_\infty$ et $\forall a, b \in G$:

$$\begin{aligned} (u * \tau)(a, b) &= \sum_{(c,d)\in G} u(c, d)\tau((c, d)^{-1} \circ (a, b)) \\ &= \sum_{(c,0)\in G} u(c, 0)\tau((-c, 0) \circ (a, b)) + \sum_{(c,1)\in G} u(c, 1)\tau((c, 1) \circ (a, b)) \\ &= \sum_{(c,0)\in G} u(c, 0)\tau(a - c, b) + \sum_{(c,1)\in G} u(c, 1)\tau(c - a, b + 1) \end{aligned}$$

Donc :

$$\begin{aligned} (u * \tau)(a, 0) &= \sum_{(c,0)\in G} u(c, 0)\tau(a - c, 0) + \sum_{(c,1)\in G} u(c, 1)\tau(c - a, 1) \\ (u * \tau)(a, 1) &= \sum_{(c,0)\in G} u(c, 0)\tau(a - c, 1) + \sum_{(c,1)\in G} u(c, 1)\tau(c - a, 0) \end{aligned}$$

Nous utilisons la bijection entre G et $\mathbb{Z}$ suivante :

$$(a, b) \mapsto 2a + b$$

Donc :

$$\begin{aligned} (u * \tau)(2a) &= \sum_{c\in\mathbb{Z}} u(2c)\tau(2a - 2c) + \sum_{c\in\mathbb{Z}} u(2c + 1)\tau(2c - 2a + 1) \\ (u * \tau)(2a + 1) &= \sum_{c\in\mathbb{Z}} u(2c)\tau(2a - 2c + 1) + \sum_{c\in\mathbb{Z}} u(2c + 1)\tau(2c - 2a) \end{aligned}$$

Nous définissons maintenant une fonction $\tilde{\tau}$ dérivée de la fonction de transfert τ comme suit :

$$\begin{aligned} \forall a \in \mathbb{Z},\ \tilde{\tau}(2a+1) &= \tau(-2a-1) \\ \tilde{\tau}(2a) &= \tau(2a) \end{aligned}$$

Avec $\tilde{\tau}$, on a :

$$\begin{aligned} (u*\tau)(2a) &= \sum_{c\in\mathbb{Z}} u(2c)\tilde{\tau}(2a-2c) + \sum_{c\in\mathbb{Z}} u(2c+1)\tilde{\tau}(2a-(2c+1)) \\ (u*\tau)(2a+1) &= \sum_{c\in\mathbb{Z}} u(2c)\tilde{\tau}(2c-(2a+1)) + \sum_{c\in\mathbb{Z}} u(2c+1)\tilde{\tau}(2c+1-(2a+1)) \end{aligned}$$

Nous pouvons regrouper les deux sommes en une comme suit :

$$\begin{aligned} (u*\tau)(2a) &= \sum_{c\in\mathbb{Z}} u(c)\tilde{\tau}(2a-c) \\ (u*\tau)(2a+1) &= \sum_{c\in\mathbb{Z}} u(c)\tilde{\tau}(c-(2a+1)) \end{aligned}$$

On peut voir que les bits d'indice pair du mot de code $u*\tau$ sont le résultat de la convolution classique de u et $\tilde{\tau}$ et que les bits d'indice impair sont le résultat de la convolution de u et le réciproque de $\tilde{\tau}$. En terme de codage par un registre à décalage, nous avons le circuit de la figure 4.1.

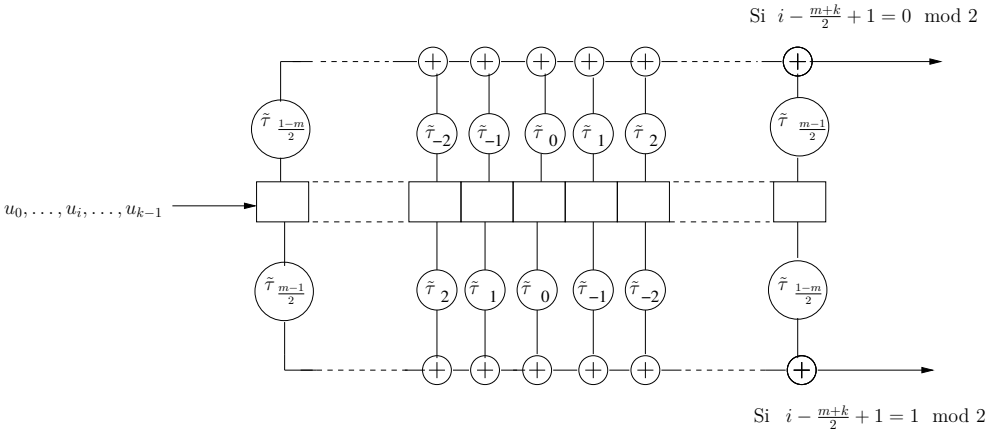


FIGURE 4.1 – Circuit d'encodage pour les codes convolutifs sur D_∞

Nous allons maintenant voir un exemple d'encodage avec $u = [1,0,1,1,1]$ et $\tau = [0,1,1,0,1]$. Nous vérifions que τ peut être utilisé comme une fonction de transfert :

$$\begin{aligned} N(\tau) &= (Y+1+XY)(Y+1+YX) \\ &= YX+1+XY \neq 0 \end{aligned}$$

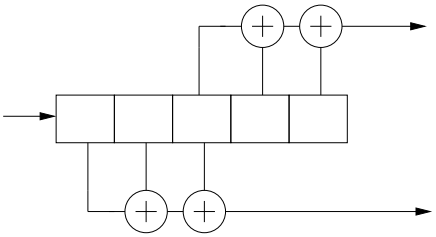


FIGURE 4.2 – Exemple de circuit d'encodage avec $\tau = [0, 1, 1, 0, 1]$

On a :

$$\begin{aligned}
\tilde{\tau}(-2) &= \tau(-2) = 0 \\
\tilde{\tau}(-1) &= \tau(1) = 0 \\
\tilde{\tau}(0) &= \tau(0) = 1 \\
\tilde{\tau}(1) &= \tau(-1) = 1 \\
\tilde{\tau}(2) &= \tau(2) = 1
\end{aligned}$$

Donc $\tilde{\tau} = [0, 0, 1, 1, 1]$. Et nous avons le circuit d'encodage de la figure 4.2. On rappelle que, lorsque $i - \frac{m+k}{2} + 1$ est pair, on utilise le circuit au-dessus du registre à décalage (correspondant à $\tilde{\tau}$) et quand $i - \frac{m+k}{2} + 1$ est impair, on utilise le circuit sous le registre à décalage (correspondant au réciproque de $\tilde{\tau}$). On a le tableau suivant avec i le temps, u_i les bits du message, m_1, m_2, m_3, m_4 la mémoire du registre ($m_0 = u_i$) et c_i le bit d'indice i du mot de code.

i	u_i	m_1	m_2	m_3	m_4	$i - \frac{m+k}{2} + 1$	c_i
0	1	0	0	0	0	-4	0
1	0	1	0	0	0	-3	1
2	1	0	1	0	0	-2	1
3	1	1	0	1	0	-1	0
4	1	1	1	0	1	0	0
5	0	1	1	1	0	1	0
6	0	0	1	1	1	2	1
7	0	0	0	1	1	3	0
8	0	0	0	0	1	4	1

Vérification :

$$\begin{aligned}
u &= [1, 0, 1, 1, 1] & \tau &= [0, 1, 1, 0, 1] \\
&= YX + 1 + X + XY & &= Y + 1 + XY
\end{aligned}$$

$$\begin{aligned}
u\tau &= (YX + 1 + X + XY)(Y + 1 + XY) \\
&= YXY + YX + 1 + Y + 1 + XY + XY + X + Y + X + XY + XYXY \\
&= YXY + YX + XY + XYXY
\end{aligned}$$

$YXYX$	YXY	YX	Y	1	X	XY	XYX	$XYXY$
0	1	1	0	0	0	1	0	1

On retrouve bien le même mot de code.

Notre façon d'encoder peut être mise sous la forme de plusieurs registres à décalage avec la même mémoire, où les bits sortants sont multiplexés à chaque instant i comme pour les codes convolutifs classiques (figure 1.2).

Nous savons que les codes convolutifs classiques de rendement $\frac{1}{n}$ peuvent toujours être réduits à une matrice de transfert de la forme suivante :

$$G_T = (g_0(X)\ g_1(X)\ \dots\ g_{n-1}(X))$$

avec $g_i(X) \in \mathbb{F}_2[X], \forall i$. La fonction de transfert τ peut s'exprimer comme :

$$\tau(X) = g_0(X^n) + Xg_1(X^n) + \ldots + X^{n-1}g_{n-1}(X^n)$$

et la convolution s'effectue avec $u(X^n)$.

Dans le cas des codes sur le groupe diédral infini, nous avons dit précédemment que la redondance est ajoutée à gauche et à droite, donc on doit utiliser un code de rendement minimal $\frac{1}{3}$. Le lien entre la matrice de transfert et la fonction de transfert est explicité dans le théorème suivant :

Théorème 4.5. *Soit G_T la matrice de transfert de la forme suivante (avec n impair) :*

$$G_T = \left(g_{\frac{1-n}{2}}(X,Y)\ g_{\frac{3-n}{2}}(X,Y)\ \dots\ g_0(X,Y)\ \dots\ g_{\frac{n-3}{2}}(X,Y)\ g_{\frac{n-1}{2}}(X,Y)\right)$$

avec $g_i(X,Y) \in \mathbb{F}_2\{X,Y\}/(X^2-1, Y^2-1), \forall i$. La fonction de transfert s'exprime comme :

$$\tau(X,Y) = \sum_{k=\frac{1-n}{2}}^{\frac{n-1}{2}} g_k\left((XY)^{\frac{n-1}{2}}X, (YX)^{\frac{n-1}{2}}Y\right)\phi(k)$$

avec ϕ la fonction :

$$\begin{aligned} \phi : \mathbb{Z} &\to \mathbb{F}_2\{X,Y\}/(X^2-1, Y^2-1) \\ y = 2a+b,\ b \in \mathbb{Z}/2\mathbb{Z} &\mapsto (XY)^a X^b \end{aligned}$$

La convolution est maintenant réalisée avec $u\left((XY)^{\frac{n-1}{2}}X, (YX)^{\frac{n-1}{2}}Y\right)$.

Démonstration. La fonction de transfert τ permet de séparer les éléments de G_T en les associant chacun à un monôme de $\mathbb{F}_2\{X,Y\}/(X^2-1, Y^2-1)$ différent. En effet, on a deux cas différents, selon la valeur de $n \mod 4$:

— Si $n = 1 \mod 4$, alors $\frac{1-n}{2} = 0 \mod 2$, donc $\frac{1-n}{2} = 2a+b \Rightarrow a = \frac{1-n}{4}$ et $b = 0$.

Donc

$$\begin{aligned}\tau(X,Y) = & g_{\frac{1-n}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{1-n}{4}} \\ & + g_{\frac{3-n}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{1-n}{4}}X \\ & + \dots \\ & + g_0\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right) \\ & + \dots \\ & + g_{\frac{n-3}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{n-5}{4}}X \\ & + g_{\frac{n-1}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{n-1}{4}}\end{aligned}$$

— Si $n = 3 \mod 4$, alors $\frac{1-n}{2} = 1 \mod 2$, donc $\frac{1-n}{2} = 2a + b \Rightarrow a = -\frac{n+1}{4}$ et $b = 1$.
Donc

$$\begin{aligned}\tau(X,Y) = & g_{\frac{1-n}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{-n-1}{4}}X \\ & + g_{\frac{3-n}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{3-n}{4}} \\ & + \dots \\ & + g_0\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right) \\ & + \dots \\ & + g_{\frac{n-3}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{n-3}{4}} \\ & + g_{\frac{n-1}{2}}\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)(XY)^{\frac{n-3}{4}}X\end{aligned}$$

Ainsi, chaque élément de G_T est bien devant un et un seul monôme de $\mathbb{F}_2\{X,Y\}$-$/(X^2-1, Y^2-1)$. On peut regrouper ces deux formules en une seule en définissant la fonction ϕ suivante :

$$\begin{aligned}\phi : \mathbb{Z} &\to \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1) \\ y = 2a+b,\ b \in \mathbb{Z}/2\mathbb{Z} &\mapsto (XY)^a X^b\end{aligned}$$

Ainsi

$$\begin{aligned}c(X,Y) &= u\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right) \times \tau(X,Y) \\ &= \sum_{k=\frac{1-n}{2}}^{\frac{n-1}{2}} u\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right) g_k\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)\phi(k) \\ &= \sum_{k=\frac{1-n}{2}}^{\frac{n-1}{2}} (ug_k)\left((XY)^{\frac{n-1}{2}}X,(YX)^{\frac{n-1}{2}}Y\right)\phi(k)\end{aligned}$$

et on retrouve chaque produit ug_k de la matrice $u \times G_T$. □

Pour avoir la forme systématique, il suffit que le registre à décalage central répète les bits du message, c'est-à-dire que l'indice du bit central du registre central soit égal à 1 et les autres indices soient égaux à zéro.

Exemple 4.3. *Soit*

$$\begin{aligned} G_T &= (011 \quad 010 \quad 110) \\ &= (1+X \quad 1 \quad 1+Y) \\ &= (g_{-1} \quad g_0 \quad g_1) \end{aligned}$$

On a bien un codage sous forme systématique car le registre central de G_T *est* 010 *ce qui correspond au fait que le polynôme* $g_0 = 1$*. On a ici* $m = 3$ *et* $n = 3$*. Calculons la fonction de transfert* τ *associée à* G_T*.*

$$\begin{aligned} \tau(X,Y) &= (1+XYX)Y + 1 + (1+YXY)X \\ &= Y + XYXY + 1 + X + YXYX \\ &= YXYX + Y + 1 + X + XYXY \end{aligned}$$

Vérifions que la multiplication par τ *est injective. Pour cela, calculons* $N(\tau)$*, la norme de* τ*.*

$$\begin{aligned} N(\tau) &= \tau\overline{\tau} \\ &= (YXYX + Y + 1 + X + XYXY)(XYXY + Y + 1 + X + YXYX) \\ &= 1 + YXYXY + YXYX + YXY + YXYXYXYX \\ &+ YXYXY + 1 + Y + YX + XYX \\ &+ XYXY + Y + 1 + X + YXYX \\ &+ YXY + XY + X + 1 + XYXYX \\ &+ XYXYXYXY + XYX + XYXY + XYXYX + 1 \\ &= YXYXYXYX + YX + 1 + XY + XYXYXYXY \\ &\neq 0 \end{aligned}$$

La multiplication par τ *est donc injective. Nous devons maintenant calculer* $\tilde{G_T}$ *pour construire le circuit.*

$$\begin{aligned} \tilde{g}_{-1} &= 1 + Y \\ \tilde{g}_0 &= 1 \\ \tilde{g}_1 &= 1 + X \end{aligned}$$

La figure 4.3 représente le circuit associé à G_T*. Nous avons volontairement séparé la mémoire en 3 pour bien voir le circuit, mais la mémoire est la même pour chaque petit circuit.*

Soit $u = 101$*. On a* $k = 3$*. Nous allons calculer la sortie (haut ou bas) de départ du circuit.*

$$i - \frac{m+k}{2} + 1 = 0 - \frac{3+3}{2} + 1 = -2 = 0 \mod 2$$

Nous allons donc commencer par la sortie du haut et alterner ensuite pour chaque bit qui entre dans la mémoire.

Le tableau 4.3 représente le codage de u par ce circuit avec i le temps, u_i *les bits du message,* m_1, m_2*, la mémoire (*$m_0 = u_i$*),* h_{-1}, h_0, h_1 *les bits de la sortie du haut et*

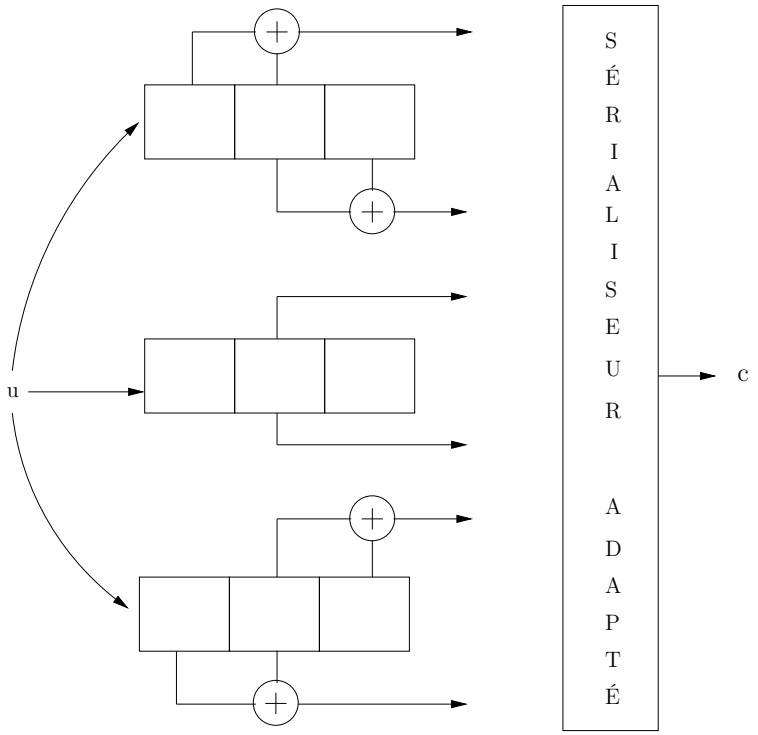


FIGURE 4.3 – Exemple de circuit d'encodage avec $G_T = (011 \;\; 010 \;\; 011)$

i	u_i	m_1	m_2	$i - \frac{m+k}{2} + 1$	h_{-1}	h_0	h_1	b_{-1}	b_0	b_1
0	1	0	0	-2	1	**0**	0			
1	0	1	0	-1				1	**1**	1
2	1	0	1	0	1	**0**	1			
3	0	1	0	1				1	**1**	1
4	0	0	1	2	0	**0**	1			

TABLE 4.3 – Codage de u avec le circuit associé à G_T

b_{-1}, b_0, b_1 les bits de la sortie du bas. On peut voir dans ce tableau les bits du message u en gras accompagnés, à chaque fois, à gauche et à droite de leur redondance.

Le sérialiseur va être adapté à ce codage sur D_∞ de façon simple : les bits issus de la sortie du haut vont être sérialisés du haut vers le bas (c'est-à-dire de la sortie de la convolution avec $g_{\frac{1-n}{2}}$ jusqu'à la sortie avec la convolution avec $g_{\frac{n-1}{2}}$) et les bits issus de la sortie du bas vont être sérialisés du bas vers le haut (c'est-à-dire de la sortie de la convolution avec $g_{\frac{n-1}{2}}$ jusqu'à la sortie avec la convolution avec $g_{\frac{1-n}{2}}$). Dans l'exemple cela donne :

$$c = 100\ 111\ 101\ 111\ 001$$

Nous vérifions que le résultat du circuit est identique à la convolution de u par τ.

$$\begin{aligned} c(X,Y) &= u(XYX, YXY) \times \tau(X,Y) \\ &= (XYX + YXY)(YXYX + Y + 1 + X + XYXY) \\ &= XYXYXYX + XYXY + XYX + XY + Y \\ &+ X + YX + YXY + YXYX + YXYXYXY \\ c &= 100111101111001 \end{aligned}$$

On retrouve bien le même mot de code.

On peut donc réaliser le codage sur le groupe non-commutatif D_∞ à l'aide d'un circuit électronique et d'un sérialiseur adaptés.

4.5 Décodage

Dans le cas des codes convolutifs classiques utilisant un canal binaire symétrique, nous avons vu que l'algorithme le plus utilisé est l'algorithme de Viterbi, qui est un algorithme utilisant le critère du maximum de vraisemblance. On rappelle que cet algorithme utilise un treillis dont les arêtes connectent chaque état de la mémoire à l'état suivant selon le bit entrant dans le registre à décalage à chaque instant.

Dans le cas des codes convolutifs sur le groupe diédral infini, on a deux treillis, un associé au circuit au-dessus du registre, l'autre associé au circuit sous le registre. Comme les deux circuits alternent dans l'encodage, les deux treillis alternent dans le décodage (figure 4.4). Donc on utilise un algorithme de Viterbi adapté pour décoder ces codes, qui utilise deux treillis au lieu d'un seul. Cependant, on ne décode plus en utilisant le critère de maximum de vraisemblance.

4.6 Propriétés

Premièrement, les codes convolutifs sur le groupe diédral infini ont une mémoire de longueur m (avec m la longueur de la fonction de transfert sous forme binaire) comme pour les codes convolutifs classiques. On a deux différentes sorties du registre à décalage mais elles alternent, donc il y a le même nombre d'opérations qu'avec un seul circuit. Donc la complexité d'encodage est la même que pour les codes

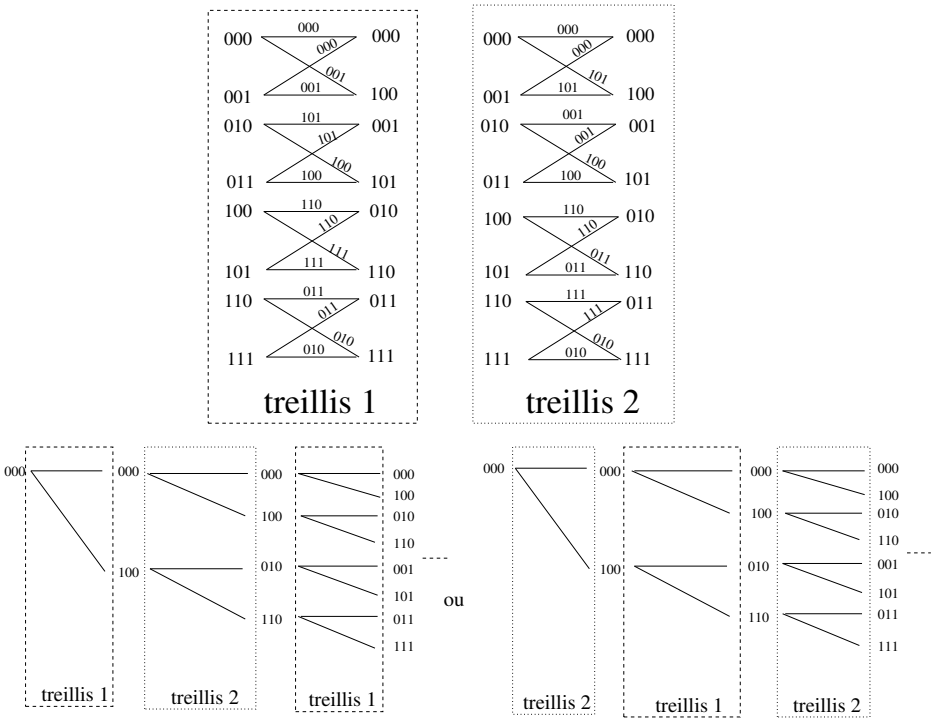


FIGURE 4.4 – Exemple de l'algorithme de Viterbi adapté

convolutifs classiques. Pour le décodage, on a deux treillis qui alternent, donc on a aussi la même complexité de décodage que pour les codes convolutifs classiques.

Deuxièmement, pour déterminer la distance libre de ces codes et pour la comparer avec la distance libre des codes convolutifs classiques, nous avons réalisé des tests dont les résultats sont dans le tableau 4.4. On voit que, pour la même longueur de mémoire et le même rendement, on a la même distance libre maximale pour les codes sur le groupe diédral infini que pour les codes convolutifs classiques. Nous savons que la distance libre d'un code dépend du treillis du code. Dans le cas des codes sur le groupe diédral infini, deux treillis alternent dans le décodage. Si on commence avec le treillis 1 alors on a une distance libre qui peut être différente de celle qu'on obtient si on commence par le treillis 2. Donc, la distance libre dépend du treillis par lequel on commence le décodage. Comme le treillis de départ dépend de la longueur du mot de code, on peut choisir par quel treillis on commence en adaptant la longueur du message. En effet :

1. Si on a une mémoire de taille $m = 1 \mod 4$
   - Si on a $k = 1 \mod 4$ alors $m+k = 2 \mod 4$, donc $-\frac{m+k}{2}+1 = 0 \mod 2$. Donc si on choisit un message de $k = 4q+1$, $q \in \mathbb{N}$, la première sortie du circuit sera toujours la sortie du haut.
   - Si on a $k = 3 \mod 4$ alors $m+k = 0 \mod 4$, donc $-\frac{m+k}{2}+1 = 1 \mod 2$. Donc si on choisit un message de $k = 4q+3$, $q \in \mathbb{N}$, la première sortie du circuit sera toujours la sortie du bas.
2. Si on a une mémoire de taille $m = 3 \mod 4$
   - Si on a $k = 1 \mod 4$ alors $m+k = 0 \mod 4$, donc $-\frac{m+k}{2}+1 = 1 \mod 2$. Donc si on choisit un message de $k = 4q+1$, $q \in \mathbb{N}$, la première sortie du

m	rende-ment	forme systé-matique	$\max(d_f)$	$\#G_{\mathbb{Z}}^{(\max(d_f))}$	$\#G_{D_\infty}^{(\max(d_f))}$	$\#G_{D_\infty}^{(\max(d_f))}$ choix du treillis de départ
3	1/3	oui	6	7	3	7
3	1/3	non	8	3	3	7
3	1/5	oui	12	4	4	12
3	1/5	non	13	10	10	70
5	1/3	oui	9	6	6	32
5	1/3	non	12	6	6	144
3	1/7	oui	17	15	15	105
3	1/7	non	18	56	56	656
5	1/5	oui	17	90	90	1636

TABLE 4.4 – Distance libre maximale et nombre de matrices de transfert atteignant ce maximum pour des codes de longueur de mémoire et de rendement donnés

circuit sera toujours la sortie du bas.

— Si on a $k = 3 \mod 4$ alors $m+k = 2 \mod 4$, donc $-\frac{m+k}{2}+1 = 0 \mod 2$. Donc si on choisit un message de $k = 4q+3$, $q \in \mathbb{N}$, la première sortie du circuit sera toujours la sortie du haut.

Si on fait ce choix, on voit que le nombre de matrices de transfert qui atteignent la distance libre maximale $\left(\#G_{D_\infty}^{(\max d_f)}\right)$ est plus grand que le nombre de matrices de transfert atteignant la distance libre maximale dans le cas des codes convolutifs classiques $\left(\#G_{\mathbb{Z}}^{(\max d_f)}\right)$.

Si on n'adapte pas la longueur du message, alors la distance libre est le minimum des deux distances libres associées à chaque treillis. Dans ce cas, on a le même nombre de matrices de transfert qui atteignent la distance libre maximale pour les codes sur D_∞ que pour les codes sur $\mathbb{Z}$.

4.7 Reconnaissance de code

Nous imaginons le scénario d'attaque suivant : un attaquant écoute le canal et veut trouver un message à partir d'un mot intercepté. Il ne connaît aucun des paramètres du code. Il doit donc faire de la reconnaissance de code.

Pour cela, il applique le critère du rang [Mar09]. Comme les n bits de sortie du codeur sont dépendants, il obtient une chute de rang pour les matrices de taille $M \times \alpha n$, avec $\alpha \geqslant 1$. Cependant, il ne peut pas en déduire k et les autres paramètres du code. S'il suppose qu'il a affaire à un code sur D_∞, il sait que les p-èmes blocs de n bits, avec $p = 0 \mod 2$ appartiennent à un code convolutif classique et que les autres blocs appartiennent au code « réciproque » de ce code classique. Il en déduit donc deux trains de bits. S'il dispose d'un algorithme permettant de retrouver la matrice de parité d'un code classique, et par suite la matrice génératrice, il l'applique à un

des trains de bits calculés et en déduit la matrice génératrice de l'autre code grâce au lien entre les deux codes.

La complexité de la reconnaissance n'est donc pas améliorée dans le cas de ces codes convolutifs sur le groupe diédral infini. Nous n'avons donc pas les propriétés cryptographiques recherchées à cause de la trop grande proximité entre le groupe D_∞ et le groupe $\mathbb{Z}$.

4.8 Généralisation de ces codes

Soit D_∞^n le groupe qui a pour présentation :

$$D_\infty^n = \langle x_0, x_1, \ldots, x_n \mid x_0^2 = x_1^2 = \ldots = x_n^2 = 1 \rangle$$

Nous constatons que $D_\infty = D_\infty^1$. À ce groupe, on associe son algèbre sur $\mathbb{F}_2$, $\mathbb{F}_2[D_\infty^n]$. En notation polynomiale on peut voir $\mathbb{F}_2[D_\infty^n]$ comme :

$$\mathbb{F}_2[D_\infty^n] = \mathbb{F}_2\{X_0, X_1, \ldots, X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$$

Définition 4.4. *Soit M un monôme de $\mathbb{F}_2[D_\infty^n]$. On appelle **degré** de M le nombre de variables X_i qui permettent d'écrire M (en comptant les répétitions).*

Exemple : le degré de $X_0X_1X_0$ est 3.

Théorème 4.6. *Soit P un polynôme de $\mathbb{F}_2[D_\infty^n]$. Alors P peut s'écrire sous la forme :*

$$P = A + X_0B$$

avec $A, B \in \mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$ comme sous-algèbre de $\mathbb{F}_2[D_\infty^n]$. Ainsi $\mathbb{F}_2[D_\infty^n]$ est un $\mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$-module à droite libre de type fini de base $1, X_0$.

Démonstration. Soit M un monôme de $\mathbb{F}_2[D_\infty^n]$.

- Soit le degré de M est pair et peut donc s'écrire à l'aide des monômes de degré 2 $X_0X_j, 1 \leqslant j \leqslant n$ ou de leurs inverses.
- Soit le degré de M est impair. Alors on multiplie M à gauche par X_0, et X_0M est de degré pair. Pour retrouver le monôme original, on remultiplie à gauche par X_0 et on a $X_0(X_0M)$ avec $X_0M \in \mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/-(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$.

□

Théorème 4.7. *L'anneau non-commutatif $\mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1,-X_1^2 - 1, \ldots, X_n^2 - 1)$ est isomorphe à l'algèbre de groupe $\mathbb{F}_2[*^n\mathbb{Z}]$ ($*^n\mathbb{Z}$ est le produit libre de n copies de $\mathbb{Z}$).*

Démonstration. Soient $S_1, \ldots, S_n$, n variables. On identifie :

- $S_j \leftrightarrow X_0X_j,\ \forall j \in \{1, \ldots, n\}$
- $S_j^{-1} \leftrightarrow X_jX_0,\ \forall j \in \{1, \ldots, n\}$
- $S_k^{-1}S_j \leftrightarrow X_kX_j,\ \forall j, k \in \{1, \ldots, n\}$

On a ainsi identifié toutes les paires X_jX_k de $\mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1, X_1^2-1, \ldots, X_n^2-1)$. Les monômes de plus haut degré s'obtiennent par combinaison libre des S_j, S_j^{-1}, $j \in \{1, \ldots, n\}$. Le groupe construit par toutes les combinaisons libres de n variables et de leurs inverses correspond à la définition du produit libre de n copies de $\mathbb{Z}$. □

Théorème 4.8. *Tout élément non nul de l'anneau* $\mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2-1, X_1^2-1, \ldots, X_n^2-1)$ *est un élément régulier à droite de l'anneau* $\mathbb{F}_2\{X_0, X_1, \ldots, X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$.

Démonstration. Soit $P \in \mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$ non nul. Nous voulons montrer que P est un élément régulier à droite de $\mathbb{F}_2\{X_0, X_1, \ldots, X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$. Soit $C \in \mathbb{F}_2\{X_0, X_1, \ldots, X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$ tel que $CP = 0$. On sait qu'il existe $A, B \in \mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$ tels que $C = A + X_0B$. On a :

$$0 + X_00 = CP = (A + X_0B)P = AP + X_0(BP).$$

Comme AP et BP sont dans $\mathbb{F}_2[*^n\mathbb{Z}]$, on en déduit que $AP = 0$ et $BP = 0$.
Or d'après [Mal48, Neu49], la conjecture de Kaplansky :
« Soit K un corps et G un groupe. L'algèbre de groupe $K[G]$ est sans diviseurs de zéro si et seulement si G est un groupe sans torsion » a été démontrée pour les groupes libres [1]. Or dans notre cas, on a $G = *^n\mathbb{Z}$ et $K = \mathbb{F}_2$. Donc $\mathbb{F}_2[*^n\mathbb{Z}]$ est sans diviseur de zéro. Comme $P \neq 0$, on a nécessairement $A = 0$ et $B = 0$ c'est-à-dire $C = 0$. □

On pourrait donc utiliser les polynômes de $\mathbb{F}_2\{X_0X_1, X_0X_2, \ldots, X_0X_n\}/(X_0^2 - 1, X_1^2 - 1, \ldots, X_n^2 - 1)$ comme fonction de transfert de ces codes sur D_∞^n.

Cependant, ces codes ne vont pas donner un rendement acceptable. En effet, le nombre de monômes dans $\mathbb{F}_2[D_\infty^n]$ de degré q, $q \geqslant 1$, est égal à $n^{q-1}(n + 1)$. Donc, si $n > 1$, on a n fois plus de monômes de degré $q + 1$ que de degré q. Le nombre de monômes dans $\mathbb{F}_2[D_\infty^n]$ de degré $\leqslant q$ est donc $1 + (n + 1)\frac{1-n^q}{1-n}$, avec $q \geqslant 1$ et $n > 1$.

Donc si on a une fonction de transfert somme de plusieurs monômes de degré 1, la longueur de la redondance ajoutée au message sera de $(n + 1)n^q$ avec q le degré du plus grand monôme du message.

Exemple 4.4. *Exemple avec* $n = 2$. *Soit* τ *une fonction de transfert égale à* X_i+X_j, $i \neq j$. *Nous allons voir dans le tableau 4.5 que la longueur de la redondance ajoutée est très grande par rapport à la longueur du message* k.

En effet, soit $u(X_0, X_1, X_2) = 1 + X_0 + X_1 + X_2$ *et* $\tau(X_0, X_1, X_2) = X_0 + X_1$. *On a :*

$$\begin{aligned} u\tau(X_0, X_1, X_2) &= (1 + X_0 + X_1 + X_2)(X_0 + X_1) \\ &= X_0 + X_1 + X_0X_1 + X_1X_0 + X_2X_0 + X_2X_1 \end{aligned}$$

1. Malcev et Neumann montrent que $K[G]$ est contenu dans un corps non-commutatif, et donc par déduction, que $K[G]$ n'a pas de diviseurs de zéro non triviaux.

k	q	longueur redondance	longueur mot de code
4	1	6	10
10	2	12	22
22	3	24	46
46	4	48	94

TABLE 4.5 – Exemple de la longueur de la redondance pour $n = 2$

Si on utilise l'ordre des éléments suivants :

$$1, X_0, X_1, X_2, X_0X_1, X_0X_2, X_1X_0, X_1X_2, X_2X_0, X_2X_1, \ldots$$

on obtient $u = 1111$ *et* $u\tau = 0110101011$*, donc un message de longueur* $k = 4$ *donne un mot de code de longueur* 10 *et donc une redondance de* $10 - 4 = 6$*.*

Nous voyons bien qu'en pratique, pour $n > 1$, ces codes n'ont pas un rendement acceptable, car ils introduisent une redondance trop grande par rapport à la taille du message.

4.9 Conclusion

Ces codes sur le groupe diédral infini sont bien définis mathématiquement et électroniquement et ont la même complexité d'encodage et de décodage que les codes convolutifs classiques. De plus, ils permettent d'obtenir plus de codes optimaux en terme de distance libre. Cette plus grande quantité de codes optimaux peut être utilisée pour concevoir un émetteur/récepteur dédié propriétaire. En effet, plus le choix de codes optimaux avec les mêmes paramètres et les mêmes performances pour encoder l'information est grand, plus il est possible d'implémenter un encodeur/décodeur dédié dont les mots de code ne seront pas décodables par certains de ses concurrents. Néanmoins, ils ne rendent pas plus difficile la reconnaissance de code et n'ont pas de propriétés cryptographiques (dans le sens où l'opération d'encodage ne dépend pas d'un élément qu'on pourrait garder secret). Enfin, ils ne permettent pas d'améliorer les performances de codage car le groupe est trop proche du groupe des entiers relatifs.

Chapitre 5

Codes sur le produit semi-direct $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$

Nous avons vu, dans le chapitre précédent, que les codes convolutifs définis sur un groupe de cardinal infini ne sont ni plus résistants à la reconnaissance de code, ni des systèmes cryptographiques. Pour avoir des propriétés cryptographiques, on pourrait faire varier l'encodage dans le temps, ce qui n'est pas le cas des codes convolutifs classiques ou des codes sur D_∞. En effet, pour les codes convolutifs classiques, le message u est défini sur l'algèbre du groupe $\mathbb{Z}$ à partir de ses bits $u_i \in \mathbb{F}_2$, $i \in \{0, \dots, k-1\}$ par l'expression suivante :

$$u = \sum_i u_i X^i$$

Et dans le cas de codes convolutifs définis sur le groupe diédral infini, le message u est défini sur l'algèbre du groupe D_∞ à partir de ses bits $u_i \in \mathbb{F}_2$, $i \in \left\{\frac{1-k}{2}, \dots, 0, \dots, \frac{k-1}{2}\right\}$ par l'expression suivante :

$$u = \sum_{i \text{ pair}} u_i (XY)^{\frac{i}{2}} + \sum_{i \text{ impair}} u_i (XY)^{\frac{i-1}{2}} X$$

Dans les chapitres suivants, nous allons remplacer ces groupes infinis par des groupes finis et ainsi faire du codage en bloc. Pour cela, nous allons tout d'abord découper le message u en blocs de tailles variables. Pour des raisons de notations, nous allons considérer dans les chapitres suivants, sauf mention contraire, que le message u est un bloc de message à encoder. Pour encoder ce message de taille k sur le groupe fini G, on l'envoie, comme expliqué au chapitre 3, sur un sous-ensemble $E = \{e_0, \dots, e_{k-1}\}$ du groupe G. L'idée est, qu'à chaque nouvel encodage, on utilise un sous-ensemble E différent du précédent, qui sera calculé de manière déterministe mais paraîtra aléatoire sans la connaissance de « la clé » de l'encodage.

Le premier objectif des chapitres suivants est donc de calculer des intervalles d'encodage chaotiques et de grandes longueurs pour avoir un bon rendement $\frac{k}{Card(G)} = \frac{k}{n}$.

Dans ce chapitre, nous allons travailler avec le groupe fini $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ de cardinal NM impair, avec N, M impairs et $PGCD(M, \varphi(N)) \neq 1$ (avec φ l'indicatrice d'Euler).

Voyons pourquoi nous avons cette dernière condition. Notons $A = \mathbb{Z}/N\mathbb{Z}$ et $H = \mathbb{Z}/M\mathbb{Z}$. Le groupe des automorphismes de A est isomorphe au groupe des inversibles de $\mathbb{Z}/N\mathbb{Z}$:

$$Aut(A) = Aut(\mathbb{Z}/N\mathbb{Z}) \simeq (\mathbb{Z}/N\mathbb{Z})^*$$

et $Card((\mathbb{Z}/N\mathbb{Z})^*) = \varphi(N)$. L'application ϕ, qui permet de définir la loi de groupe du produit semi-direct, est un morphisme de $H = \mathbb{Z}/M\mathbb{Z}$ dans $Aut(A) = (\mathbb{Z}/N\mathbb{Z})^*$ donc, pour avoir un morphisme non-trivial, et avoir un produit semi-direct non direct, nous avons besoin de la condition :

$$PGCD(Card(H), Card(Aut(A)) = PGCD(M, \varphi(N)) \neq 1$$

Comme le $PGCD(Card(\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}), car(\mathbb{F}_2)) = 1$ (avec $car(K)$ la caractéristique de l'anneau K), on va pouvoir utiliser la théorie de Fourier. On va ainsi simplifier la convolution de deux fonctions de $\mathbb{F}_2^{\mathbb{Z}/N\mathbb{Z}\rtimes\mathbb{Z}/M\mathbb{Z}}$ par la transformée de Fourier inverse du produit de leur transformée de Fourier. Nous allons donc tout d'abord introduire la théorie de Fourier pour les groupes finis.

5.1 Théorie de Fourier

Les définitions de cette partie sont tirées de [RS10, Ser71].

Notation :
Dans cette partie, V désigne un espace vectoriel, $GL(V)$ le groupe des isomorphismes linéaires de V dans lui-même.

Définition 5.1. *Une **représentation linéaire** (ρ, V) du groupe G est la donnée d'un espace vectoriel V appelé espace de la représentation et d'un morphisme de groupe*

$$\rho : G \to GL(V)$$
$$g \mapsto \rho_g$$

La dimension de la représentation (ρ, V) est la dimension de V. On la note d_ρ.

Exemple 5.1. *La représentation triviale de G est celle où $V = \mathbb{C}$ et tout $g \in G$ agit comme l'identité de $\mathbb{C}$. On la note 1_G.*

Définition 5.2. *Soit (ρ, V) une représentation du groupe G. Un sous-espace W de V est dit invariant par ρ si pour tout $g \in G$, $\rho_g(W) \subset W$. On peut alors parler de la restriction de ρ à W que l'on note $(\rho_{|W}, W)$. Une telle représentation restreinte à un sous-espace invariant s'appelle une **sous-représentation** de G.*

Définition 5.3. *Soit K un anneau commutatif, soit G un groupe fini et soit $K[G]$ l'algèbre de groupe de G sur K. Comme nous l'avons vu dans le chapitre 2, tout élément f de $K[G]$ s'écrit de façon unique sous la forme*

$$f = \sum_{s \in G} \alpha_s s, \quad avec\ \alpha_s \in K$$

On peut alors étendre, et ce, de façon unique, la représentation (ρ, V) de G en un morphisme de $K[G]$ vers $End(V)$ en posant :

$$\rho\left(\sum_{s\in G}\alpha_s s\right) = \sum_{s\in G}\alpha_s\rho_s$$

Ceci fait de V un $K[G]$-module. On dit aussi que V est un G-module. Réciproquement, la donnée d'un $K[G]$-module fournit une représentation de G.

Définition 5.4. *Une représentation (ρ, V) du groupe G est dite **irréductible** si elle est non nulle et n'admet aucun sous-espace autre que $\{0\}$ et V invariant par ρ. On dit aussi que V est simple en tant que $K[G]$-module.*

Définition 5.5. *Soient (ρ, V) et (ρ', V') deux représentations du groupe G. Un **opérateur d'entrelacement** $T : V \to V'$ est une application linéaire de V dans V' vérifiant :*

$$T(\rho_g(v)) = \rho'_g(T(v)),\ g \in G, v \in V$$

Définition 5.6. *Soient (ρ, V) et (ρ', V') deux représentations du groupe G. Elles sont **équivalentes** (ou isomorphes) s'il existe un opérateur d'entrelacement inversible $T : V \to V'$. Si T est un tel opérateur d'entrelacement inversible, T^{-1} est bien sûr aussi un opérateur d'entrelacement et*

$$\rho'_g = T \circ \rho_g \circ T^{-1},\ g \in G$$

Définition 5.7. *Un module V sur un anneau est dit **semi-simple** ou complètement réductible s'il est somme directe de sous-modules simples.*
Une représentation ρ est dite semi-simple ou complètement réductible si elle est somme directe de sous-représentations irréductibles.

Théorème 5.1 (Théorème de Maschke)**.** *Si G est un groupe fini et si la caractéristique de K ne divise pas $Card(G)$, alors tout $K[G]$-module est semi-simple. On peut donc décomposer toutes les représentations de G en somme directe de représentations irréductibles.*

Définition 5.8. *Notons $\hat{G}$ l'ensemble des classes d'équivalences de représentations irréductibles du groupe G. Choisissons un représentant (ρ_i, V_i) de chaque classe de $\hat{G}$. Soit $f \in K[G]$, l'algèbre de groupe de G. La **transformée de Fourier** de f est un élément du produit $\prod_{\rho_i\in\hat{G}} End(V_i)$ définie par :*

$$\hat{f}(\rho_i) = \sum_{x\in G} f(x)\rho_i(x)$$

Pour définir la transformée de Fourier inverse, il faut rappeler la définition de la trace d'une application linéaire :

Définition 5.9. *La **trace** d'une matrice carrée A est définie comme la somme des coefficients diagonaux et est notée $Tr(A)$. Si u est un endomorphisme d'un espace vectoriel de dimension finie sur un anneau K, on peut définir la trace de l'opérateur u comme la trace de sa matrice dans n'importe quelle base.*

Définition 5.10. *Soit $F \in \prod_{\rho_i \in \hat{G}} End(V_i)$. La **transformée de Fourier inverse** est définie sur $K[G]$ (avec la caractéristique de K qui ne divise pas $Card(G)$) par :*

$$\check{F}(x) = \frac{1}{Card(G)} \sum_{\rho_i \in \hat{G}} d_\rho Tr(F(\rho_i)\rho_i(x)^{-1})$$

Théorème 5.2. *Soient $f_1, f_2 \in K[G]$. On a :*

$$\widehat{f_1 * f_2} = \widehat{f_1}\widehat{f_2}$$

Démonstration. Soient $f_1, f_2 \in K[G]$.

$$\begin{aligned}
\widehat{f_1 * f_2}(\rho_i) &= \sum_{x \in G} (f_1 * f_2)(x)\rho_i(x) \\
&= \sum_{x \in G} \sum_{t \in G} f_1(t) f_2(t^{-1}x)\rho_i(x) \\
&= \sum_{x \in G} \sum_{t \in G} f_1(t) f_2(x)\rho_i(tx) \\
&= \sum_{x \in G} \sum_{t \in G} f_1(t) f_2(x)\rho_i(t)\rho_i(x) \\
&= \left(\sum_{t \in G} f_1(t)\rho_i(t)\right)\left(\sum_{x \in G} f_2(x)\rho_i(x)\right) \\
&= \widehat{f_1}(\rho_i)\widehat{f_2}(\rho_i)
\end{aligned}$$

□

Théorème 5.3. *Soit $f \in K[G]$. Alors $\check{\hat{f}} = f$.*

Démonstration. Démonstration réalisée dans [RS10] Lemme II.5.3 □

Définition 5.11. *Soit (ρ, V) une représentation linéaire de G, et soit ρ_H sa restriction à H. Soit (θ, W) une sous-représentation de ρ_H. Soit $s \in G$, l'espace vectoriel $\rho_s(W)$ ne dépend que de la classe à gauche sH de s ; en effet, si l'on remplace s par st avec $t \in H$, on a $\rho_{st}(W) = \rho_s\rho_t(W) = \rho_s(W)$ puisque $\rho_t(W) = W$. Si σ est une classe à gauche modulo H, on peut donc définir un sous-espace W_σ de V comme étant les $\rho_s(w)$ pour tout $s \in \sigma$. La somme $\sum_{\sigma \in G/H} W_\sigma$ est une sous-représentation de V.*

*On dit que la représentation ρ de G dans V est **induite** par la représentation θ de H dans W si V est égal à la somme de W_σ ($\sigma \in G/H$) et si cette somme est une somme directe (c'est-à-dire $V = \oplus_{\sigma \in G/H} W_\sigma$).*

Définition 5.12. *Soient V_1 et V_2 deux espaces vectoriels. On appelle produit tensoriel de V_1 et V_2 un espace vectoriel W muni d'une application $(x_1, x_2) \mapsto x_1.x_2$ de $V_1 \times V_2$ dans W, cette application vérifiant les conditions suivantes :*

1. *$x_1.x_2$ dépend linéairement de chacune des variables x_1 et x_2.*
2. *Si (e_{i_1}) est une base de V_1 et (e_{i_2}) est une base de V_2, la famille des produits $e_{i_1}.e_{i_2}$ est une base de W.*

On montre facilement qu'un tel espace W existe, et est unique (à isomorphisme près) et on le note $V_1 \otimes V_2$. La condition 2 montre que :

$$dim(V_1 \otimes V_2) = dim(V_1).dim(V_2)$$

Soient maintenant (ρ^1, V_1) et (ρ^2, V_2) deux représentations linéaires d'un groupe G. Si $s \in G$, on définit un élément ρ_s de $GL(V_1 \otimes V_2)$ par la condition :

$$\rho_s(x_1.x_2) = \rho_s^1(x_1).\rho_s^2(x_2), \ \forall x_1 \in V_1, x_2 \in V_2$$

On écrit :

$$\rho_s = \rho_s^1 \otimes \rho_s^2$$

Les ρ_s définissent une représentation linéaire de G dans $V_1 \otimes V_2$ qui est appelée le ***produit tensoriel*** *des représentations données.*

Soient A et H deux sous-groupes du groupe G, le sous-groupe A étant distingué. Faisons les hypothèses suivantes :

1. A est commutatif
2. $G = A \rtimes H$

Nous allons voir que les représentations irréductibles de G peuvent être construites à partir de celles de certains sous-groupes de H.

Comme A est commutatif, ses représentations irréductibles sont de dimension 1 et forment un groupe $X = Hom(A, \mathbb{C}^*)$. Le groupe G opère sur X par :

$$(s\chi)(a) = \chi(s^{-1}as), \text{ avec } s \in G, \chi \in X, a \in A$$

Soit $(\chi_i)_{i \in X/H}$ un système de représentants des orbites de H dans X. Pour tout $i \in X/H$, soit H_i le sous-groupe de H formé des éléments h tels que $h\chi_i = \chi_i$, et soit $G_i = A.H_i$ le sous-groupe correspondant de G. Prolongeons la fonction χ_i à G_i en posant :

$$\chi_i(ah) = \chi_i(a), \text{ avec } a \in A, h \in H_i$$

En utilisant le fait que $h\chi_i = \chi_i$ pour tout $h \in H_i$, on voit que χ_i est une représentation de dimension 1 de G_i. Soit d'autre part ρ une représentation irréductible de H_i ; en composant ρ et la projection canonique $G_i \to H_i$ on en déduit une représentation irréductible $\tilde{\rho}$ de G_i. Enfin, par produit tensoriel de χ_i et de $\tilde{\rho}$, on obtient une représentation irréductible $\chi_i \otimes \tilde{\rho}$ de G_i, soit $\theta_{i,\rho}$ la représentation induite correspondant de G.

Proposition 5.1. *1. $\theta_{i,\rho}$ est irréductible.*

2. Si $\theta_{i,\rho}$ et $\theta_{i',\rho'}$ sont isomorphes, on a $i = i'$ et ρ est isomorphe à ρ'.

3. Toute représentation irréductible de G est isomorphe à l'une des $\theta_{i,\rho}$.

Démonstration. Preuve réalisée dans [Ser71] paragraphe 8.2 □

Dans le cas du groupe $G = \mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$, comme la caractéristique de $\mathbb{F}_2$ est 2 ce qui ne divise pas le cardinal de G qui est impair, on va pouvoir appliquer le théorème 5.1 de Maschke et on va ainsi étudier les représentations irréductibles de G pour pouvoir définir la transformée de Fourier de toute fonction $f \in \mathbb{F}_2[G]$.

5.2 Codes sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

Le groupe de plus petit cardinal de la forme $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ est le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ que nous allons étudier en détails dans la suite. Nous allons comparer certaines de ses propriétés avec les groupes $\mathbb{Z}/13\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ (de cardinal 39 et avec $PGCD(12,3) = 3$) et $\mathbb{Z}/11\mathbb{Z} \rtimes \mathbb{Z}/5\mathbb{Z}$ (de cardinal 55 et avec $PGCD(10,5) = 5$).

Nous allons donc étudier en détails les codes construits sur $G = \mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$. C'est un groupe de cardinal $7 \times 3 = 21$ avec $\varphi(7) = 6$ et $PGCD(3,6) = 3 \neq 1$ donc le produit semi-direct n'est pas un produit direct. G est l'ensemble $\mathbb{Z}/7\mathbb{Z} \times \mathbb{Z}/3\mathbb{Z}$ muni de la loi

$$\forall (a,h),(a',h') \in G, \quad (a,h) + (a',h') = (a + \phi_h(a'), h + h')$$

avec $\phi_h \in Aut(\mathbb{Z}/7\mathbb{Z})$. Nous avons choisi $\phi_h(a) = 2^h a$. La loi de G peut aussi s'écrire sous la forme :

$$\forall a \in \mathbb{Z}/7\mathbb{Z},\ h \in \mathbb{Z}/3\mathbb{Z}, \quad (0,h) + (a,0) - (0,h) = (\phi_h(a), 0)$$

Comme $\mathbb{Z}/7\mathbb{Z}$ est commutatif, ses représentations en caractéristique 2 sont de dimension 1 et forment un groupe $X = Hom(\mathbb{Z}/7\mathbb{Z}, \mathbb{F}_8)$. Pour faciliter les notations, notons :

$$A = \{(a,0) \in G,\ a \in \mathbb{Z}/7\mathbb{Z}\}$$

et

$$H = \{(0,h) \in G,\ h \in \mathbb{Z}/3\mathbb{Z}\}$$

Les représentations de A sont :

	$(0,0)$	$(1,0)$	$(2,0)$	$(3,0)$	$(4,0)$	$(5,0)$	$(6,0)$
ψ_0	1	1	1	1	1	1	1
ψ_1	1	α	α^2	α^3	α^4	α^5	α^6
ψ_2	1	α^2	α^4	α^6	α	α^3	α^5
ψ_3	1	α^3	α^6	α^2	α^5	α	α^4
ψ_4	1	α^4	α	α^5	α^2	α^6	α^3
ψ_5	1	α^5	α^3	α	α^6	α^4	α^2
ψ_6	1	α^6	α^5	α^4	α^3	α^2	α

avec $\alpha^7 = 1$ donc $\alpha \in \mathbb{F}_8$.

Le groupe G opère sur X par :

$((s_1,s_2)+\psi)((a,0)) = \psi((s_1,s_2)+(a,0)-(s_1,s_2))$ avec $(s_1,s_2) \in G,\ \psi \in X,\ a \in \mathbb{Z}/7\mathbb{Z}$

Nous allons calculer les orbites de X sous l'action à gauche de H :

$$H + \psi_0 = \{(0,0) + \psi_0, (0,1) + \psi_0, (0,2) + \psi_0\} = \{\psi_0\}$$

$$H + \psi_1 = \{(0,0) + \psi_1, (0,1) + \psi_1, (0,2) + \psi_1\} = \{\psi_1, \psi_2, \psi_4\}$$

$$H + \psi_3 = \{(0,0) + \psi_3, (0,1) + \psi_3, (0,2) + \psi_3\} = \{\psi_3, \psi_5, \psi_6\}$$

$\{\psi_0, \psi_1, \psi_3\}$ est donc un système de représentants des orbites de X sous l'action de H. Considérons les représentations qui vont venir de ψ_0. Or pour tout $h \in H, h+\psi_0 = \psi_0$. Prolongeons la fonction ψ_0 à G en posant :

$$\psi_0((a,h)) = \psi_0((a,0)) \ \ \forall a \in A, \ h \in H$$

Ainsi, ψ_0 est une représentation de dimension 1 de G. Intéressons nous maintenant aux représentations irréductibles de H notées dans le tableau suivant :

	$(0,0)$	$(0,1)$	$(0,2)$
ρ_0	1	1	1
ρ_1	1	β	β^2
ρ_2	1	β^2	β

avec $\beta^3 = 1$ donc $\beta \in \mathbb{F}_4$. Soit ρ_i une représentation irréductible de H. En composant ρ_i et la projection canonique $G \to H$, on en déduit une représentation irréductible $\tilde{\rho}_i$ de G. Puis, par produit tensoriel de $\tilde{\rho}_i$ et de ψ_0, on obtient une représentation irréductible de G.

Les fonctions ψ_0 et $\psi_0 \otimes \rho_0$ nous donnent la représentation triviale de G, puis les représentations irréductibles $\psi_0 \otimes \rho_1$ et $\psi_0 \otimes \rho_2$.

Nous nous intéressons ensuite aux représentations de G qui vont être induites de ψ_1. On forme le sous-groupe H_1 de H composé des éléments $h \in H$ tel que $h + \psi_1 = \psi_1$. On a $H_1 = \{(0,0)\}$ et donc $G_1 = A \rtimes H_1 = A$. La représentation ψ_1 de $G_1 = A$ est de dimension 1. Une représentation irréductible de H_1 est donnée par $\rho((0,0)) = 1$. On compose ρ et la projection canonique $G_1 \to H_1$ pour obtenir $\tilde{\rho}$ et on effectue le produit tensoriel $\psi_1 \otimes \tilde{\rho}$ qui nous donne une représentation irréductible de G_1. Calculons la représentation induite correspondante sur G, notée $\theta^{(1)}$. Comme $Card(G_1) = 7 = \frac{Card(G)}{3}$, $\theta^{(1)}$ sera de dimension 3.

Une base du module associé à cette représentation est $e_1 = (1,0,0)$, $e_2 = (0,1,0)$ et $e_3 = (0,0,1)$. Notons que e_2 est l'image de e_1 par $\theta^{(1)}_{(0,1)}$ et e_3 l'image de e_1 par $\theta^{(1)}_{(0,2)}$ car $(0,1)$ et $(0,2)$ ne sont pas dans G_1. On a :

$$\begin{array}{lll} \theta^{(1)}_{(0,0)}(e_1) = e_1 & \theta^{(1)}_{(0,1)}(e_1) = e_2 & \theta^{(1)}_{(0,2)}(e_1) = e_3 \\ \theta^{(1)}_{(0,0)}(e_2) = e_2 & \theta^{(1)}_{(0,1)}(e_2) = e_3 & \theta^{(1)}_{(0,2)}(e_2) = e_1 \\ \theta^{(1)}_{(0,0)}(e_3) = e_3 & \theta^{(1)}_{(0,1)}(e_3) = e_1 & \theta^{(1)}_{(0,2)}(e_3) = e_2 \end{array}$$

Et :

$$\forall j \in A\backslash\{0\}, \ \forall i \in \{1,2,3\}, \ \theta^{(1)}_{(j,0)}(e_i) = \psi_1(j,0)(e_i)$$

Enfin :

$$\forall j \in A\backslash\{0\}, \ \forall k \in H\backslash\{0\}, \ \forall i \in \{1,2,3\}, \ \theta^{(1)}_{(j,k)}(e_i) = \theta^{(1)}_{(j,0)+(0,k)}(e_i) = \theta^{(1)}_{(0,k)}\left(\theta^{(1)}_{(j,0)}(e_i)\right)$$

Le même raisonnement est effectué pour ψ_3. Ainsi, nous obtenons le tableau A.1 des représentations irréductibles de G, placé en annexe A.

Ces représentations font intervenir α et β tels que $\alpha^7 = 1$ et $\beta^3 = 1$, donc nous sommes tentés d'exprimer α et β en fonction d'un élément ω tel que $\omega^{21} = 1$. Or $21 = \frac{63}{3} = \frac{64-1}{3} = \frac{2^6-1}{3}$. Les éléments α et β pourraient s'exprimer en fonction d'un élément $\gamma \in \mathbb{F}_{64}$ tel que $\gamma^{63} = 1$. On en déduit le nouveau tableau A.2 des représentations irréductibles, en annexe A.

Nous pouvons ainsi définir la transformée de Fourier $\hat{f}$ de $f : G \to \mathbb{F}_{64}$ par :

$$\forall R_i \in \hat{G}, \hat{f}(R_i) = \sum_{x \in G} f(x) R_i(x)$$

Et la transformée de Fourier inverse sera définie par :

$$\forall x \in G,\ f(x) = \frac{1}{21} \sum_{R_i \in \hat{G}} d_{R_i} Tr\left(\hat{f}(R_i)(R_i(x))^{-1}\right)$$

D'après la proposition 5.2, on a :

$$\widehat{f * \tau} = \hat{f}.\hat{\tau}$$

Théorème 5.4. *Soient* $f, \tau \in \mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}}$.
Si $\forall i,\ 1 \leqslant i \leqslant 5,\ \widehat{\tau}(R_i)$ *est inversible alors l'application* $f \mapsto f * \tau$ *est injective.*

Démonstration. Soit i, tel que $1 \leqslant i \leqslant 5$ et $\tau \in \mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}}$

$$\begin{aligned}
\forall i,\ \widehat{\tau}(R_i) \text{ est inversible} &\Rightarrow \forall f \in \mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}} \backslash \{0\},\ \widehat{f}\widehat{\tau} \neq 0 \\
&\Rightarrow \forall f \in \mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}} \backslash \{0\},\ \widehat{f * \tau} \neq 0 \\
&\Rightarrow \forall f \in \mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}} \backslash \{0\},\ f * \tau \neq 0 \\
&\Rightarrow f \mapsto f * \tau \text{ est injective}
\end{aligned}$$

□

Donc, pour avoir un codage injectif, on peut calculer la transformée de Fourier de toute fonction $\tau \in \mathbb{F}_2[G]$ et vérifier que $\forall i, \widehat{\tau}(R_i)$ est inversible.

5.3 Parcours des éléments

Nous voulons encoder le message u sur le groupe $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$, de sorte que le support du message sur le groupe soit différent à chaque encodage. L'idée est de parcourir les éléments de $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$, de façon déterministe et chaotique.

Soit $\mathbb{Z}^2$ l'ensemble des points à coordonnées dans $\mathbb{Z}$ du plan euclidien et soit :

$$\begin{aligned}
&f : \mathbb{Z}^2 \to \mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z} \\
&(x, y) \mapsto (x \mod N, y \mod M)
\end{aligned}$$

Soit $H = \{(Na, Mb)\ a, b \in \mathbb{Z}\}$. On a $f(x, y) = f(x', y')$ si et seulement si $(x, y) - (x', y') \in H$. En effet :

$$\begin{aligned} f(x,y) = f(x',y') &\Leftrightarrow (x \bmod N, y \bmod M) = (x' \bmod N, y' \bmod M) \\ &\Leftrightarrow (x \bmod N, y \bmod M) - (x' \bmod N, y' \bmod M) \\ &\quad = (0 \bmod N, 0 \bmod M) \\ &\Leftrightarrow (x \bmod N, y \bmod M) + (-x' - 2^{-y'} \bmod N, -y' \bmod M) \\ &\quad = (0 \bmod N, 0 \bmod M) \\ &\Leftrightarrow (x - x'2^{-y'}2^{y} \bmod N, y - y' \bmod M) \\ &\Leftrightarrow (x - x'2^{y-y'} \bmod N, y - y' \bmod M) \end{aligned}$$

Donc $y' = y + Mb, \forall b \in \mathbb{Z}$, d'où $2^{y-y'} = 2^{Mb} = 1 \mod N$ et donc $x' = x + Na, \forall a \in \mathbb{Z}$. Donc $f(x, y) = f(x', y') \Leftrightarrow (x, y) - (x', y') \in H$.

Donc l'ensemble $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ s'identifie avec l'ensemble quotient $\mathbb{Z}^2/H$. Nous allons calculer un domaine fondamental de H sur le plan euclidien. L'action de H sur $\mathbb{Z}^2$ envoie le point (x, y) vers le point $(Na + x, Mb + y)$. On a :

$$H.(0,0) = \{(0,0), (N,0), (0,M), (N,M), \ldots\}$$

On étend l'action de H sur $\mathbb{Z}^2$ à $\mathbb{R}^2$ et ainsi, un domaine fondamental de H est le rectangle $([0, N] \times [0, M]) \setminus ([(0, M), (N, M)] \cup [(N, 0), (N, M)])$ du plan euclidien, avec les segments $[(0, M), (N, M)]$ et $[(N, 0), (N, M)]$ qui sont identifiés respectivement aux segments $[(0, 0), (N, 0)]$ et $[(0, 0), (0, M)]$. Ce domaine est représenté sur la figure 5.1. En recollant les côtés identifiés deux à deux, on obtient un tore.

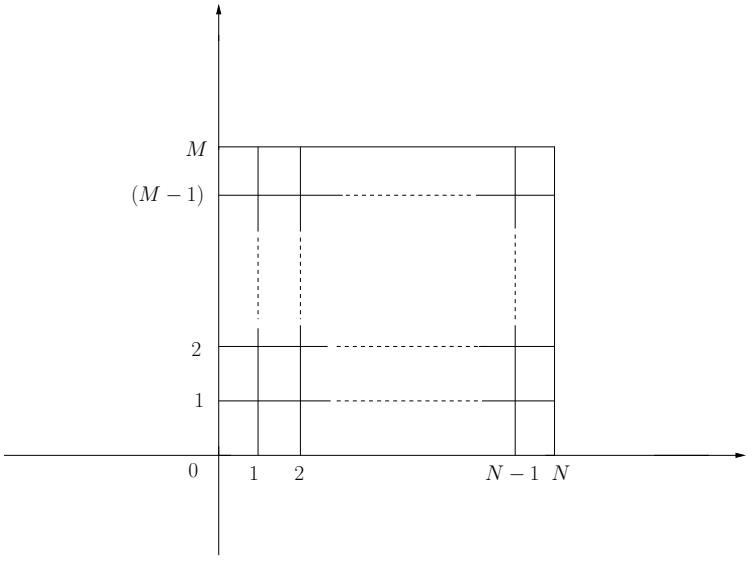


FIGURE 5.1 – Domaine fondamental de H

Nous allons ainsi représenter dans le plan chaque élément du groupe $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ en faisant correspondre chaque élément (i, j) du groupe avec le petit rectangle $[(i, j), (i + 1, j), (i + 1, j + 1), (i, j + 1)]$ du tore $\mathbb{R}^2/H$. Cette représentation va nous

permettre de parcourir les éléments dans un ordre différent de l'ordre naturel et de façon chaotique.

Nous allons tout d'abord parcourir les cases du domaine par une droite de pente irrationnelle, puis par un système dynamique discret. Les parcours obtenus doivent être chaotiques et éviter que les répétitions d'éléments surviennent trop tôt afin d'avoir des intervalles d'encodage suffisamment longs.

5.3.1 Parcours avec une droite de pente irrationnelle

Les droites délimitant les cases dans le domaine ont pour équation $x = b$ ou $y = b$ avec $b \in \mathbb{Z}$. Si on parcourt le domaine avec des droites $y = ax$ de pentes irrationnelles, on est certain de ne jamais tomber sur un sommet d'une case. Nous allons donc parcourir le domaine avec de telles droites, noter les cases qu'elles traversent dans des intervalles d'encodage sans répétitions et étudier la longueur des intervalles d'encodage associés à chacune de ces pentes.

Trois groupes $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ avec $(N, M) = (7, 3)$, $(N, M) = (13, 3)$ et $(N, M) = (11, 5)$ ont été pris comme références. L'algorithme 5.1 a été implémenté en Sage. Les résultats de cet algorithme pour les trois groupes cités précédemment sont respectivement dans les graphiques 5.2, 5.3 et 5.4. Sur ces graphiques, on a, en abscisse les différentes pentes, et, en ordonnée, la moyenne des longueurs des intervalles d'encodage.

Algorithme 5.1 Calcul de la moyenne des longueurs des intervalles d'encodage en fonction de la pente

Entrée: N, M
Sortie: Le tableau des moyennes $TabMoy$
  Calculer le domaine associé au groupe $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$
  Initialiser $TabMoy$ à un tableau de $M \times 100$ cases
  Pour $i = 0$ à M par pas de 0.01 **Faire**
    Tirer aléatoirement t_i un nombre irrationnel entre i et $i + 0.01$
    Initialiser un tableau vide $FacesTraversees_i$
    Tant que $longueur(FacesTraversees_i) < N \times M \times 100$ **Faire**
      Parcourir le domaine avec la droite de pente t_i
      Noter les faces traversées dans le tableau $FacesTraversees_i$
    Fin Tant que
    Calculer les longueurs des intervalles d'encodage associées au tableau $FacesTraversees_i$ et mettre le résultat dans le tableau $LongEncod_i$
    $TabMoy(i) = moyenne(LongEncod_i)$
  Fin Pour
  Retourner $TabMoy$

Sur chaque graphique, nous observons deux pics :
— pour $(N, M) = (7, 3)$ en 2 et environ 0.17
— pour $(N, M) = (13, 3)$ en 2 et environ 0.09
— pour $(N, M) = (11, 5)$ en 4 et environ 0.1

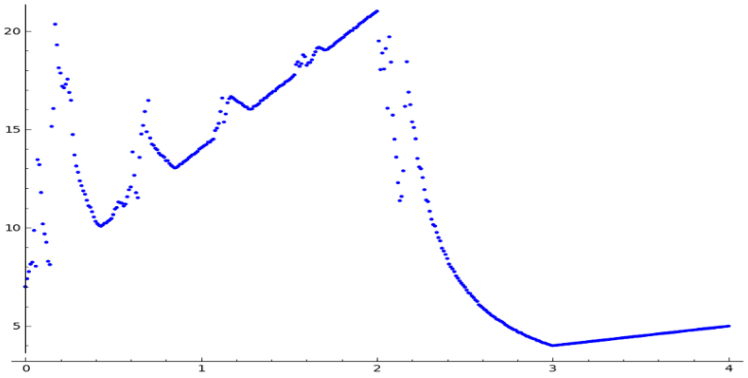


FIGURE 5.2 – Longueur moyenne d'encodage en fonction de la pente de la droite pour $N = 7$ et $M = 3$

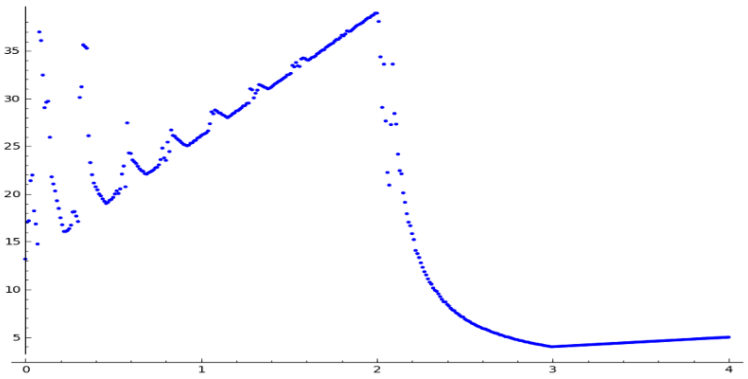


FIGURE 5.3 – Longueur moyenne d'encodage en fonction de la pente de la droite pour $N = 13$ et $M = 3$

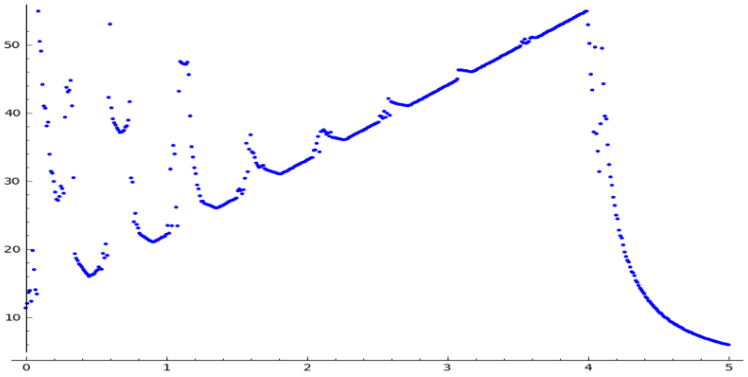


FIGURE 5.4 – Longueur moyenne d'encodage en fonction de la pente de la droite pour $N = 11$ et $M = 5$

Sur la figure 5.5, nous pouvons voir à quoi ressemblent les parcours par des droites de telles pentes pour $(N, M) = (7, 3)$. Nous constatons que la droite de pente proche de 2 passe près du point $(1, M - 1)$ du plan. Ce qui veut dire que le 2 correspond à $M - 1$. L'autre pic correspond à une pente proche de $\frac{1}{N-1}$.

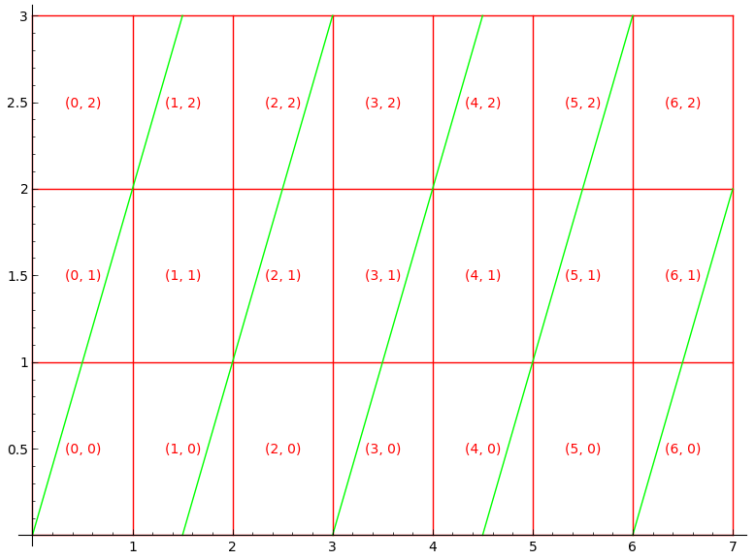


FIGURE 5.5 – Parcours de la droite de pente $M - 1 = 2,0000001$ avec $N = 7$ et $M = 3$

Soit $N, M \in \mathbb{Z}$. Nous allons nous placer dans $\mathbb{Z}^2$. Nous allons appeler H l'action de passer d'une case à celle immédiatement au-dessus et D l'action de passer d'une case à celle immédiatement à droite (schéma 5.6).

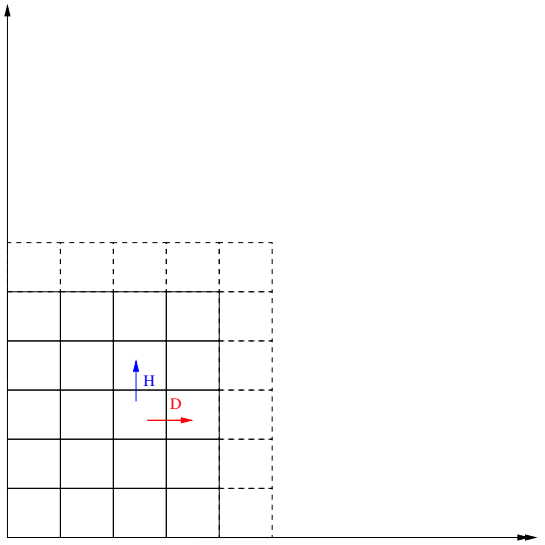


FIGURE 5.6 – Actions H et D

Une droite qui parcourt les cases ne peut le faire qu'en passant d'une case à celle du dessus ou à celle de droite. Si on regarde plus précisément les parcours de cases

des droites de pente irrationnelle très proche de $(M-1)$ tout en étant supérieures, c'est-à-dire

$$y = ax,\ a = M - 1 + \epsilon, \epsilon > 0$$

dans $\mathbb{Z}^2$, on peut constater que se succèdent $(M-1)$ H et 1 D de façon périodique. On obtient MN cases au bout de N périodes. Donc les cases parcourues sont de la forme :

$$(i, i(M-1)+j) \text{ avec } 0 \leqslant i \leqslant N-1 \text{ et } 0 \leqslant j \leqslant M-1$$

Si on se place dans le rectangle $\mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/M\mathbb{Z}$ on obtient les cases :

$$(i \mod N, i(M-1)+j \mod M) \text{ avec } 0 \leqslant i \leqslant N-1 \text{ et } 0 \leqslant j \leqslant M-1$$

Nous constatons qu'on parcourt toutes les cases du rectangle une et une seule fois comme on peut le voir sur la figure 5.5 avec $N = 7$ et $M = 3$.

Par symétrie, il se produit le même phénomène pour les pentes irrationnelles proches (tout en étant inférieures) de $\frac{1}{N-1}$.

Si on veut un code de rendement $\frac{k}{n}$ acceptable (avec k la longueur des intervalles d'encodage), on ne peut pas choisir une pente au hasard, il faut la choisir de telle sorte qu'elle produise des intervalles de longueur suffisante. Pour $(N, M) = (7, 3)$, si on veut un code de rendement supérieur à $\frac{14}{21} = \frac{2}{3}$, on peut choisir la pente entre 1.2 et 2 par exemple. On aura peut-être des intervalles de petites longueurs, mais en moyenne on aura un code de rendement supérieur à $\frac{2}{3}$.

Nous allons maintenant étudier l'aspect chaotique des intervalles calculés. Nous allons utiliser la définition 3.3 du chapitre 3. Nous avons testé la propriété de sensibilité aux conditions initiales et pour cela, choisi deux pentes a, b au hasard, proches l'une de l'autre d'un certain écart et calculé pour $l = 2100$, $\sum_{i=0}^{l} \frac{\phi_i(a,b)}{l}$. Les résultats sont présentés dans le tableau 5.1.

écarts e	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
$\frac{1}{2100}\sum_{i=0}^{2099} \phi_i(a, a+e)$	0,052	0,060	0,552	0,970	0,996	0,998	1

TABLE 5.1 – Calcul de $\dfrac{1}{2100}\sum_{i=0}^{2099} \phi_i(a, a+e)$ en fonction des écarts e entre les pentes des droites

On peut voir que plus les écarts e diminuent, plus $\dfrac{1}{2100}\sum_{i=0}^{2099} \phi_i(a, a+e)$ augmente et donc plus les parcours sont les mêmes, les parcours ne sont donc pas chaotiques.

De plus, si on regarde les éléments obtenus, on obtient souvent des cycles, c'est-à-dire qu'à partir d'un certain moment, la même suite d'éléments revient, le parcours n'est donc définitivement pas chaotique.

5.3.2 Parcours basé sur un système dynamique discret

Les intervalles d'encodage n'étant pas chaotiques dans le cas du parcours par une droite de pente irrationnelle, on va s'intéresser maintenant à un parcours que

l'on sait chaotique (Théorème 4.8 de [Dev89]), qui est le système dynamique discret sur le tore appelé « chat d'Arnold », du nom de celui qui l'a décrit en 1967 avec une photo de chat.

Comme ce système est habituellement décrit dans le carré unité, nous allons changer l'échelle du domaine fondamental, en divisant les abscisses par N et les ordonnées par M, afin de se ramener au carré $[0, 1, 1+i, i]$ sur $\mathbb{C}$.

Soit $A = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}$. A permet de définir l'automorphisme φ du tore $\mathbb{T}$ suivant :

$$\begin{gathered} \varphi : \mathbb{T} \to \mathbb{T} \\ \begin{pmatrix} x \\ y \end{pmatrix} \mapsto A \begin{pmatrix} x \\ y \end{pmatrix} \end{gathered}$$

Les itérations $\{\varphi^n\}_{n\in\mathbb{Z}}$ définissent un système dynamique discret.

Nous allons étudier sur quelles cases du domaine se situent les points (x, y) calculés à chaque itération par l'application φ à partir d'un point de départ choisi aléatoirement sur le domaine. Ce système est entièrement déterministe, mais se comporte comme un système aléatoire, de telle sorte qu'une case a une probabilité $\frac{1}{MN}$ d'être traversée. Ainsi la probabilité qu'un intervalle d'encodage soit de longueur k est la probabilité p_1 que $k-1$ cases soient différentes

$$p_1 = \frac{NM!}{(NM-k)!} \frac{1}{(NM)^k}$$

multiplié par la probabilité p_2 que la case $k+1$ soit égale à une des k cases précédentes

$$p_2 = \frac{k}{NM}$$

Ainsi $P(X = k) = p_1 p_2 = \dfrac{(NM)!}{(NM-k)!} \dfrac{1}{(NM)^k} \dfrac{k}{NM}$.

Le graphique 5.7 montre en bleu la longueur des intervalles calculés pour $N = 7$ et $M = 3$ et en rouge, la distribution des probabilités. On peut voir que les courbes coïncident et que la longueur des intervalles est très petite pour avoir un code de rendement acceptable.

Vérifions maintenant l'aspect chaotique des parcours, c'est-à-dire testons tout d'abord la propriété de sensibilité aux conditions initiales. Pour cela, on prend deux points proches l'un de l'autre ($x_2 = x_1 + \epsilon$, $y_2 = y_1 + \epsilon$), et on calcule pour $l = 2100$, $\sum_{i=0}^{l} \frac{\phi_i((x_1,y_1),(x_2,y_2))}{l}$. Le résultat est dans le tableau 5.2.

écarts e	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
$\dfrac{1}{2100} \displaystyle\sum_{i=0}^{2099} \phi_i((x_1, y_1), (x_2, y_2))$	0,050	0,045	0,044	0,048	0,045	0,047	0,051

TABLE 5.2 – Calcul de $\sum_{i=0}^{2099} \frac{\phi_i((x_1,y_1),(x_2,y_2))}{2100}$ en fonction des écarts entre (x_1, y_1) et (x_2, y_2)

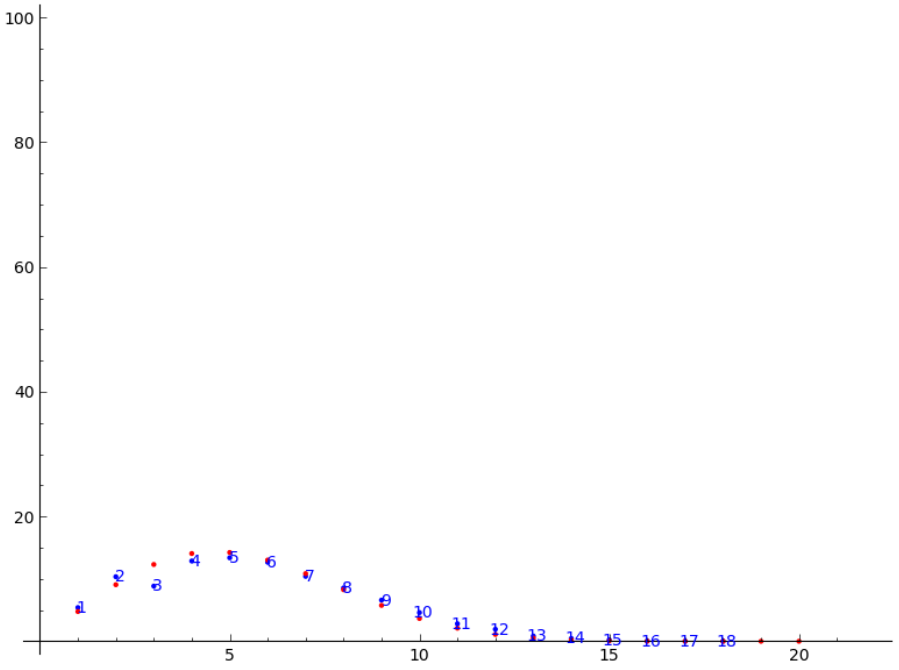


FIGURE 5.7 – Comparaison entre les longueurs des intervalles d'encodage associées aux cases traversées par le système dynamique discret sur $\mathbb{Z}/7\mathbb{Z} \times \mathbb{Z}/3\mathbb{Z}$ (en bleu) et la probabilité qu'un intervalle d'encodage soit de longueur donnée en abscisse (en rouge)

On peut constater dans ce tableau que, quelque soit l'écart entre les points, il y a toujours environ $\frac{1}{21} \simeq 0,0476$ éléments qui sont égaux. La propriété de sensibilité aux conditions initiales est donc bien vérifiée.

Nous allons maintenant tester la propriété d'uniforme répartition des éléments. Pour cela, nous calculons le nombre d'occurrence o_g de chaque élément g dans une suite de 2100 éléments et calculons $\frac{o_g}{2100}$ que nous comparons avec $\frac{1}{21} \simeq 0,0476$. Le résultat se trouve dans le tableau 5.3.

Les valeurs de $\frac{o_g}{2100}$ sont proches de $\frac{1}{21}$ donc on peut constater que les éléments sont uniformément répartis dans le parcours. Donc on peut en déduire que les intervalles d'encodage issus de ce parcours sont bien chaotiques.

5.3.3 Intervalles d'encodage de longueur choisie

Nous allons maintenant aborder le problème d'une autre façon. Si on parcourt le domaine avec le système dynamique discret précédent, on obtient des parcours chaotiques, cependant, on a rapidement des répétitions d'éléments ce qui conduit à une longueur d'intervalles d'encodage trop faible. Nous allons regarder ce qui se passe si on décide d'ignorer ces répétitions. Nous allons donc parcourir le domaine et, à chaque nouvelle face traversée, nous allons tester si elle appartient déjà à l'intervalle que nous voulons remplir. Si oui, l'algorithme continue en « laissant tomber » cette face, si non, cette face est ajoutée à l'intervalle que nous voulons remplir. On peut donc remplir des intervalles de la longueur k qu'on souhaite. Toute cette procédure

indice de g	o_g	$\frac{o_g}{2100}$
0	94	0.0447
1	85	0.0404
2	101	0.048
3	101	0.048
4	97	0.0461
5	121	0.0576
6	101	0.048
7	110	0.0523
8	90	0.0428
9	103	0.049
10	102	0.0485
11	110	0.0523
12	95	0.0452
13	95	0.0452
14	74	0.0352
15	98	0.0466
16	100	0.0476
17	110	0.0523
18	104	0.0495
19	104	0.0495
20	105	0.05

TABLE 5.3 – Calcul des occurrences des éléments g dans une suite de 2100 éléments parcourus sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

est réalisée par l'algorithme 5.2. Nous faisons ainsi du codage de rendement $\frac{k}{n}$ avec k choisi.

Utiliser cet algorithme va être coûteux en parcours des éléments, c'est-à-dire qu'on va devoir parcourir plus de cases du domaine avant d'avoir les k cases distinctes qui nous intéressent pour remplir un intervalle. Dans le tableau 5.4, nous avons calculé le nombre de cases à parcourir pour avoir 100 intervalles de longueur k sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$. Dans ces intervalles, il y a donc $100k$ éléments du groupe qui vont être utilisés pour l'encodage de 100 messages.

Pour $k = 20$, on doit donc parcourir environ 2.8 fois plus de cases pour avoir les 2000 éléments dans les intervalles d'encodage. Le coût est donc plus petit que 3.

On obtient donc des intervalles d'encodage de longueur k choisie et chaotiques.

5.3.4 Problèmes de précision

Tout d'abord, il est évident que, si l'ordinateur de l'expéditeur utilise une certaine précision pour les nombres irrationnels, il faut que l'ordinateur du destinataire utilise la même précision pour calculer les intervalles d'encodage E. C'est d'autant plus indispensable dans la partie 5.3.2, à cause du caractère chaotique du système.

Ensuite, dans la partie 5.3.1, nous parcourons les faces du domaine avec une droite de pente irrationnelle, pour éviter les sommets de faces. Tous les logiciels ne peuvent pas travailler en précision infinie afin de conserver toutes les décimales des

Algorithme 5.2 Calcul d'intervalles d'encodage de longueur k sur le groupe $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$

Entrée: N, M, $NbInterval$ le nombre d'intervalles, (x, y) le point de départ, k la longueur des intervalles d'encodage
Sortie: $TabInterval$ le tableau contenant tous les intervalles

```
Construire le domaine associé à Z/NZ × Z/MZ
Initialiser un vecteur DansInterval de taille N × M.
NumeroElt = 1
Tant que longueur(TabInterval) < NbInterval Faire
  ( x ) = A ( x )
  ( y )     ( y )
  Si x > 1 Alors
    x = x − floor(x)
  Fin Si
  Si y > 1 Alors
    y = y − floor(y)
  Fin Si
  ft = face(x, y) le numéro de la face dans laquelle se trouve le point (x, y).
  Si DansInterval(ft) == 0 Alors
    DansInterval(ft) = NumeroElt
    NumeroElt + +
    Si NumeroElt == k + 1 Alors
      Initialiser un vecteur Interval de taille k.
      Pour j = 1 à N × M Faire
        Si DansInterval(j) ≠ 0 Alors
          Interval(DansInterval(j)) = j
        Fin Si
      Fin Pour
      Ajouter Interval au tableau TabInterval
      NumeroElt = 1
      Réinitialiser DansInterval à un vecteur de taille N × M.
    Fin Si
  Fin Si
Fin Tant que
Retourner TabInterval
```

k	$100k$	nombre de cases parcourues
1	100	100
2	200	206
3	300	318
4	400	437
5	500	557
6	600	693
7	700	833
8	800	978
9	900	1156
10	1000	1316
11	1100	1507
12	1200	1721
13	1300	1987
14	1400	2198
15	1500	2520
16	1600	2879
17	1700	3301
18	1800	3805
19	1900	4561
20	2000	5535

TABLE 5.4 – Nombre moyen de cases à parcourir pour avoir 100 intervalles de longueur k

irrationnels. Souvent une troncature s'effectue sur le nombre irrationnel et il devient un nombre rationnel. Pour le parcours de la droite, on utilise l'équation $y = ax$ qu'on ramène à chaque fois par identification des cotés au carré $N \times M$. Si $a = \frac{p}{q}$ est un nombre rationnel sous forme réduite, on atteindra pour la première fois un sommet de face lorsque $x = q$ car ainsi $y = p$. Il suffira d'arrêter l'algorithme une fois qu'on parviendra à ce problème et recommencer avec une autre pente.

Enfin, dans la partie 5.3.2, nous parcourons le domaine avec un système dynamique discret. On utilise un point (x, y) irrationnel, pour éviter de tomber sur une abscisse $x = \frac{p}{N}$ ou une ordonnée $y = \frac{q}{M}$. Si le logiciel remplace l'irrationnel par un rationnel proche, on risque de tomber sur le problème qu'on cherche à éviter.

Tout d'abord, il y a le risque que le point (x, y) revienne au point de départ et donc que la suite des faces soient un cycle trop court pour avoir des intervalles chaotiques. En effet, si on prend $(x, y) = (1/8, 3/8)$ on obtient la suite des points $(1/8, 3/8), (1/2, 7/8), (3/8, 1/4), (5/8, 7/8), (1/2, 3/8), (7/8, 1/4), (1/8, 3/8), \ldots$ donc la période est de 6. D'après [DF92], si on divise $[0, 1]$ en intervalles de taille $\frac{1}{R}$ pour chaque dimension du carré unité, la période m_R du système sera le plus petit entier tel que

$$A^{m_R} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \mod R$$

Donc si on divise le carré unité en intervalles de longueur $\frac{1}{8}$, on aura :

$$A^6 = \begin{pmatrix} 89 & 144 \\ 144 & 233 \end{pmatrix} \mod 8 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \mod 8$$

Et la période sera bien de 6. Donc, pour avoir un système qui ne revient pas rapidement à son état de départ, il faut diviser le carré unité en un grand nombre d'intervalles. Soit $x = \frac{p_x}{q_x}$ et $y = \frac{p_y}{q_y}$ sous forme réduite. Le nombre d'intervalles par lequel sera divisé le carré unité sera le $PPCM(q_x, q_y)$. Ensuite, il faut s'assurer qu'il y ait suffisamment d'itérations du point (x, y) avant d'atteindre $x = \frac{p}{N}$ ou $y = \frac{q}{M}$. Si on choisit q_x et q_y premiers avec N et M, les seuls points problématiques que l'on puisse atteindre sont sur les droites $x = 0$ ou $y = 0$ modulo 1. Si on note $B = A^t,\ t \in \mathbb{N}^*$, on aura :

$$\begin{aligned} x_t &= B_{1,1}x + B_{1,2}y \in \mathbb{N} \\ y_t &= B_{2,1}x + B_{2,2}y \in \mathbb{N} \\ \Rightarrow x_t &= \frac{B_{1,1}p_x}{q_x} + \frac{B_{1,2}p_y}{q_y} \in \mathbb{N} \\ y_t &= \frac{B_{2,1}p_x}{q_x} + \frac{B_{2,2}p_y}{q_y} \in \mathbb{N} \end{aligned}$$

$$\begin{aligned} \Rightarrow &\forall p_x,\ q_x \text{ divise } B_{1,1} \text{ et } \forall p_y,\ q_y \text{ divise } B_{1,2} \\ &\forall p_x,\ q_x \text{ divise } B_{2,1} \text{ et } \forall p_y,\ q_y \text{ divise } B_{2,2} \end{aligned}$$

Donc on cherche le plus petit entier t tel que :

— soit q_x divise $B_{1,1}$ et q_y divise $B_{1,2}$

— soit q_x divise $B_{2,1}$ et q_y divise $B_{2,2}$

Ce t sera la première itération au cours de laquelle une itération de x ou y sera sur le bord du domaine. Si ce t est plus petit que le nombre de bits de notre message alors il faudra changer de dénominateurs q_x et q_y si on travaille avec des rationnels ou changer de points x et y si on travaille avec des irrationnels.

5.4 Codage

Nous allons étudier le codage sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$, dont nous avons étudié la théorie de Fourier au début du chapitre. Supposons que nous ayons calculé des intervalles d'encodage. Nous allons étudier les propriétés d'un code en bloc construit sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$. Soit U un message de longueur l. On divise u en différents blocs de tailles correspondant aux longueurs des intervalles d'encodage. Chaque bloc de message va être envoyé sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ et être convolué avec une fonction de transfert sur le groupe.

Nous allons tout d'abord voir quelles sont les fonctions de transfert qui rendent le codage injectif dans $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$. Pour cela, nous allons appliquer le théorème 5.4 à chaque fonction de $\mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}}$. Le nombre de fonctions de $\mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}}$ est de $2^{21} = 2\ 097\ 152$. Nous avons testé si chacune de ces fonctions rendaient le codage

injectif et 84 672 fonctions ont passé le test, ce qui fait un peu plus de 4% des fonctions de $\mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}}$.

Nous allons voir dans l'exemple suivant comment on effectue le codage d'un message $u \in \mathbb{F}_2^k$ sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$.

Exemple 5.2. *Tout d'abord, on rappelle la loi de groupe*[1] *de* $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ *:*

$$\forall (a,b),(c,d) \in \mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z},\ (a,b) \oplus (c,d) = (a + 2^b c \mod 7, b + d \mod 3)$$

A chaque élément $(i,j) \in \mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$*, on associe un numéro tel que* $(i,j) \mapsto Mi + j = 3i + j$*. Donc on a le tableau de correspondance entre éléments et numéros suivant :*

$(0,0)$	$(1,0)$	$(2,0)$	$(3,0)$	$(4,0)$	$(5,0)$	$(6,0)$
g_0	g_3	g_6	g_9	g_{12}	g_{15}	g_{18}
$(0,1)$	$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$	$(6,1)$
g_1	g_4	g_7	g_{10}	g_{13}	g_{16}	g_{19}
$(0,2)$	$(1,2)$	$(2,2)$	$(3,2)$	$(4,2)$	$(5,2)$	$(6,2)$
g_2	g_5	g_8	g_{11}	g_{14}	g_{17}	g_{20}

Soit $u = [1,0,1,0,0,0,1,1,0,1,1,1,0,1]$*. Sa longueur est* $k = 14$*. Soit* $(x,y) = \left(\frac{\sqrt{2}}{2}, \frac{16\sqrt{2}}{113}\right)$ *un point irrationnel du domaine de* $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$*. On calcule l'intervalle d'encodage de taille* $k = 14$ *(à l'aide du système dynamique discret de la partie 5.3.3) associé à* (x,y)*. Le point de départ* (x,y) *a la propriété suivante :*

$$\frac{4}{7} < x < \frac{5}{7} \text{ et } 0 < y < \frac{1}{3}$$

Par conséquence, le premier élément de E *est* $(4,0)$*, soit* g_{12}*. La première itération du système dynamique nous donne le second point* $(x,y) = \left(\frac{145\sqrt{2}}{226}, \frac{177\sqrt{2}}{226} - 1\right)$*. Nous avons que* $\frac{6}{7} < x < 1$ *et* $0 < y < \frac{1}{3}$*, donc le second élément de* E *est* $(6,0)$*, c'est-à-dire* g_{18}*. Après plusieurs itérations, nous obtenons :*

$$\begin{aligned} E &= [(4,0),(6,0),(0,0),(2,1),(0,2),(5,1),(1,2),(6,2),(4,1),(0,1),\\ &\quad (3,2),(2,0),(5,0),(5,2)]\\ &= [g_{12}, g_{18}, g_0, g_7, g_2, g_{16}, g_5, g_{20}, g_{13}, g_1, g_{11}, g_6, g_{15}, g_{17}] \end{aligned}$$

Le message sur le groupe sera donc :

$$\begin{aligned} u &= \sum_i u_i E(i)\\ &= (4,0) + (0,0) + (1,2) + (6,2) + (0,1) + (3,2) + (2,0) + (5,2) \end{aligned}$$

Soit τ *la fonction de transfert telle que :*

$$\begin{aligned} \tau &= [0,0,0,0,0,0,0,0,1,0,1,1,1,0,1,1,1,0,0,0,0]\\ &= \sum_{i=0}^{n-1} \tau_i g_i = g_8 + g_{10} + g_{11} + g_{12} + g_{14} + g_{15} + g_{16}\\ &= (2,2) + (3,1) + (3,2) + (4,0) + (4,2) + (5,0) + (5,1) \end{aligned}$$

1. On préfère la noter $\oplus$ afin de ne pas confondre avec le signe $+$ permettant de représenter les éléments de l'algèbre de groupe

Donc la convolution de u et τ sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ est :

$$\begin{aligned}
u * \tau &= ((4,0) + (0,0) + (1,2) + (6,2) + (0,1) + (3,2) + (2,0) + (5,2)) \\
&\quad \oplus ((2,2) + (3,1) + (3,2) + (4,0) + (4,2) + (5,0) + (5,1)) \\
&= (4,0) \oplus (2,2) + (4,0) \oplus (3,1) + (4,0) \oplus (3,2) + (4,0) \oplus (4,0) + (4,0) \oplus (4,2) \\
&\quad + (4,0) \oplus (5,0) + (4,0) \oplus (5,1) + ((0,0) + (1,2) + (6,2) + (0,1) + (3,2) \\
&\quad + (2,0) + (5,2)) \oplus ((2,2) + (3,1) + (3,2) + (4,0) + (4,2) + (5,0) + (5,1)) \\
&= (4 + 2^0 \times 2 \mod 7, 0 + 2 \mod 3) + (4 + 2^0 \times 3 \mod 7, 0 + 1 \mod 3) \\
&\quad + (4 + 2^0 \times 3 \mod 7, 0 + 2 \mod 3) + (4,0) \oplus (4,0) + (4,0) \oplus (4,2) \\
&\quad + (4,0) \oplus (5,0) + (4,0) \oplus (5,1) + ((0,0) + (1,2) + (6,2) + (0,1) + (3,2) \\
&\quad + (2,0) + (5,2)) \oplus ((2,2) + (3,1) + (3,2) + (4,0) + (4,2) + (5,0) + (5,1)) \\
&= (6,2) + (0,1) + (0,2) + (4,0) \oplus (4,0) + (4,0) \oplus (4,2) + (4,0) \oplus (5,0) \\
&\quad + (4,0) \oplus (5,1) + ((0,0) + (1,2) + (6,2) + (0,1) + (3,2) + (2,0) + (5,2)) \\
&\quad \oplus ((2,2) + (3,1) + (3,2) + (4,0) + (4,2) + (5,0) + (5,1)) \\
&= (0,2) + (1,0) + (1,1) + (3,0) + (3,2) + (4,2) + (5,1) + (5,2) + (6,0) + (6,2) \\
&= g_2 + g_3 + g_4 + g_9 + g_{11} + g_{14} + g_{16} + g_{17} + g_{18} + g_{20} \\
&= (0,0,1,1,1,0,0,0,0,1,0,1,0,0,1,0,1,1,1,0,1)
\end{aligned}$$

Donc le mot de code est $c = (0,0,1,1,1,0,0,0,0,1,0,1,0,0,1,0,1,1,1,0,1)$

5.5 Décodage

L'opération de convolution a la propriété d'être linéaire [2]. En effet, soit G un groupe et soient $u, v, \tau \in \mathbb{F}_2^G$, on a :

$$\begin{aligned}
\forall y \in G,\ (u+v) * \tau(y) &= \sum_{x \in G} (u+v)(x)\tau(x^{-1}y) \\
&= \sum_{x \in G} (u(x) + v(x))\tau(x^{-1}y) \\
&= \sum_{x \in G} u(x)\tau(x^{-1}y) + v(x)\tau(x^{-1}y) \\
&= \sum_{x \in G} u(x)\tau(x^{-1}y) + \sum_{x \in G} v(x)\tau(x^{-1}y) \\
&= u * \tau(y) + v * \tau(y)
\end{aligned}$$

Donc $(u+v) * \tau = u * \tau + v * \tau$.

L'opération de convolution sur un groupe fini permet donc de définir des codes linéaires en bloc. Le décodage des codes linéaires en bloc s'effectue par syndrome comme nous l'avons vu dans la partie 1.2.

Exemple 5.3. *Reprenons l'exemple précédent et supposons que nous avons envoyé le mot de code c et que le destinataire a reçu :*

$$r = (0,0,1,1,1,1,0,0,0,1,0,1,0,0,1,0,1,1,1,0,1)$$

2. Cela découle aussi du fait que la convolution coïncide avec le produit dans l'algèbre de groupe $\mathbb{F}_2[G]$

Il connaît l'intervalle d'encodage E et la fonction de transfert τ utilisés. Il peut donc construire la matrice de parité du code grâce à l'algorithme 5.3.

Algorithme 5.3 Calcul de la matrice de parité du code défini par un intervalle d'encodage E et une fonction de transfert τ sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

Entrée: E l'intervalle d'encodage et τ la fonction de transfert
Sortie: H la matrice de parité du code
$l = longueur(E)$
Initialiser G_T la matrice génératrice de taille $l \times 21$.
Pour $i = 0$ à $l - 1$ **Faire**
Initialiser f de taille 21
$f(E(i)) = 1$;
$G_T(i, :) = f * \tau$
Fin Pour
Réduire G_T sous forme systématique
En déduire H la matrice de parité

Dans l'exemple, on a :

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

Et si on multiplie H et r (mis en colonne), on obtient le vecteur colonne $(0, 1, 1, 0, 0, 0, 0)$*, qui correspond au syndrome d'une erreur sur le 6ème bit. Donc le destinataire corrige la valeur du 6ème bit et obtient* $c = (0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1)$ *qui est bien le mot de code envoyé.*

Ensuite, il faut revenir au message envoyé. Pour cela, le destinataire connaît τ donc il effectue l'opération suivante sur le groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$:

$$\begin{aligned} u &= c * \tau^{-1} \\ &= ((0,2) + (1,0) + (1,1) + (3,0) + (3,2) + (4,2) + (5,1) + (5,2) + (6,0) + (6,2)) \\ &\oplus ((0,2) + (1,1) + (1,2) + (2,0) + (2,2) + (3,0) + (3,1) \\ &+ (3,2) + (4,1) + (5,1) + (6,1)) \\ &= (0,0) + (0,1) + (1,2) + (2,0) + (3,2) + (4,0) + (5,2) + (6,2) \\ &= g_0 + g_1 + g_5 + g_6 + g_{11} + g_{12} + g_{17} + g_{20} \end{aligned}$$

Puis il identifie les bits u_i avec l'intervalle d'encodage E.

$$\begin{aligned} E &= [g_{12}, g_{18}, g_0, g_7, g_2, g_{16}, g_5, g_{20}, g_{13}, g_1, g_{11}, g_6, g_{15}, g_{17}] \\ u &= [1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1] \end{aligned}$$

5.6 Propriétés

Nous aimerions maintenant calculer la distance minimale des codes obtenus pour chacune des 84672 fonctions de transfert τ et chacun des 2^{21} intervalles d'encodage E possibles. Nous allons voir dans la partie suivante que leur nombre peut être réduit grâce à l'action du groupe G.

5.6.1 Actions de groupe et distance minimale

Soient f et τ deux fonctions de X^G avec X un ensemble et G un groupe, vu comme un G-ensemble à gauche ou un G-ensemble à droite.

Proposition 5.2. *Soient $f, \tau \in X^G$ et $g \in G$,*

$$g.(f * \tau) = f * (g.\tau)$$

Démonstration. Soit $x \in G$.

$$\begin{aligned}(g.(f*\tau))(x) &= (f*\tau)(x.g) \text{ d'après la définition 2.28}\\ &= \sum_{t\in G} f(t)\tau(t^{-1}xg)\\ &= \sum_{t\in G} f(t)(g.\tau)(t^{-1}x) \text{ d'après la définition 2.28}\\ &= (f*(g.\tau))(x)\end{aligned}$$

□

Proposition 5.3. *Soient $f, \tau \in X^G$ et $g \in G$,*

$$(f * \tau).g = (f.g) * \tau$$

Démonstration. Soit $x \in G$.

$$\begin{aligned}((f*\tau).g)(x) &= (f*\tau)(g.x) \text{ d'après la définition 2.27}\\ &= \sum_{t\in G} f(t)\tau(t^{-1}gx)\\ &= \sum_{y\in G} f(gy)\tau(y^{-1}x) \text{ avec } t = gy\\ &= \sum_{y\in G} (f.g)(y)\tau(y^{-1}x) \text{ d'après la définition 2.27}\\ &= ((f.g)*\tau)(x)\end{aligned}$$

□

Soit w la fonction qui calcule le poids d'un train de bits. Soit $f \in \mathbb{F}_2^G$, on note $w(f) = Card\,(\{x \in G \mid f(x) \neq 0\})$.

Proposition 5.4. *Soient $g \in G$ et $f \in \mathbb{F}_2^G$, on a :*

$$w(f.g) = w(g.f) = w(f)$$

Démonstration. Soit $g \in G$, on a :

$$\begin{aligned} w(f.g) &= Card\left(\{x \in G \mid (f.g)(x) \neq 0\}\right) \\ &= Card\left(\{x \in G \mid f(g.x) \neq 0\}\right) \end{aligned}$$

Or l'action de groupe induit seulement une permutation sur les éléments de G, donc :

$$w(f.g) = w(f)$$

De même pour l'action à gauche :

$$w(g.f) = w(f)$$

□

Ceci nous permet d'en déduire les théorèmes suivants.

Théorème 5.5. *Soient $f, \tau \in \mathbb{F}_2^G$ et $g \in G$,*

$$w(f * \tau) = w((f.g) * \tau)$$

Démonstration. D'après les propositions 5.4 et 5.2, on a :

$$w(f * \tau) = w(g.(f * \tau)) = w(f * (g.\tau))$$

□

Théorème 5.6. *Soient $f, \tau \in \mathbb{F}_2^G$ et $g \in G$,*

$$w(f * \tau) = w((f.g) * \tau)$$

Démonstration. D'après les propositions 5.4 et 5.3, on a :

$$w(f * \tau) = w((f * \tau).g) = w((f.g) * \tau)$$

□

Ainsi, pour calculer la distance minimale de la convolution de deux fonctions, on peut se limiter au calcul de la distance minimale de la convolution des f appartenant à des classes à droite différentes, ce qui signifie que si f est définie sur un sous-ensemble E, alors on se limite aux classes d'équivalences à gauche de ces sous-ensembles (car $X^E.g = X^{g^{-1}.E}$) et des τ appartenant à des classes à gauche différentes.

5.6.2 Calcul de la distance minimale et comparaison avec celle des codes linéaires en bloc

Avec les propriétés précédentes, on en déduit que si $f \in \mathbb{F}_2^E$ alors $f.g \in \mathbb{F}_2^E.g = \mathbb{F}_2^{g^{-1}.E}$ et $\forall x \in G,\ (g.\tau)(x) = \tau(x.g)$. On peut donc étudier seulement la distance minimale de la convolution d'un message défini sur les orbites par l'action à gauche de G sur les ensembles E avec les orbites de l'action à gauche de G sur les fonctions de transfert τ (ce qui revient à étudier les orbites de l'action à droite de G sur le support de τ). Cela revient à calculer la distance des codes définis avec $\frac{84\ 672}{21} = 4\ 032$ fonctions τ et 99 950 ensembles E, soit $4\ 032 \times 99\ 950 = 402\ 998\ 400$ codes à tester. Nous avons calculé la distance minimale maximale pour chaque longueur k, que nous comparons avec la distance minimale des meilleurs codes linéaires en bloc que l'on trouve dans [Gra07], ainsi que le nombre d'ensemble E qui atteignent telle distance. Les résultats sont dans les tableaux 5.5 et 5.6.

k	d_{min} codes en bloc	d_{min} codes sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$
2	14	13
3	12	11
4	10	10
5	10	9
6	8	8
7	8	7
8	8	7
9	8	6
10	7	5
11	6	5
12	5	4
13	4	4
14	4	4
15	4	3
16	3	3
17	2	2
18	2	2
19	2	2
20	2	1

TABLE 5.5 – Comparaison des distances minimales des codes linéaires en bloc connus avec celles des codes convolutifs sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

On peut voir, tout d'abord, que la distance minimale maximale des codes sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ est plus faible pour certains k que la distance des codes linéaires en bloc classiques et, est identique pour les longueurs k égales à $4, 6, 13, 14, 16, 17, 18, 19$.

Ensuite, on peut voir que tous les ensembles E ne permettent pas d'atteindre la distance minimale maximale. Nous avons mis quelques exemples de couples (τ, E) qui permettent d'atteindre la meilleure distance en annexe B.

Comme ces ensembles E vont être choisis de manière chaotique, on ne va pas pouvoir choisir de « bons » ensembles. Seul le choix de la fonction de transfert sera

k	combien de E atteignent telle distance
2	10 $E \to d = 13$
3	68 $E \to d = 11$
4	217 $E \to d = 10$ et 68 $E \to d = 9$
5	863 $E \to d = 9$ et 106 $E \to d = 8$
6	2596 $E \to d = 8$ et 2 $E \to d = 7$
7	5535 $E \to d = 7$ et 3 $E \to d = 6$
8	106 $E \to d = 7$ et 9584 $E \to d = 6$
9	12261 $E \to d = 6$ et 1759 $E \to d = 5$
10	16796 $E \to d = 5$
11	6204 $E \to d = 5$ et 10592 $E \to d = 4$
12	14020 $E \to d = 4$
13	9097 $E \to d = 4$ et 593 $E \to d = 3$
14	27 $E \to d = 4$ et 5511 $E \to d = 3$
15	2598 $E \to d = 3$
16	710 $E \to d = 3$ et 259 $E \to d = 2$
17	285 $E \to d = 2$
18	68 $E \to d = 2$
19	10 $E \to d = 2$
20	1 $E \to d = 1$

TABLE 5.6 – Nombre d'ensembles E atteignant telle distance sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

important pour avoir une bonne distance minimale. Pour cela, nous avons calculé, pour chaque fonction de transfert, la moyenne de la distance pour tous les ensembles E de longueur k. Nous avons obtenu les fonctions de transfert optimales en moyenne pour chaque longueur k qui se trouvent en annexe B. Le tableau 5.7 nous présente quelle distance moyenne atteint chacune de ces fonctions optimales pour chaque longueur k des intervalles d'encodage.

On peut constater que la distance moyenne optimale est plus faible que la distance minimale des meilleurs codes linéaires en bloc. Cependant, nous avons des propriétés cryptographiques que nous allons présenter dans la suite.

5.6.3 Reconnaissance de code et aspects cryptographiques

Imaginons qu'une tierce personne écoute la communication et intercepte le train de bits. Chaque mot de code est le résultat de la convolution d'un message et d'une fonction de transfert sur $G = \mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$, chaque message étant envoyé sur un sous-ensemble E de G différent. Chaque mot de code possède donc des bits dépendants, mais les relations de dépendance sont différentes pour chaque mot reçu. Donc si nous utilisons le critère du rang, nous constatons que toutes les matrices sont de rang plein, et nous ne pourrons pas en déduire les paramètres du code avec cette méthode.

Dans la partie 5.3.3, nous avons vu que, pour avoir une suite d'intervalles d'encodage E, il suffisait de choisir un point (x, y) au hasard sur le carré $[0, 1, 1 + i, i]$.

k	distance moyenne optimale
2	10,4
3	8,9
4	7,7895
5	7,0268
6	6,0517
7	5,3374
8	4,8088
9	4,3097
10	3.9243
11	3,6456
12	3,1897
13	2,8278
14	2,5845
15	2,1840
16	2
17	2
18	1,9365
19	1,44
20	1

TABLE 5.7 – Distance moyenne atteinte par les fonctions de transfert τ optimales pour tous les ensembles E de longueur k

Nous avons également vu que, si on prenait un point (x', y') différent et aussi proche que l'on veut du point (x, y), alors on obtenait une suite d'intervalles d'encodage E totalement différente, grâce au caractère chaotique du système dynamique mis en jeu. Ces intervalles d'encodage sont utilisés pour encoder le message sur le groupe et également pour le décoder et retrouver le message sous forme binaire envoyé. Il faut donc que l'expéditeur et le destinataire possède ces intervalles d'encodage. Pour cela, il leur suffit d'avoir le même point (x, y) et ils pourront calculer, en mettant leur logiciel à la même précision, ces intervalles d'encodage. Il paraît clair que, si on n'a pas les intervalles d'encodage, on ne peut pas décoder le message. En effet, on pourrait calculer $u = c * \tau^{-1}$ en supposant qu'on connaisse τ, donc on connaîtrait le message sur le groupe. Cependant, comme $u = \sum_i u_i E(i)$, on ne pourrait pas en déduire la valeur des u_i binaires qui composent le message. On pourrait donc en déduire un cryptosystème symétrique qui aurait pour clé secrète le point (x, y). Toute personne ne connaissant pas (x, y) serait incapable de décoder le mot reçu et même de corriger les erreurs, ne pouvant pas calculer la matrice de parité H. Le protocole serait ainsi :

Alice et Bob se mettent d'accord, sans la divulguer, sur une clé, c'est-à-dire un point (x, y) dans le carré $[0, 1, 1+i, i]$. Ils se mettent d'accord également sur les autres paramètres du code, N, M, k et la fonction de transfert τ qui doit être inversible.

Alice calcule autant d'intervalles d'encodage E que nécessaire pour envoyer tous les blocs de k bits qui composent son message à l'aide du point (x, y) et du système dynamique discret défini en 5.3.2. Elle calcule tous les $u = \sum_i u_i E(i)$ messages sur le groupe choisi. Elle encode ses messages par convolution sur le groupe avec la fonction

τ. Elle en déduit les bits des mots de code c à envoyer. Puis elle envoie tous ses mots de code.

Bob reçoit les mots r, mots de code qu'Alice a envoyé, qui ont subi des perturbations sur le canal. Il calcule, comme Alice, autant d'intervalles d'encodage E qu'il a reçu de mots r. Il en déduit la matrice de parité de chacun des codes. Il utilise le décodage par syndrome pour corriger les erreurs et retrouve les mots de code c envoyés par Alice. Il calcule $c * \tau^{-1}$ pour chacun des mots de code et retrouve le message u sur le groupe. Enfin, il utilise les intervalles d'encodage pour retrouver les bits u_i du message d'Alice.

5.7 Exemple complet

Nous allons voir un exemple complet d'encodage et de décodage.

Soit $U = (0,1,1,0,0,1,0,1,0,1,0,0,1,1,0,0,0,1,0,0,0,1,1,0,0,1,0,1,0,1,0,0,1,1,0,0,0,1,0)$. Soit $k = 13$. On sépare U en 3 blocs de 13 éléments :

$$u_1 = (0,1,1,0,0,1,0,1,0,1,0,0,1)$$

$$u_2 = (1,0,0,0,1,0,0,0,1,1,0,0,1)$$

$$u_3 = (0,1,0,1,0,0,1,1,0,0,0,1,0)$$

Soit $(x,y) = (\frac{4211\sqrt{2}}{2305} - 2, \frac{1912\sqrt{2}}{1533} - 1)$. Comme dans l'exemple 5.2, on itère le système dynamique et on regarde dans quelle case tombe le point (x,y) à chaque itération. On obtient la suite des éléments suivante :

$g_{14}, g_6, g_{10}, g_1, g_{12}, g_{15}, g_{18}, g_{20}, \cancel{g_{20}}, g_{17}, g_9, \cancel{g_{15}}, \cancel{g_{18}}, \cancel{g_{18}}, g_0, \cancel{g_0}, \cancel{g_0}, g_4, \cancel{g_{12}}, \cancel{g_{17}}, g_{13}$

$g_1, g_{16}, g_4, \cancel{g_{16}}, g_8, g_3, \cancel{g_3}, g_{14}, g_{10}, g_{19}, \cancel{g_4}, \cancel{g_{19}}, \cancel{g_8}, g_6, \cancel{g_{14}}, \cancel{g_{10}}, \cancel{g_{19}}, g_7, \cancel{g_{19}}, g_{12}, g_{15}, g_{18}$

$g_{18}, g_0, g_3, g_7, g_1, g_{12}, \cancel{g_{18}}, g_4, g_{11}, g_6, g_{14}, g_9, g_{15}, \cancel{g_{18}}, \cancel{g_0}, \cancel{g_3}, \cancel{g_{11}}, g_5$

Les éléments barrés sont ceux qui sont déjà apparus précédemment dans un bloc de 13 éléments. On obtient donc 3 intervalles d'encodage différents. On rappelle qu'on a la bijection $(i,j) \to Mi + j = 3i + j$ entre les éléments du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ et les numéros des éléments du groupe dans les intervalles d'encodage. Ce qui nous donne les intervalles d'encodage suivants sur le groupe :

$E_1 = ((4,2),(2,0),(3,1),(0,1),(4,0),(5,0),(6,0),(6,2),(5,2),(3,0),(0,0),$
$(1,1),(4,1))$

$E_2 = ((0,1),(5,1),(1,1),(2,2),(1,0),(4,2),(3,1),(6,1),(2,0),(2,1),(4,0),$
$(5,0),(6,0))$

$E_3 = ((6,0),(0,0),(1,0),(2,1),(0,1),(4,0),(1,1),(3,2),(2,0),(4,2),(3,0),$
$(5,0),(1,2))$

Donc, sur l'algèbre du groupe, les blocs de message sont égaux à :

$$u_1 = \sum_i u_{1_i} E_{1_i} = (2,0) + (3,1) + (5,0) + (6,2) + (3,0) + (4,1)$$
$$u_2 = \sum_i u_{2_i} E_{2_i} = (0,1) + (1,0) + (2,0) + (2,1) + (6,0)$$
$$u_3 = \sum_i u_{3_i} E_{3_i} = (0,0) + (2,1) + (1,1) + (3,2) + (5,0)$$

Nous choisissons τ parmi ceux qui donnent la meilleure distance moyenne :

$$\begin{aligned}\tau &= (0,0,0,0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,1) \\ &= \sum_i \tau_i g_i = (2,1) + (2,2) + (3,0) + (3,1) + (4,0) + (4,2) + (5,0) + (5,1) + (6,0) \\ &+ (6,1) + (6,2)\end{aligned}$$

Nous effectuons la convolution de chaque u_i avec τ et nous obtenons 3 mots de code c_i :

$$\begin{aligned}c_1 &= (1,0) + (1,1) + (2,2) + (3,0) + (3,2) + (4,0) + (4,1) + (4,2) + (5,0) + (5,1) \\ c_2 &= (0,0) + (0,1) + (1,2) + (2,0) + (2,1) + (3,1) + (3,2) + (4,0) + (4,1) + (6,0) \\ &+ (6,2) \\ c_3 &= (0,0) + (0,1) + (0,2) + (2,0) + (2,2) + (3,0) + (3,1) + (4,0) + (4,1) + (4,2) \\ &+ (5,1) + (5,2) + (6,2)\end{aligned}$$

Nous transformons les mots de code c_i en binaire avec le fait que $c_i = \sum_j c_{i_j} g_j$ (avec $c_{i_j} \in \mathbb{F}_2$) et nous avons :

$$c_1 = (0,0,0,1,1,0,0,0,1,1,0,1,1,1,1,1,1,0,0,0,0)$$
$$c_2 = (1,1,0,0,0,1,1,1,0,0,1,1,1,1,0,0,0,0,1,0,1)$$
$$c_3 = (1,1,1,0,0,0,1,0,1,1,1,0,1,1,1,0,1,1,0,1,0)$$

Imaginons que les mots de code ont été envoyés à travers un canal binaire symétrique et que le destinataire reçoit les mots :

$$r_1 = (0,0,1,1,1,0,0,0,1,1,0,1,1,1,1,1,1,0,0,0,0)$$
$$r_2 = (1,1,0,0,0,1,1,0,0,0,1,1,1,1,0,0,0,0,1,0,1)$$
$$r_3 = (1,1,1,0,0,0,1,0,1,1,1,0,1,1,1,0,1,1,0,0,0)$$

Le destinataire possède le point de départ (x, y) du système et calcule donc les ensembles E_1, E_2, E_3. Il peut ainsi calculer la matrice génératrice de chacun des codes c_i qui est l'ensemble des vecteurs de convolutions entre les vecteurs v_j ayant un seul

1 à l'indice E_{i_j} et la fonction τ. Nous mettons les matrices génératrices sous la forme systématique et nous en déduisons les matrices de parité H_i suivantes :

$$H_1 = \begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&1&0 \\ 0&1&0&0&0&0&0&0&0&0&1&1&0&1&1&1&0&1&0&1&0 \\ 0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&1&0&0&0&0&1&0&0&0&0&1&1&1&1&1&1&1&0 \\ 0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&1&1&0&0&0&1 \\ 0&0&0&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0&0&1&0 \\ 0&0&0&0&0&0&1&0&0&1&0&1&1&0&1&1&1&0&0&0&1 \\ 0&0&0&0&0&0&0&1&1&1&1&0&1&0&0&1&0&0&1&1&0 \end{pmatrix}$$

$$H_2 = \begin{pmatrix} 1&0&0&0&0&0&0&0&1&1&0&0&1&1&0&0&1&0&1&0&0 \\ 0&1&0&0&0&0&0&0&0&1&0&1&0&1&1&0&0&1&1&1&0 \\ 0&0&1&0&0&0&0&0&1&1&1&0&0&0&0&1&0&0&0&1&1 \\ 0&0&0&1&0&0&0&0&0&0&1&1&1&1&0&0&0&1&1&0&1 \\ 0&0&0&0&1&0&0&0&0&1&0&1&1&0&1&0&0&0&1&1&1 \\ 0&0&0&0&0&1&0&0&1&0&0&1&0&0&1&1&1&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&1&0&1&1&0&1&0&0&0&1&1&0 \\ 0&0&0&0&0&0&0&1&1&1&0&1&1&0&1&0&0&0&1&1&0 \end{pmatrix}$$

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

Nous calculons le syndrome s_i de chaque mot reçu r_i à l'aide de H_i et nous obtenons :

$$s_1 = {r_1}^t H_1 = (0,0,1,0,0,0,0,0)$$
$$s_2 = {r_2}^t H_2 = (0,0,0,0,0,0,0,1)$$
$$s_3 = {r_3}^t H_3 = (1,1,1,0,0,1,0,0)$$

Le syndrome s_1 est identique à la $3^{\text{ème}}$ colonne de H_1, donc le $3^{\text{ème}}$ bit du mot reçu est erroné, ainsi le mot de code reçu est

$$c_1 = (0,0,0,1,1,0,0,0,1,1,0,1,1,1,1,1,0,0,0,0)$$

Le syndrome s_2 est identique à la $8^{\text{ème}}$ colonne de H_2 donc le $8^{\text{ème}}$ bit du mot reçu est erroné, ainsi le mot de code reçu est

$$c_2 = (1,1,0,0,0,1,1,1,0,0,1,1,1,1,0,0,0,0,1,0,1)$$

Enfin, le syndrome s_3 est identique à la $20^{\text{ème}}$ colonne de H_3, donc le $20^{\text{ème}}$ bit du mot reçu est erroné, ainsi le mot de code reçu est

$$c_3 = (1,1,1,0,0,0,1,0,1,1,1,0,1,1,1,0,1,1,0,1,0)$$

On retrouve bien les mots de code envoyés par l'expéditeur.

Pour remonter au message encodé, le destinataire calcule τ^{-1} sur le groupe. Pour calculer cette inverse, on calcule tout d'abord la matrice M associée à τ dans $M_{21}(\mathbb{F}_2)$ (définie dans le critère 2.5), puis on calcule l'inverse M^{-1} de cette matrice, et enfin on calcule la fonction τ^{-1} associée à M^{-1}. Nous obtenons ici :

$$\tau^{-1} = (0,0)+(0,2)+(1,0)+(1,1)+(2,0)+(2,1)+(2,2)+(4,2)+(5,1)+(5,2)+(6,0)$$

Avec cette fonction, on calcule le message sur le groupe en effectuant le produit de convolution entre c_i et τ^{-1}.

$$\begin{aligned}
u_1 = c_1 * \tau^{-1} &= ((1,0)+(1,1)+(2,2)+(3,0)+(3,2)+(4,0)+(4,1)+(4,2)+(5,0)\\
&\quad +(5,1)) \oplus ((0,0)+(0,2)+(1,0)+(1,1)+(2,0)+(2,1)+(2,2)+(4,2)+(5,1)\\
&\quad +(5,2)+(6,0))\\
&= (2,0)+(3,1)+(5,0)+(6,2)+(3,0)+(4,1)\\
u_2 = c_2 * \tau^{-1} &= ((0,0)+(0,1)+(1,2)+(2,0)+(2,1)+(3,1)+(3,2)+(4,0)+(4,1)\\
&\quad +(6,0)+(6,2)) \oplus ((0,0)+(0,2)+(1,0)+(1,1)+(2,0)+(2,1)+(2,2)+(4,2)\\
&\quad +(5,1)+(5,2)+(6,0))\\
&= (0,1)+(1,0)+(2,0)+(2,1)+(6,0)\\
u_3 = c_3 * \tau^{-1} &= ((0,0)+(0,1)+(0,2)+(2,0)+(2,2)+(3,0)+(3,1)+(4,0)+(4,1)\\
&\quad +(4,2)+(5,1)+(5,2)+(6,2)) \oplus ((0,0)+(0,2)+(1,0)+(1,1)+(2,0)+(2,1)\\
&\quad +(2,2)+(4,2)+(5,1)+(5,2)+(6,0))\\
&= (0,0)+(2,1)+(1,1)+(3,2)+(5,0)
\end{aligned}$$

Enfin, pour retrouver les messages u_i sous forme binaire, il faut effectuer la correspondance entre les u_i et les ensembles E_i. On a ainsi :

$$\begin{aligned}
E_1 &= ((4,2),(2,0),(3,1),(0,1),(4,0),(5,0),(6,0),(6,2),(5,2),(3,0),(0,0),\\
&\quad (1,1),(4,1))\\
u_1 &= (0,1,1,0,0,1,0,1,0,1,0,0,1)\\
E_2 &= ((0,1),(5,1),(1,1),(2,2),(1,0),(4,2),(3,1),(6,1),(2,0),(2,1),(4,0),\\
&\quad (5,0),(6,0))\\
u_2 &= (1,0,0,0,1,0,0,0,1,1,0,0,1)\\
E_3 &= ((6,0),(0,0),(1,0),(2,1),(0,1),(4,0),(1,1),(3,2),(2,0),(4,2),(3,0),\\
&\quad (5,0),(1,2))\\
u_3 &= (0,1,0,1,0,0,1,1,0,0,0,1,0)
\end{aligned}$$

Le destinataire a bien retrouvé le message $U = (u_1, u_2, u_3)$.

5.8 Conclusion

Dans ce chapitre, nous avons remplacé le groupe des entiers $\mathbb{Z}$ par un groupe de la forme $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ avec N, M impairs et $PGCD(M, \varphi(N)) \neq 1$. Nous avons étudié plus particulièrement le cas $N = 7$ et $M = 3$ dont nous avons défini la transformée de Fourier. Cette dernière permet de remplacer la convolution de deux fonctions de $\mathbb{F}_2^{\mathbb{Z}/7\mathbb{Z}\rtimes\mathbb{Z}/3\mathbb{Z}}$ par la transformée de Fourier inverse de leur transformée de Fourier. Nous avons ainsi constaté que les fonctions de transfert τ qui rendent le codage injectif étaient celles qui n'avaient que des termes inversibles dans leur transformée de Fourier.

Nous avons ensuite utilisé un domaine fondamental de l'action de ce groupe sur le plan euclidien, ce qui nous a permis d'obtenir un tore pavé par des cases, chacune des cases représentant un élément du groupe. Sur ce tore, nous avons tout d'abord fait parcourir une droite de pente irrationnelle et noté toutes les cases qu'elle traversait, tout ceci dans le but de définir des intervalles d'encodage chaotiques de grande longueur. Malheureusement, la suite des cases traversées n'a pas le comportement chaotique qu'on espérait, et on observe l'apparition de nombreux cycles dans cette suite de cases. Pour s'assurer d'avoir un comportement chaotique, nous avons utilisé un système dynamique discret connu. On part d'un point (x, y) du carré unité et la matrice du système calcule un nouveau point du carré unité à chaque itération. Les suites d'éléments obtenus grâce à ce système ne permettent pas non plus d'avoir des intervalles d'encodage de longueur raisonnable pour construire un code de bon rendement. L'idée, pour avoir une bonne longueur d'intervalles d'encodage k, est de la choisir, puis de parcourir le domaine jusqu'à avoir les k cases voulues, tout en ignorant les cases déjà parcourues auparavant dans l'intervalle considéré. On a ainsi des intervalles chaotiques de longueur k aussi grande que possible. Pour calculer de tels intervalles, on doit au maximum parcourir 3 fois plus de cases du domaine.

A partir de ces intervalles, nous avons défini le code convolutif sur $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$. Le message u sur $\mathbb{F}_2^k$ est envoyé sur le groupe en utilisant un intervalle d'encodage E, puis est convolué avec une fonction de transfert τ. La correction d'erreur s'effectue par syndrome, puis le décodage est la convolution du mot de code reçu avec l'inverse de la fonction de transfert τ. On retrouve ensuite les bits du message envoyé en utilisant l'intervalle d'encodage E. Pour que l'expéditeur et le destinataire calculent tous deux les mêmes intervalles d'encodage, il est indispensable que leurs logiciels de calcul utilisent la même précision.

Pour certains k, ces codes ont la même distance minimale maximale que celle des meilleurs codes linéaires en bloc. Cependant, si nous utilisons le parcours chaotiques des éléments, nous ne pouvons choisir des ensembles E optimaux en terme de distance minimale. Seule la fonction de transfert peut être choisie pour avoir une bonne distance. Nous avons donc calculé la distance moyenne pour chaque fonction de transfert et constaté que cette distance moyenne est plus faible que la distance minimale des meilleurs codes linéaires en bloc.

Malgré leurs performances en tant que code correcteur moins bonnes que les codes déjà existants, ces codes sont intéressants car ils ont des propriétés cryptographiques. En effet, si on considère le point (x, y) de départ du système dynamique discret comme une clé symétrique partagée par Alice et Bob, on constate que, sans cette clé, une tierce personne ne parviendrait pas à décoder le mot de code envoyé par Alice à Bob.

L'utilisateur fait donc un compromis entre la sécurité et la capacité de correction en utilisant ces codes.

Chapitre 6

Étude des sous-groupes de congruence $\Gamma(N)$, $\Gamma'_0(N)$ et $\Gamma'_1(N)$

Dans le chapitre précédent, nous avons vu comment construire des codes convolutifs non-commutatifs sur des groupes de cardinal impair. Nous voulons maintenant construire des codes convolutifs sur des groupes de cardinal pair. Nous allons, pour cela, utiliser des systèmes dynamiques sur le plan hyperbolique, qui ont la propriété d'être chaotiques.

Nous allons étudier les sous-groupes de congruence notés $\Gamma(N)$, $\Gamma'_0(N)$ et $\Gamma'_1(N)$ qui agissent sur le plan hyperbolique. Tout d'abord, définissons la géométrie hyperbolique.

6.1 Géométrie hyperbolique

La géométrie hyperbolique est née suite aux critiques des cinq axiomes de la géométrie énoncés par Euclide [EPI66]. Ces axiomes sont :

1. Une droite peut être tracée d'un point à un autre point.
2. Une droite finie peut être étendue de façon continue en une droite.
3. Un cercle peut être tracé avec un centre et un rayon
4. Tous les angles droits sont égaux.
5. Si une droite tombant sur deux droites fait les angles intérieurs du même côté plus petits que deux angles droits, ces droites, prolongées à l'infini, se rencontreront du côté où les angles sont plus petits que deux angles droits.

Le cinquième axiome est équivalent à l'axiome moderne des parallèles de la géométrie euclidienne :

Par un point extérieur à une droite donnée, il y a une et une seule droite parallèle à la droite donnée.

Plusieurs mathématiciens ont critiqué la nécessité de ce cinquième axiome pour définir la géométrie et ont donc pensé que cet axiome pouvait être déduit des quatre précédents. Pendant 2000 ans, les mathématiciens ont tenté de prouver le cinquième axiome. C'est au $19^{\text{ème}}$ siècle qu'a été découvert que le cinquième axiome était indépendant des quatre autres. La preuve de cette indépendance a été le résultat d'une

découverte complètement inattendue. La négation du cinquième axiome a conduit à une nouvelle géométrie consistante. C'est Gauss qui a été le premier à faire cette découverte. Gauss a commencé son travail sur la théorie des parallèles vers 1792. Après avoir essayé de prouver le cinquième axiome pendant 20 ans, Gauss a découvert que la négation de cet axiome conduisait à une nouvelle géométrie étrange, appelée la géométrie non-euclidienne. Dans cette géométrie, Gauss a trouvé que la somme des angles d'un triangle est plus petite que 180°. Après avoir étudié ses propriétés pendant 10 ans et n'avoir découvert aucune inconsistance, Gauss a été convaincu de sa consistance. Malheureusement, Gauss n'a jamais publié ses résultats sur la géométrie non-euclidienne.

Après quelques années, la géométrie non-euclidienne a été redécouverte indépendamment par Lobachevsky [Lob30] et Bolyai [Bol32]. Lobachevsky a publié le premier article introduisant la géométrie non-euclidienne en 1829. Quelques années après, Bolyai a publié une explication de la géométrie non-euclidienne, indépendante de l'article de Lobachevsky.

Aujourd'hui la géométrie non-euclidienne de Gauss, Lobachevsky et Bolyai est appelé la géométrie hyperbolique et le terme de géométrie non-euclidienne fait référence à toute géométrie qui n'est pas euclidienne.

Il y a plusieurs modèles qui permettent de représenter le plan hyperbolique dans le plan euclidien. On peut trouver ces modèles et leurs propriétés dans [Rat06]. Dans chaque modèle, on peut définir une distance. Cette distance nous permet de définir le chemin le plus court entre deux points qu'on appelle une **géodésique** finie. C'est donc l'équivalent d'un segment de la géométrie euclidienne. Si on prolonge ce segment à l'infini, on obtient une droite, dont l'équivalent dans la géométrie hyperbolique sera une géodésique infinie. Par abus de langage, nous allons utiliser dans la suite le terme « géodésique » pour désigner les géodésiques finies et les géodésiques infinies.

L'axiome des parallèles dans la géométrie hyperbolique est donc le suivant :
Par un point extérieur à une géodésique donnée, il y a une infinité de géodésiques parallèles à la géodésique donnée.

Nous allons maintenant introduire les modèles que nous avons utilisé dans ce chapitre et dans le chapitre suivant.

6.1.1 Demi-plan de Poincaré

Un des modèles en dimension 2 de la géométrie hyperbolique est le demi-plan de Poincaré. Il est défini comme le demi-plan euclidien supérieur, c'est-à-dire :

$$\mathbb{H} = \{z = x + iy \in \mathbb{C} \mid y > 0\}$$

On le munit de la distance suivante.

Définition 6.1. *Soient $z = x + iy$ et $z' = x' + iy' \in \mathbb{H}$.*
La distance $d(z, z')$ entre z et z' est :

$$d(z, z') = arccosh\left(1 + \frac{(x - x')^2 + (y - y')^2}{2yy'}\right)$$

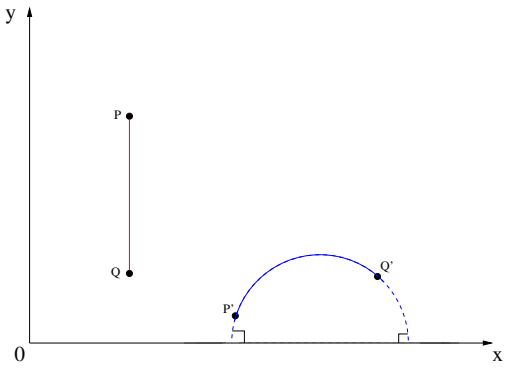


FIGURE 6.1 – Géodésiques sur le demi-plan de Poincaré

Les géodésiques de ce modèle sont les demi-droites parallèles à l'axe des ordonnées (en rouge) ou les demi-cercles perpendiculaires à l'axe des abscisses (en bleu), comme on peut le voir sur la figure 6.1.

6.1.2 Disque de Poincaré

Un autre modèle en dimension 2 est le disque de Poincaré $\mathbb{D}$. C'est le disque ouvert de centre 0 et de rayon 1, que l'on munit de la distance suivante :

Définition 6.2. *Soient $z = x + iy$ et $z' = x' + iy' \in \mathbb{D}$.*
La distance $d(z, z')$ entre z et z' est :

$$d(z, z') = arccosh\left(1 + \frac{(x - x')^2 + (y - y')^2}{(1 - (x^2 + y^2))(1 - (x'^2 + y'^2))}\right)$$

Dans ce modèle, les géodésiques sont les diamètres du disque (en bleu) et les arcs de cercles perpendiculaires au bord du disque (en rouge), comme on peut le voir sur la figure 6.2.

6.2 Étude des groupes

Dans ce chapitre, nous nous plaçons sur le demi-plan de Poincaré $\mathbb{H}$. Le groupe

$$PSL_2(\mathbb{R}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mid ad - bc = 1 \right\} / \{I, -I\}$$

avec I la matrice identité, agit à gauche sur l'ensemble des points de $\mathbb{H}$. Cette action est appelée la transformation de Möbius et est définie comme

$$z \mapsto \frac{az + b}{cz + d}$$

$PSL_2(\mathbb{Z})$, appelé le groupe modulaire, agit donc aussi sur les points de $\mathbb{H}$. Un domaine fondamental D de l'action de $PSL_2(\mathbb{Z})$ sur $\mathbb{H}$ est :

$$D = \left\{ z \in \mathbb{H} \mid -\frac{1}{2} < Re(z) \leqslant \frac{1}{2} \text{ et } |z| \geqslant 1 \text{ et si } Re(z) < 0 \text{ alors } |z| > 1 \right\}$$

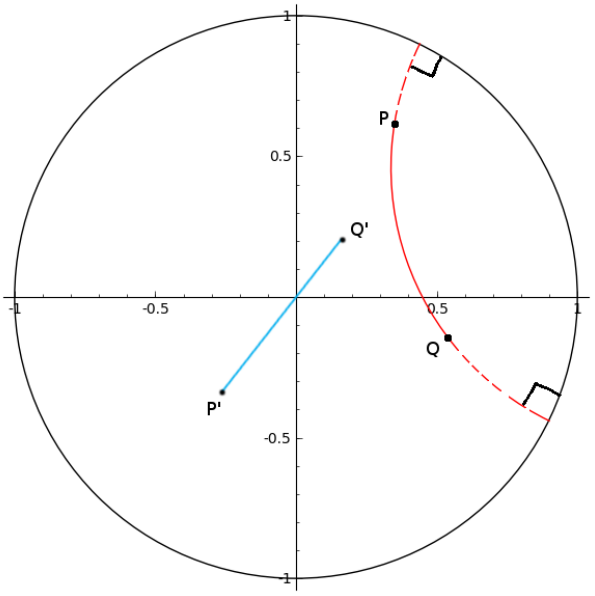


FIGURE 6.2 – Géodésiques sur le disque de Poincaré

Il est représenté par la figure [1] 6.3.

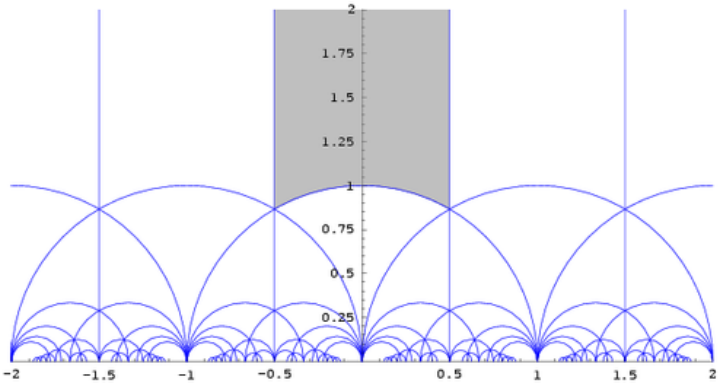


FIGURE 6.3 – Domaine fondamental (en gris) de l'action du groupe modulaire sur $\mathbb{H}$

Nous allons étudier $\Gamma(2)$ un sous-groupe de $PSL_2(\mathbb{Z})$ défini comme :

$$\Gamma(2) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in PSL_2(\mathbb{Z}),\ \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \mod 2 \right\}$$

Il est engendré par deux générateurs

$$\begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \text{ et } \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}$$

1. « ModularGroup-FundamentalDomain-01 » by Fropuff - from en.wikipedia. Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons http://commons.wikimedia.org/wiki/File:ModularGroup-FundamentalDomain-01.png

Soit Δ le triangle de $\mathbb{H}$ de sommets ∞, -1 et 1. Plus précisément,

$$\Delta = \{z = x + iy \mid \ -1 < x < 1 \text{ et } |z| > 1\}$$

Soit Δ' le triangle de $\mathbb{H}$ de sommets 0, 1, -1. Plus précisément,

$$\Delta' = \left\{z = x + iy \mid |z| < 1, \left|z + \frac{1}{2}\right| > \frac{1}{2} \text{ et } \left|z - \frac{1}{2}\right| > \frac{1}{2}\right\}$$

L'union $\Delta \cup \Delta'$ est un domaine fondamental pour l'action de $\Gamma(2)$, noté $\Delta(2)$ dans la suite. Dans ce domaine fondamental, le côté $[-1, \infty]$ est envoyé par transformation de Möbius sur le côté $[1, \infty]$ et le côté $[-1, 0]$ est envoyé sur le côté $[1, 0]$. Ce domaine fondamental est représenté sur la figure 6.4 avec en rouge, les côtés extérieurs au domaine, en rose le côté séparant les triangles Δ et Δ' et en pointillé, les côtés extérieurs qui ne sont pas dans le domaine.

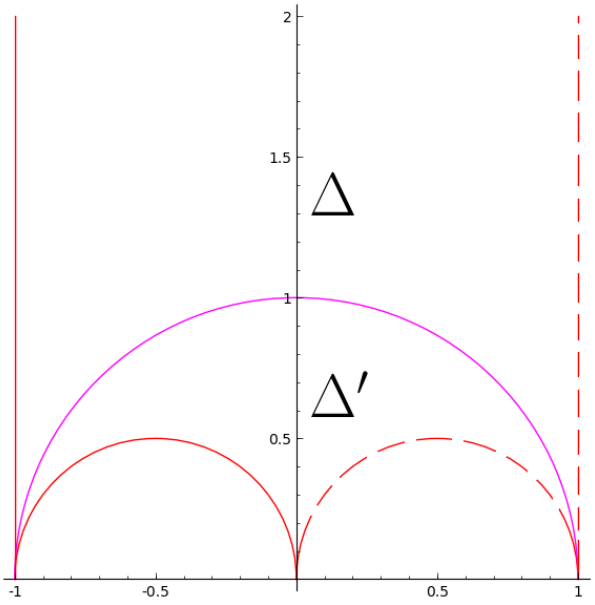


FIGURE 6.4 – Domaine fondamental de l'action de $\Gamma(2)$ sur $\mathbb{H}$

6.2.1 Étude de $\Gamma(N)$

Soit $N > 2$ un entier pair et soit ϕ le morphisme de réduction modulo N c'est-à-dire :

$$\begin{gathered}\phi : \Gamma(2) \to PSL_2(\mathbb{Z}/N\mathbb{Z}) \\ \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \mapsto \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \mod N \\ \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \mapsto \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \mod N\end{gathered}$$

On a

$$H = Ker(\phi) = \Gamma(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in PSL_2(\mathbb{Z}) \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} = I \mod N \right\}$$

Soit $n = Card(Im(\phi)) = Card(\Gamma(2)/\Gamma(N))$. Pour tout $g_i \in \Gamma(2)/\Gamma(N)$, $i = 1, \ldots, n$, un domaine fondamental $\Delta(N)$ de $\Gamma(N)$ est :

$$\Delta(N) = \bigcup_{i=1}^{n} g_i(\Delta(2))$$

avec $\Delta(2)$ un domaine fondamental de $\Gamma(2)$. Dans la suite, chaque $g_i(\Delta(2))$ va être appelé une face du domaine. On peut donc associer à chaque face, un élément g_i du quotient $\Gamma(2)/\Gamma(N)$.

Pour chaque coté extérieur γ de $\Delta(N)$, il existe un unique $g_\gamma \in \Gamma(N)$ tel que g_γ est un autre côté extérieur de $\Delta(N)$. Les éléments g_γ constituent une famille génératrice de $\Gamma(N)$.

Exemple 6.1. *Exemple pour* $\Gamma(4)$ *:*

$$n = Card(Im(\Gamma(2) \to PSL_2(\mathbb{Z}/4\mathbb{Z})))$$

On a :

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \in PSL_2(\mathbb{Z}/4\mathbb{Z})$$
$$A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \in PSL_2(\mathbb{Z}/4\mathbb{Z})$$
$$B = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \in PSL_2(\mathbb{Z}/4\mathbb{Z})$$
$$AB = \begin{pmatrix} 5 & 2 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} \mod 4 = BA \mod 4$$

$A^2, B^2, ABA, BAB, \ldots$ *sont égaux aux 4 matrices ci-dessus modulo 4. On a donc* $n = 4$*. Les représentants des classes de* $\Gamma(2)/\Gamma(4)$ *sont donc*

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}, \begin{pmatrix} 5 & 2 \\ 2 & 1 \end{pmatrix}$$

On applique :

$$\Delta(4) = \bigcup_{i=1}^{4} g_i(\Delta(2))$$

et on obtient un domaine fondamental 6.5 avec en rouge les côtés extérieurs au domaine, en rose l'image du côté rose de $\Gamma(2)$ *qui sépare les deux triangles d'une même face et en bleu les côtés intérieurs séparant les faces.*

Calculons quelles arêtes sont identifiées entre elles. Le tableau 6.1 représente par quelle matrice une arête de $\Delta(2)$ *est envoyé sur une arête de* $\Delta(4)$*.*

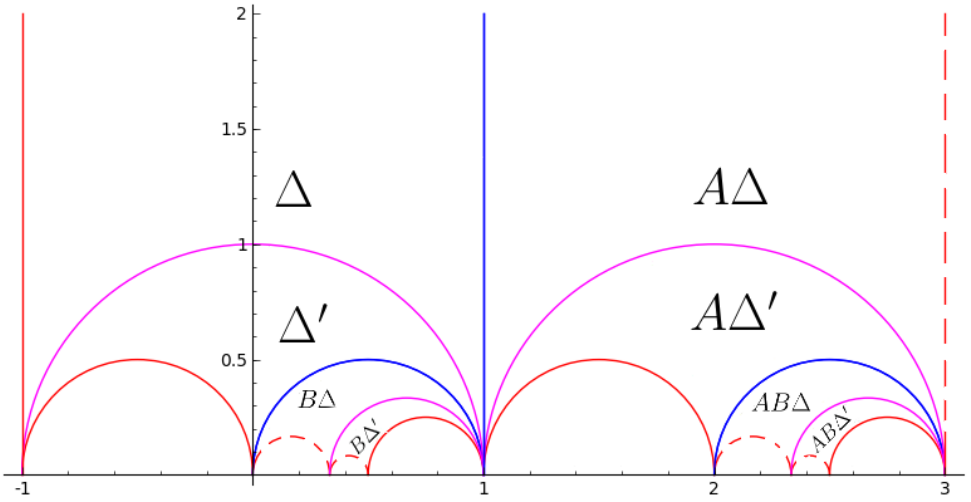


FIGURE 6.5 – Domaine fondamental de l'action de $\Gamma(4)$ sur $\mathbb{H}$

	$\begin{pmatrix}1&0\\0&1\end{pmatrix}$	$\begin{pmatrix}1&2\\0&1\end{pmatrix}$	$\begin{pmatrix}1&0\\2&1\end{pmatrix}$	$\begin{pmatrix}5&2\\2&1\end{pmatrix}$
$[-1,\infty]$	$[-1,\infty]$	$[1,\infty]$	$\left[1,\frac{1}{2}\right]$	$\left[3,\frac{5}{2}\right]$
$[-1,0]$	$[-1,0]$	$[1,2]$	$[1,0]$	$[3,2]$
$[0,1]$	$[0,1]$	$[2,3]$	$\left[0,\frac{1}{3}\right]$	$\left[2,\frac{7}{3}\right]$
$[1,\infty]$	$[1,\infty]$	$[3,\infty]$	$\left[\frac{1}{3},\frac{1}{2}\right]$	$\left[\frac{7}{3},\frac{5}{2}\right]$

TABLE 6.1 – Liens entre les arêtes de $\Delta(2)$ et celles de $\Delta(4)$

On peut voir dans ce tableau que $[-1,\infty]$ *s'envoie sur* $[1,\infty]$ *qui lui-même s'envoie sur* $[3,\infty]$ *et cela par le produit de matrices*

$$\begin{pmatrix}1&2\\0&1\end{pmatrix}\begin{pmatrix}1&2\\0&1\end{pmatrix}=\begin{pmatrix}1&4\\0&1\end{pmatrix}\in\Gamma(4)$$

$[-1,\infty]$ *est donc identifié à* $[1,\infty]$. *On peut aussi constater que* $[-1,0]$ *s'envoie sur* $[1,0]$ *qui s'envoie sur* $[2,\frac{7}{3}]$ *par le produit de matrices*

$$\begin{pmatrix}5&2\\2&1\end{pmatrix}\begin{pmatrix}1&0\\2&1\end{pmatrix}=\begin{pmatrix}9&2\\4&1\end{pmatrix}\notin\Gamma(4)$$

donc $[-1,0]$ *ne s'identifie pas à* $[2,\frac{7}{3}]$. *Cependant* $[-1,0]$ *s'envoie sur* $\left[\frac{1}{3},0\right]$ *par :*

$$\begin{pmatrix}1&0\\2&1\end{pmatrix}\begin{pmatrix}1&0\\2&1\end{pmatrix}=\begin{pmatrix}1&0\\4&1\end{pmatrix}\in\Gamma(4)$$

De même, $[1,2]\to[-1,0]\to[1,0]\to\left[\frac{7}{3},2\right]$ *par :*

$$\begin{pmatrix}5&2\\2&1\end{pmatrix}\begin{pmatrix}1&0\\2&1\end{pmatrix}\begin{pmatrix}1&0\\2&1\end{pmatrix}^{-1}=\begin{pmatrix}9&-16\\4&-7\end{pmatrix}\in\Gamma(4)$$

et $\left[1, \frac{1}{2}\right] \to [-1, \infty] \to [1, \infty] \to \left[\frac{7}{3}, \frac{5}{2}\right]$ *par :*

$$\begin{pmatrix} 5 & 2 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} -19 & 12 \\ -8 & 5 \end{pmatrix} \in \Gamma(4)$$

Enfin, $\left[3, \frac{5}{2}\right] \to [-1, \infty] \to [1, \infty] \to \left[\frac{1}{3}, \frac{1}{2}\right]$ *par :*

$$\begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 5 & 2 \\ 2 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} -3 & 8 \\ -8 & 21 \end{pmatrix} \in \Gamma(4)$$

Ces matrices constituent une famille génératrice de $\Gamma(4)$.

6.2.2 Étude de $\Gamma'_1(N)$ et de $\Gamma'_0(N)$

Nous nous sommes aussi intéressés à d'autres quotients $\Gamma(2)/H$ qui ne sont pas nécessairement des groupes. Soient

$$\Gamma_1(N) = \left\{ A \in PSL_2(\mathbb{Z}) \mid A = \begin{pmatrix} 1 & * \\ 0 & 1 \end{pmatrix} \mod N \right\}$$

et

$$\Gamma_0(N) = \left\{ A \in PSL_2(\mathbb{Z}) \mid A = \begin{pmatrix} * & * \\ 0 & * \end{pmatrix} \mod N \right\}$$

On s'intéresse aux groupes :

$$H = \Gamma'_1(N) = \Gamma_1(N) \cap \Gamma(2) \text{ et } H = \Gamma'_0(N) = \Gamma_0(N) \cap \Gamma(2)$$

Un domaine fondamental de ces groupes se construit de façon similaire à celui de $\Gamma(N)$ en cherchant un système de représentants g_i de $\Gamma(2)/H$ et en calculant

$$\Delta(N) = \bigcup_{i=1}^{Card(\Gamma(2)/H)} g_i(\Delta(2))$$

Par exemple, la figure 6.6 représente un domaine fondamental de $\Gamma'_0(4)$ et $\Gamma'_1(4)$ (leurs domaines sont les mêmes).

6.3 Parcours des éléments

Comme dans le chapitre précédent, chaque face du domaine $g_i(\Delta(2))$ est associé à l'élément g_i du quotient $\Gamma(2)/H$. De plus, chaque côté extérieur du domaine est identifié à un autre côté extérieur. On peut donc choisir une géodésique qui parcourt les faces. En effet, lorsque la géodésique sort du domaine par un côté extérieur g_γ, on cherche le côté identifié à g_γ et la géodésique rentre dans le domaine par ce côté identifié. Nous pouvons ainsi noter l'élément qui correspond à chaque face parcourue et avoir une suite d'éléments du quotient $\Gamma(2)/H$. Cette suite va être utilisée pour créer des intervalles d'encodage, de telle sorte qu'un intervalle se termine lorsqu'un élément de la suite se répète. On utilise ce dernier élément comme premier élément du prochain intervalle.

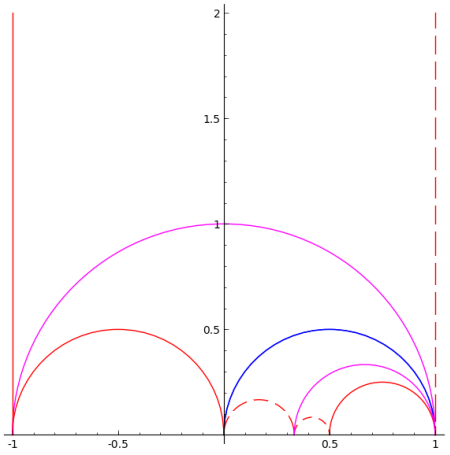


FIGURE 6.6 – Domaine fondamental de l'action de $\Gamma'_0(4)$ et de l'action de $\Gamma'_1(4)$ sur $\mathbb{H}$

6.3.1 Parcours sur les domaines de $\Gamma(8)$, $\Gamma'_1(16)$ et $\Gamma'_0(42)$

Nous allons étudier les intervalles d'encodage de trois quotients de $\Gamma(2)$, tous de même cardinal, afin de comparer les résultats obtenus. Ces quotients sont $\Gamma(2)/\Gamma(8)$, $\Gamma(2)/\Gamma'_1(16)$ et $\Gamma(2)/\Gamma'_0(42)$. Ils sont tous de cardinal 32. Pour calculer ces intervalles d'encodage, nous parcourons les faces du domaine à l'aide d'une géodésique « irrationnelle » c'est-à-dire dont la valeur centre+rayon est irrationnelle. Nous faisons ceci pour être sûr que la géodésique n'intersecte jamais un sommet de faces, car les sommets sont toujours des nombres rationnels. Pour calculer la longueur de ces intervalles d'encodage, nous avons utilisé l'algorithme 6.1.

Algorithme 6.1 Calcul des longueurs des intervalles d'encodage

Entrée: Quotient considéré
Sortie: Le tableau des longueurs des intervalles d'encodage $TabLongEncod$
  Construire un domaine associé au quotient
  Pour i de 1 à 100 **Faire**
    Tirer aléatoirement $geod_i = (centre_i, rayon_i)$ tel que $centre_i + rayon_i \in \mathbb{R}\backslash\mathbb{Q}$ et $centre_i + rayon_i$ soit entre les bords verticaux du domaine
    Tant que $longueur(FacesTraversees_i) < 320$ **Faire**
      Parcourir le domaine avec $geod$
      Noter chaque face traversée dans le tableau $FacesTraversees_i$
    Fin Tant que
    Calculer les longueurs des intervalles d'encodage associées à $FacesTraversees_i$ et les mettre dans le tableau $longencod_i$.
    Concaténer $TabLongEncod$ et $longencod_i$.
  Fin Pour
  Retourner $TabLongEncod$

Nous avons donc implémenté cet algorithme en Sage et calculé le pourcentage d'apparition de chaque longueur des intervalles d'encodage. Les résultats pour chacun des quotients sont respectivement dans les figures 6.7, 6.8 et 6.9.

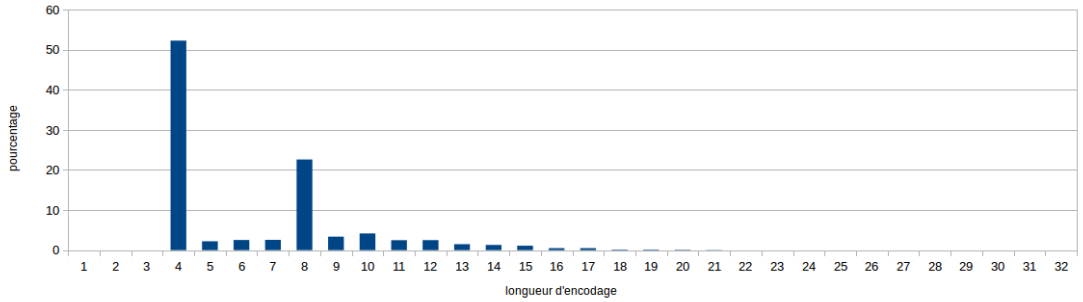


FIGURE 6.7 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma(8)$

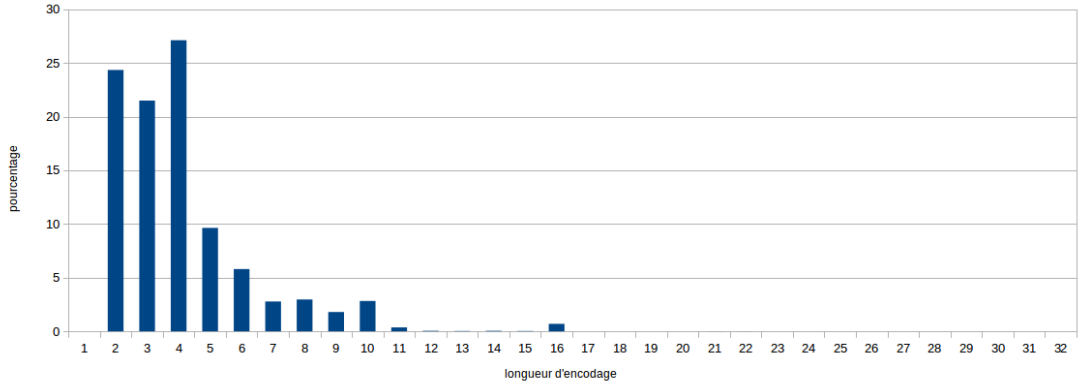


FIGURE 6.8 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_1(16)$

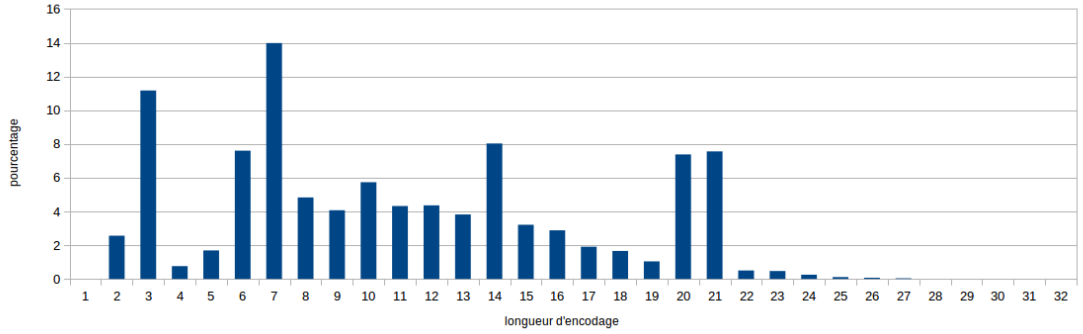


FIGURE 6.9 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_0(42)$

Pour le quotient $\Gamma(2)/\Gamma(8)$, on peut constater que 75% des longueurs des intervalles d'encodage sont soit 4, soit 8, ce qui est très petit devant 32, le cardinal du quotient.

Pour le quotient $\Gamma(2)/\Gamma_1'(16)$, 70% des longueurs des intervalles d'encodage sont 2, 3 ou 4, ce qui est encore plus petit par rapport à 32.

Enfin pour le quotient $\Gamma(2)/\Gamma_0'(42)$, on a certes deux petits pics en 20 et 21, mais 80% des intervalles ont une longueur plus petite que 16.

On cherche quelle est la cause de ces petites longueurs des intervalles d'encodage. Pour $\Gamma(2)/\Gamma(8)$ on a, par exemple, la suite de faces suivantes (que nous avons numéroté de 1 à 32), séparées par intervalle d'encodage :

$$
\begin{aligned}
&0, 1, 3, 4,\\
&0, 1, 19, 26, 22, 23, 27, 6, 5, 11, 7, 13, 28, 30,\\
&27, 6, 5, 18, 25, 26, 28, 30,\\
&27, 23, 24, 14, 22,\\
&23, 24, 14, 22,\\
&23, 24, 14, 22,\\
&23, 24, 14, 22
\end{aligned}
$$

On peut voir qu'à la fin de cette suite, on a un cycle, c'est-à-dire une suite d'éléments qui se répète un certain nombre de fois. Cela signifie que la géodésique qui parcourt les faces, « tourne » autour d'une pointe. Si on regarde d'autres exemples de géodésiques, on peut voir ce phénomène se répéter très souvent.

Pour $\Gamma(2)/\Gamma_1'(16)$, on peut constater le même phénomène avec, par exemple, cette suite d'éléments :

$$
\begin{aligned}
&0, 14, 10, 7, 3, 6, 11,\\
&7, 3, 6, 11,\\
&7, 3, 6, 11,\\
&7, 3, 6, 11,\\
&7, 3, 6, 11
\end{aligned}
$$

De même avec $\Gamma(2)/\Gamma_0'(42)$:

$$
\begin{aligned}
&22, 11, 5, 2, 1, 0,\\
&22, 11, 21, 17, 3, 6, 12, 31, 27, 7, 14, 15, 10, 20, 29,\\
&31, 12, 23, 19, 8, 4, 2, 1, 13, 6, 3, 7, 14, 15,\\
&8, 16, 20, 29, 30, 2, 1, 3, 17, 21, 11, 22, 14, 15,\\
&8, 16, 20, 29, 30, 2, 1, 3, 17, 21, 11, 22, 14, 15,\\
&8, 16, 20, 29, 30, 2, 1, 3, 17, 21, 11, 22, 14, 15,\\
&8, 16, 20, 29, 30, 2, 1, 3, 17, 21, 11, 22, 14, 15
\end{aligned}
$$

Ces cycles sont la cause de certaines petites longueurs des intervalles d'encodage. Voyons ce qui se passe si l'on supprime ces cycles dans le parcours des éléments. Nous

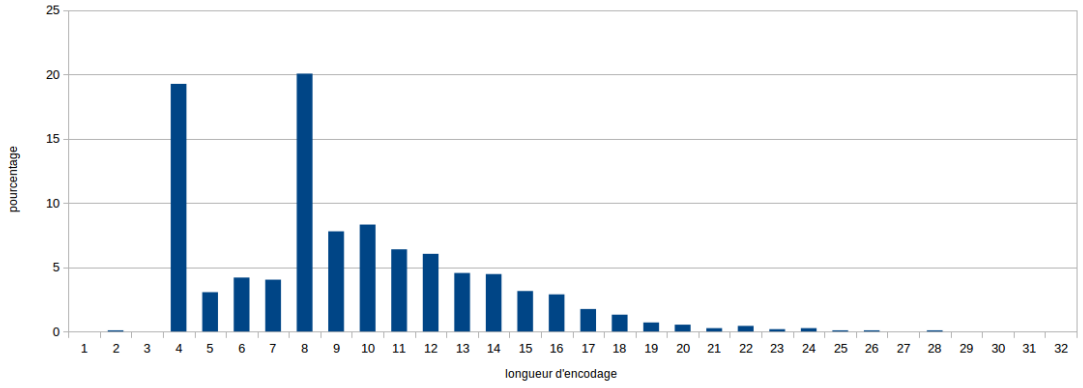


FIGURE 6.10 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma(8)$ en supprimant les cycles

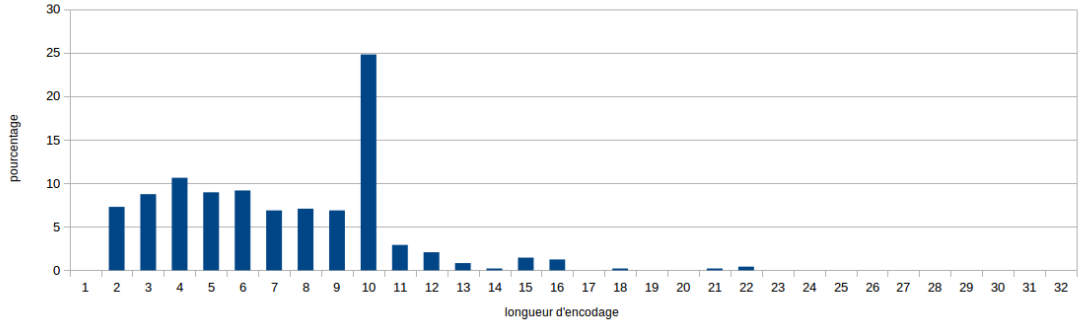


FIGURE 6.11 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_1(16)$ en supprimant les cycles

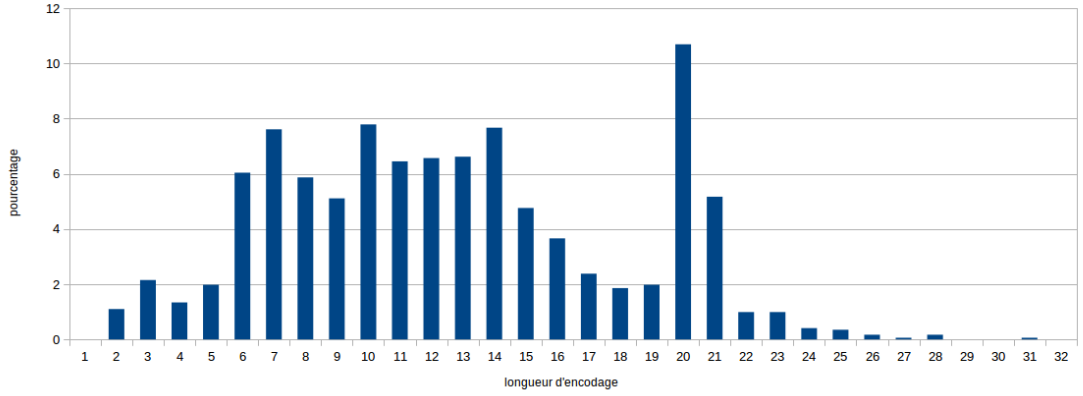


FIGURE 6.12 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur $\Gamma(2)/\Gamma'_0(42)$ en supprimant les cycles

avons, de nouveau, testé le parcours des éléments pour les trois quotients considérés précédemment, cette fois-ci en supprimant la répétition des cycles (c'est-à-dire que si on détecte l cycles, on garde le premier cycle et on supprime les $l-1$ suivants). On applique l'algorithme de suppression des cycles à chaque tableau $FacesTraversees_i$ de l'algorithme 6.1. Les résultats sont dans les graphiques 6.10, 6.11 et 6.12.

Pour $\Gamma(2)/\Gamma(8)$, on peut constater une amélioration des longueurs des intervalles d'encodage, la longueur 4 apparaissant presque trois fois moins souvent. Cependant, il y a encore beaucoup de petites longueurs.

Pour $\Gamma(2)/\Gamma'_1(16)$, on est passé de beaucoup de longueurs égales à 2, 3 ou 4, à une longueur 10 très présente, cependant, le rendement $\frac{10}{32}$ reste faible.

Pour $\Gamma(2)/\Gamma'_0(42)$, les pics en 3 et 7 se sont atténués, cependant, on a toujours 75% des intervalles qui sont de longueur $\leqslant 16$.

Les cycles ne sont donc pas les seuls responsables des petites longueurs des intervalles d'encodage. De plus, comme on a supprimé les cycles, on a également supprimé des intervalles d'encodage et donc des éléments du tableau $FacesTraversees$. Or, si on a un message de l bits à partager en blocs, la somme des longueurs des intervalles d'encodage doit être égale ou supérieure à l pour encoder tous les bits sur le groupe. Donc, si on supprime les cycles, il va falloir parcourir plus longtemps le domaine pour avoir les l éléments du groupe désirés. Nous allons maintenant voir combien d'éléments du quotient considéré, placés dans les intervalles d'encodage, on obtient quand on parcourt 32 000 faces du domaine et qu'on supprime les cycles. Les résultats sont dans le tableau 6.2.

	$\Gamma(2)/\Gamma(8)$	$\Gamma(2)/\Gamma'_1(16)$	$\Gamma(2)/\Gamma'_0(42)$
parcours classique	32 000	32 000	32 000
parcours en supprimant les cycles	11 195	4 637	20 938

TABLE 6.2 – Nombre d'éléments du quotient considéré placés dans les intervalles d'encodage si on parcourt 32 000 faces du domaine

On peut voir dans ce tableau que l'action de supprimer les cycles supprime beaucoup d'éléments, puisqu'il ne reste respectivement que 11 195, 4 637 et 20 938 éléments dans les intervalles d'encodage. Si on a un message à encoder de taille l, on doit parcourir $\sim 3 \times l$ faces dans le cas de $\Gamma(2)/\Gamma(8)$, $\sim 8 \times l$ faces dans le cas de $\Gamma(2)/\Gamma'_1(16)$ et $\sim \frac{3}{2} \times l$ faces dans le cas de $\Gamma(2)/\Gamma'_0(42)$, pour être sûr d'avoir assez d'éléments dans les intervalles d'encodage.

En conclusion, les longueurs des intervalles d'encodage restent trop petites en supprimant les cycles et de plus, le coût de calcul est multiplié respectivement par 3, 8 ou $\frac{3}{2}$ selon le quotient considéré.

6.3.2 Parcours sur des domaines construits à partir de copies de $\Gamma(2)$

Pour s'assurer du résultat précédent et vérifier que nos résultats ne sont pas liés à la particularité des quotients choisis, nous avons construit des quotients de $\Gamma(2)$ de façon aléatoire. Pour cela, nous avons construit un domaine connexe à partir

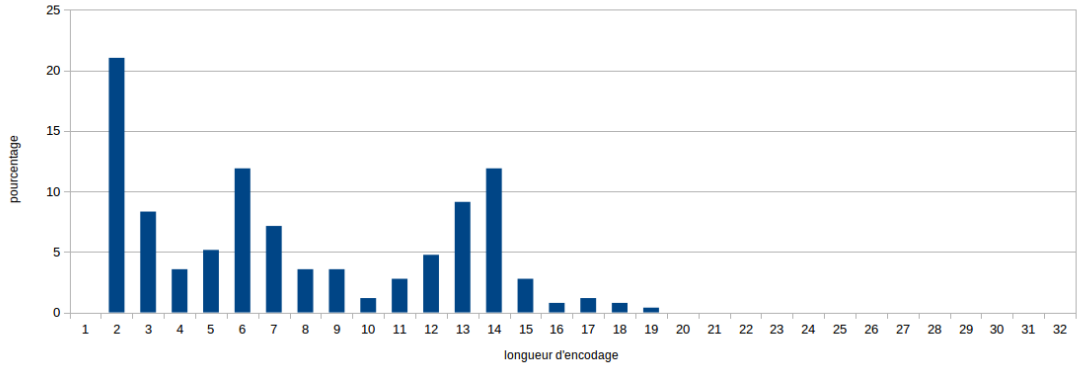


FIGURE 6.13 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur un groupe $\Gamma(2)/H_1$ construit aléatoirement de cardinal 32 en supprimant les cycles

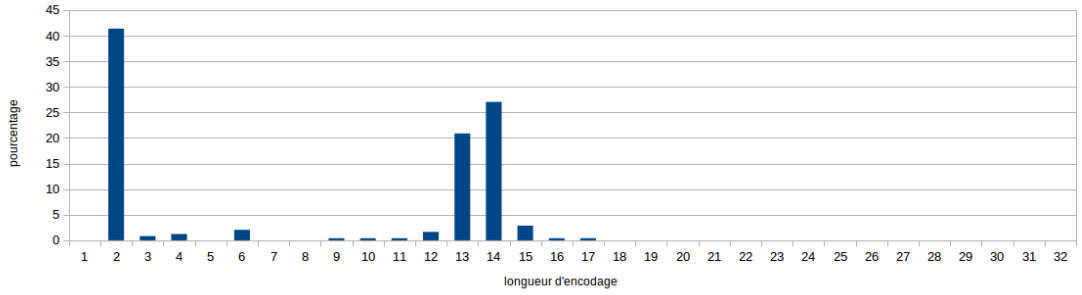


FIGURE 6.14 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur un groupe $\Gamma(2)/H_2$ construit aléatoirement de cardinal 32 en supprimant les cycles

de copies du domaine de $\Gamma(2)$ et nous avons choisi aléatoirement les identifications des côtés extérieurs. On a ainsi un domaine d'un groupe H et on peut définir le quotient $\Gamma(2)/H$. Pour comparer ces quotients avec les quotients précédents, nous avons pris 32 copies de $\Gamma(2)$ pour construire un domaine de H. Ainsi $\Gamma(2)/H$ est de cardinal 32. On a ainsi construit 3 domaines aléatoirement qu'on a parcouru comme précédemment et on a calculé les longueurs des intervalles d'encodage obtenues, en supprimant les cycles. Les résultats sont dans les figures 6.13, 6.14, 6.15

Dans les deux premiers cas, on obtient respectivement plus de 20% et plus de 40% d'intervalles d'encodage de longueur 2, ce qui est très faible. Ces résultats sont pires que dans les cas précédents.

Dans le troisième cas, on a une répartition beaucoup plus homogène des longueurs, cependant on a encore 78% des intervalles qui sont de longueurs $\leqslant 16$, ce qui va donner des codes de rendement $\leqslant \frac{1}{2}$.

Donc on constate qu'il y a toujours de nombreuses petites longueurs d'intervalles d'encodage, ce qui n'est pas satisfaisant en terme de rendement. Cela vient également des sommets de faces dont certaines géodésiques font le tour.

6.4 Conclusion

Nous avons tout d'abord étudié la longueurs des intervalles d'encodage sur les quotients de cardinal 32, $\Gamma(2)/\Gamma(8)$, $\Gamma(2)/\Gamma'_1(16)$ et $\Gamma(2)/\Gamma'_0(42)$ et nous avons constaté qu'il y avait de nombreux intervalles de petite longueur et certaines longueurs étaient bien plus probables que d'autres. Certains de ces intervalles de petites longueurs sont en fait des cycles, c'est-à-dire que la géodésique qui parcourt les faces du domaine fait le tour de certaines sommets de faces un certain nombre de fois ce qui engendre la même suite d'éléments.

Ces cycles étant très nombreux, nous avons décidé de tester le même algorithme en supprimant les répétitions de cycles et on a obtenu des intervalles plus longs sans cependant avoir une longueur acceptable pour un bon rendement.

Puis nous avons testé si nos quotients considérés n'étaient pas des cas particuliers défavorables, en définissant des quotients de $\Gamma(2)$ à partir d'un domaine fondamental construit aléatoirement à partir de copies du domaine de $\Gamma(2)$ et d'identifications de côtés aléatoires. Nous avons constaté qu'il y avait dans certains cas, de pires résultats que les quotients considérés précédemment et dans d'autres cas, de meilleurs résultats sans toutefois avoir des longueurs d'intervalles d'encodage acceptables.

Ces petites longueurs d'intervalles d'encodage étant liées aux sommets de faces autour duquel certaines géodésiques tournent, nous allons nous placer, dans le chapitre suivant, dans des domaines où il n'y a pas de sommet de faces n'appartenant pas au domaine, afin de contourner le problème.

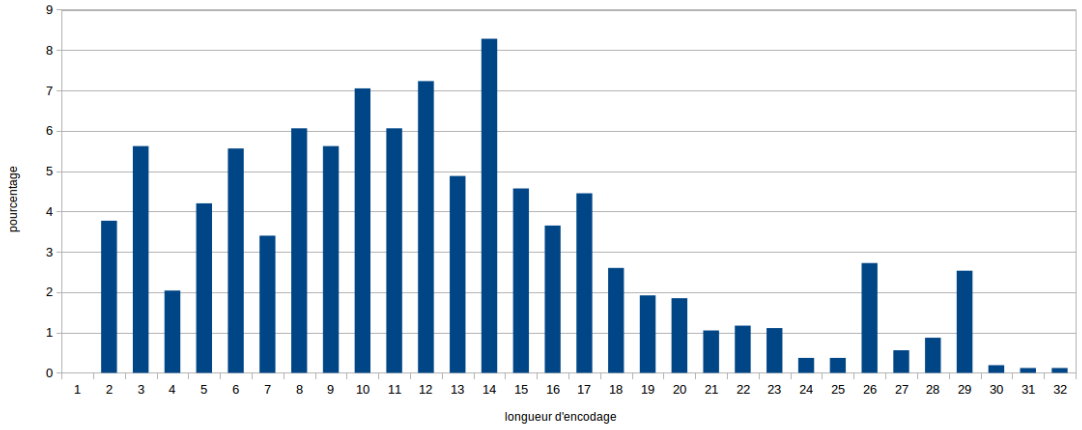


FIGURE 6.15 – Pourcentages de chaque longueur des intervalles d'encodage calculés sur un groupe $\Gamma(2)/H_3$ construit aléatoirement de cardinal 32 en supprimant les cycles

Chapitre 7

Codes sur des quotients finis du groupe du triangle

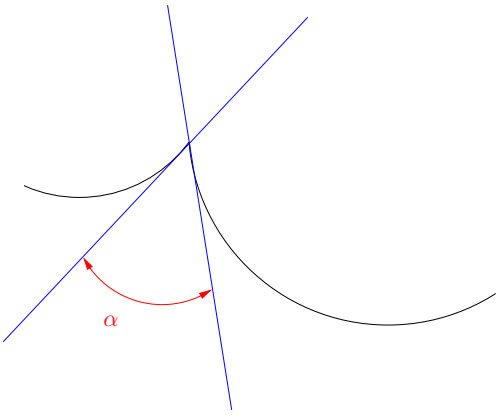
Le chapitre précédent n'ayant pas permis d'avoir des intervalles d'encodage de longueur suffisante à cause des sommets de face n'appartenant pas au domaine, nous allons maintenant travailler sur des quotients finis du groupe du triangle.

7.1 Groupe du triangle et construction

Nous nous plaçons sur un autre modèle de la géométrie hyperbolique, le disque de Poincaré, défini dans la partie 6.1.2.

Soient $a, b, c \in \mathbb{N}\backslash\{0, 1\}$. Un groupe du triangle $T(a, b, c)$ est le groupe généré par les réflexions des côtés d'un triangle d'angles $\frac{\pi}{a}$, $\frac{\pi}{b}$ et $\frac{\pi}{c}$.

Définition 7.1. *Un* ***angle hyperbolique*** *entre deux géodésiques sur le disque de Poincaré est l'angle euclidien mesuré entre les tangentes aux deux géodésiques au niveau du sommet de l'angle, comme on peut le voir sur l'image ci-dessous.*



Par conséquence, si les réflexions engendrant le groupe sont notées $\sigma_1, \sigma_2, \sigma_3$ (représentées dans la figure 7.1) et les angles entre les côtés sont dans l'ordre donné précédemment, alors on a :

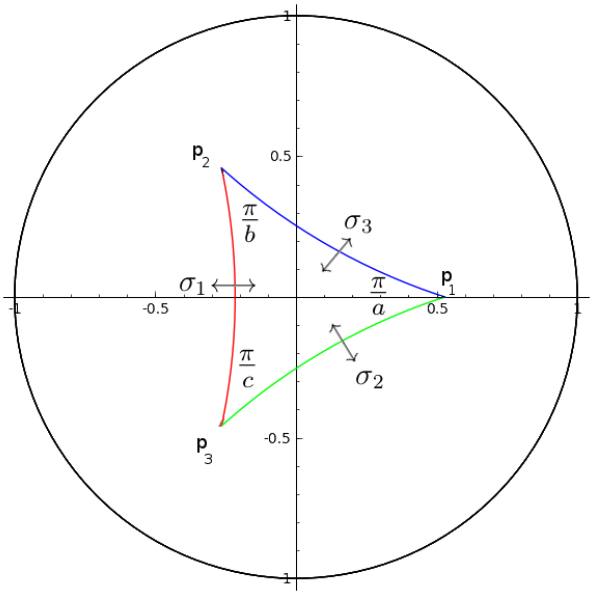


FIGURE 7.1 – Triangle hyperbolique et réflexions σ_1, σ_2 et σ_3

1. $\sigma_1^2 = \sigma_2^2 = \sigma_3^2 = 1$
2. $(\sigma_1\sigma_2)^c = (\sigma_2\sigma_3)^a = (\sigma_3\sigma_1)^b = 1$

Le groupe du triangle $T(a, b, c)$ admet donc pour présentation :

$$T(a,b,c) = \langle \sigma_1, \sigma_2, \sigma_3 \mid \sigma_1^2 = \sigma_2^2 = \sigma_3^2 = (\sigma_1\sigma_2)^c = (\sigma_2\sigma_3)^a = (\sigma_3\sigma_1)^b = 1 \rangle$$

Dans le cas d'un groupe du triangle sur le plan hyperbolique, on a la condition que

$$\frac{1}{a} + \frac{1}{b} + \frac{1}{c} < 1 \tag{7.1}$$

On peut voir un exemple d'un tel groupe pour $a = 2, b = 3, c = 7$ dans la figure 7.2.

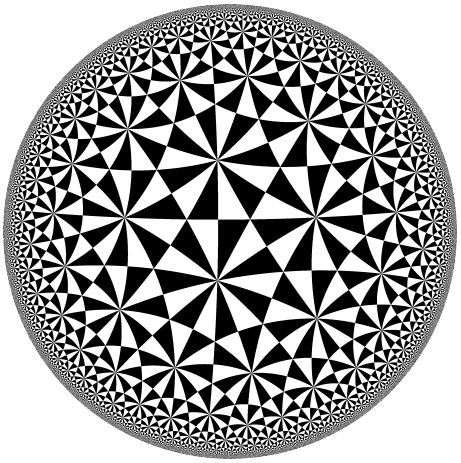


FIGURE 7.2 – Groupe du triangle $T(2, 3, 7)$

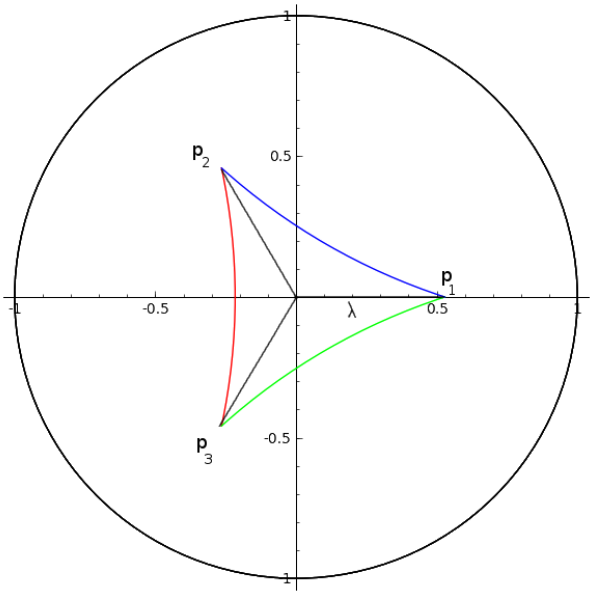


FIGURE 7.3 – Points p_1, p_2, p_3 sur le disque de Poincaré

Nous avons choisi un cas particulier de ces groupes du triangle, en prenant un triangle équilatéral, c'est-à-dire $a = b = c$. On appelle α l'angle $\frac{\pi}{a}$. On applique la condition 7.1 et on a :

$$3 \times \frac{1}{a} < 1 \Rightarrow a > 3 \Rightarrow \alpha < \frac{\pi}{3}$$

Pour tracer un triangle équilatéral sur le disque de Poincaré avec α compris entre 0 et $\frac{\pi}{3}$, nous allons prendre 3 points (p_1, p_2, p_3) de la forme :

$$\begin{aligned} p_1 &= \lambda \\ p_2 &= \lambda j \\ p_3 &= \lambda j^2 \end{aligned}$$

avec $0 < \lambda < 1$ et j le nombre complexe égal à $-\frac{1}{2} + \frac{\sqrt{3}}{2}i$. Nous obtenons le triangle de la figure 7.3.

Nous allons tout d'abord calculer l'angle α à partir de λ. C'est l'angle entre deux côtés du triangle. Pour cela, nous calculons les équations des géodésiques passant par p_1 et p_2 et par p_1 et p_3 qui sont des cercles orthogonaux au cercle unité donc d'équation $x^2 + y^2 + ax + by + 1 = 0$. La formule générale connaissant deux points $x_1 + iy_1$ et $x_2 + iy_2$ est donc (avec $c + id$ le centre du cercle et r le rayon) :

$$\begin{aligned} c + id &= -\frac{a}{2} - i\frac{b}{2} \\ r &= \sqrt{\frac{a^2}{4} + \frac{b^2}{4} - 1} \end{aligned}$$

avec :

$$a = \frac{y_1(x_2^2 + y_2^2) - y_2(x_1^2 + y_1^2) + y_1 - y_2}{x_1 y_2 - y_1 x_2}$$
$$b = \frac{x_2(x_1^2 + y_1^2) - x_1(x_2^2 + y_2^2) + x_2 - x_1}{x_1 y_2 - y_1 x_2}$$

Pour $p_1 = \lambda$ et $p_2 = \lambda j$, on a :

$$a = \frac{-\lambda^2 - 1}{\lambda}$$
$$b = -\sqrt{3}\lambda - \frac{\sqrt{3}}{\lambda}$$
$$r = \sqrt{\frac{\lambda^4 + 2\lambda^2 + 1}{\lambda} - 1}$$
$$c + id = \frac{\lambda^2 + 1}{2\lambda} + i\frac{\sqrt{3}\lambda^2 + \sqrt{3}}{2\lambda}$$

Nous allons calculer l'équation de la tangente passant par $x_1 + iy_1$ avec pour centre du cercle $c + id$ et de rayon r. Soit un point $M = x + iy$ sur la tangente. On a :

$(x - x_1)(x_1 - c) + (y - y_1)(y_1 - d) = 0$
$xx_1 - xc - x_1^2 + cx_1 + yy_1 - dy - y_1^2 + dy_1 = 0$
Or (x_1, y_1) est sur le cercle
$(x_1 - c)^2 + (y_1 - d)^2 = r^2$
$x_1^2 - 2x_1c + c^2 + y_1^2 - 2y_1d + d^2 - r^2 = 0$
On additionne les deux équations
$xx_1 - x_c - cx_1 + yy_1 - dy - y_1d + c^2 + d^2 - r^2 = 0$
$y = \frac{c - x_1}{y_1 - d}x + \frac{cx_1 + dy_1 - c^2 - d^2 + r^2}{y_1 d}$

Nous appliquons cette équation au point p_1 et au cercle C_1 passant par p_1 et p_2 et nous obtenons le coefficient directeur de la tangente :

$$a_{C_1} = \frac{\sqrt{3}}{3}\frac{\lambda^2 - 1}{\lambda^2 + 1}$$

Nous constatons que le coefficient directeur de la tangente au cercle C_2 passant par p_1 et p_3 est l'opposé de a_{C_1} (car les deux centres des cercles ont leur partie imaginaire opposée et leur partie réelle égale). L'angle entre 2 droites de coefficient directeur a_{C_1} et a_{C_2} apparaît dans l'équation suivante :

$$\tan(\alpha) = \frac{a_{C_2} - a_{C_1}}{1 + a_{C_2}a_{C_1}}$$

On remplace $\tan(\alpha)$ par β et a_{C_1} et a_{C_2} par leur expression en λ respective et on obtient :

$$\beta = -\sqrt{3}\frac{\lambda^4 - 1}{\lambda^4 + 4\lambda^2 + 1}$$

On remplace λ^2 par t, on a l'équation suivante :

$$(-\sqrt{3} - \beta)t^2 - 4\beta t + \sqrt{3} - \beta = 0$$

On a :

$$\Delta = 12\beta^2 + 12 > 0$$

et donc :

$$\lambda = \sqrt{\frac{4\beta - \sqrt{\Delta}}{-2(\sqrt{3} + \beta)}}$$

Avec l'équation ci-dessus, on peut donc retrouver λ à partir de α, et ainsi dessiner le triangle équilatéral d'angle α.

7.2 Étude des quotients du groupe du triangle

Rappelons que ce triangle définit les trois réflexions σ_1, σ_2 et σ_3, avec $\forall i \in \{1, 2, 3\}$, $\sigma_i^2 = 1$ et $(\sigma_1\sigma_2)^a = (\sigma_2\sigma_3)^a = (\sigma_3\sigma_1)^a = 1$. Soit K un groupe fini. Un morphisme f de T dans K est déterminé par les images k_1, k_2 et k_3 des éléments σ_1, σ_2 et σ_3 respectivement. Si f est un morphisme, il faut :

1. $f(\sigma_i^2) = f(1) = 1_K = f(\sigma_i)^2$ donc $k_i^2 = 1_K$, $\forall i \in \{1, 2, 3\}$.
2. $f((\sigma_i\sigma_j)^a) = f(1) = 1_K = (f(\sigma_i)f(\sigma_j))^a$ donc $(k_ik_j)^a = 1_K$, $\forall (i, j) \in \{(1, 2), (2, 3), (3, 1)\}$.

Les premiers groupes K auquel nous avons pensé sont les groupes symétriques S_n d'indice n. Soit $H = Ker(f)$, H est un sous-groupe distingué de T. Donc :

$$T/H = G = Im(f) \subseteq S_n$$

Les groupes G considérés seront donc des sous-groupes de S_n.

Nous allons ensuite calculer un domaine fondamental de H sur le disque de Poincaré. Comme le groupe H est infini, nous n'allons pas le calculer explicitement. Nous allons utiliser le fait que $G = Im(f)$ pour calculer tous les éléments de T dont les images par f sont envoyées une à une sur un élément différent de G. On aura ainsi un domaine de H. Pour avoir un domaine connexe, il suffit que les éléments de T calculés précédemment soient tous « collés » sur le plan hyperbolique.

Exemple 7.1. *Soit $K = S_4$, nous avons choisi :*

$$\begin{aligned} f : T &\to S_4 \\ \sigma_1 &\mapsto (34) \\ \sigma_2 &\mapsto (23) \\ \sigma_3 &\mapsto (13) \end{aligned}$$

$H = Ker(f)$. La fonction f est surjective donc $G = Im(f) = S_4$. On a $(34)^2 = (23)^2 = (13)^2 = 1$ et $(34)(23) = (243)$ est d'ordre 3 tout comme $(23)(13) = (123)$ et $(13)(34) = (134)$. Donc a sera un multiple de 3 supérieur à 3, nous avons choisi $a = 6$.

Nous prenons les trois réflexions de départ $\sigma_1, \sigma_2, \sigma_3$ et leur image dans G par f, puis nous les composons jusqu'à obtenir tous les éléments de G. Ainsi, nous avons un domaine fondamental connexe de H, dont nous allons numéroter les éléments (numérotation issue de la fonction `SymmetricGroup` de Sage) afin de voir où ils se situent[1] sur le domaine de la figure 7.4 :

0	1	2	3	4	5
Id	σ_1	σ_2	$\sigma_2\sigma_1$	$\sigma_1\sigma_2$	$\sigma_1\sigma_2\sigma_1$
6	7	8	9	10	11
$\sigma_2\sigma_3\sigma_2$	$\sigma_2\sigma_3\sigma_2\sigma_1$	$\sigma_2\sigma_3$	$\sigma_2\sigma_3\sigma_1$	$\sigma_1\sigma_2\sigma_3$	$\sigma_1\sigma_2\sigma_3\sigma_1$
12	13	14	15	16	17
$\sigma_3\sigma_2$	$\sigma_3\sigma_2\sigma_1$	σ_3	$\sigma_3\sigma_1$	$\sigma_3\sigma_1\sigma_2\sigma_1$	$\sigma_3\sigma_1\sigma_1$
18	19	20	21	22	23
$\sigma_1\sigma_3\sigma_2$	$\sigma_1\sigma_3\sigma_2\sigma_1$	$\sigma_1\sigma_3$	$\sigma_1\sigma_3\sigma_1$	$\sigma_2\sigma_1\sigma_3$	$\sigma_2\sigma_1\sigma_3\sigma_1$

On a, en rouge, les côtés qui sont des images de σ_1, en vert, les images de σ_2 et en bleu les images de σ_3. Nous regardons ensuite quels côtés du bord ne sont pas dans le domaine fondamental, c'est-à-dire qu'ils sont l'image d'un autre côté du bord du domaine par un élément de H. En effet, par un élément de H, chaque triangle du domaine est envoyé sur un triangle hors du domaine. Pour trouver les identifications, nous devons donc trouver l'unique élément de H qui envoie un côté d'un triangle au bord du domaine sur un autre côté au bord du domaine, tout en envoyant le triangle hors du domaine.

Nous allons identifier chaque triangle par le mot sur les réflexions $\sigma_1, \sigma_2, \sigma_3$ qui permettent de le dessiner à partir du triangle de base. Prenons le triangle numéroté 4, image de $\sigma_1\sigma_2$. Son côté vert sur le dessin est un côté extérieur du domaine. Nous allons chercher le mot m sur les réflexions $\sigma_1, \sigma_2, \sigma_3$ qui envoie le triangle 4, qui est au bord du domaine de H, sur un triangle hors du domaine fondamental de H, mais limitrophe. On voit que :

$$m = (\sigma_1\sigma_2)^{-1}(\sigma_2\sigma_1\sigma_2\sigma_1) = \sigma_2\sigma_1\sigma_2\sigma_1\sigma_2\sigma_1$$

On a bien $m \in H$ car

$$\begin{aligned} f(m) &= f(\sigma_2\sigma_1\sigma_2\sigma_1\sigma_2\sigma_1) \\ &= (23)(34)(23)(34)(23)(34) \\ &= Id \end{aligned}$$

Donc le côté vert du triangle 4 s'identifie au côté vert du triangle 5. On fait de même pour les autres côtés et on obtient le tableau d'identification 7.1.

Ainsi, on a le domaine de la figure 7.5, avec les côtés identifiés qui sont marqués d'une même lettre.

1. Par commodité, nous avons considéré l'action à droite de H sur T

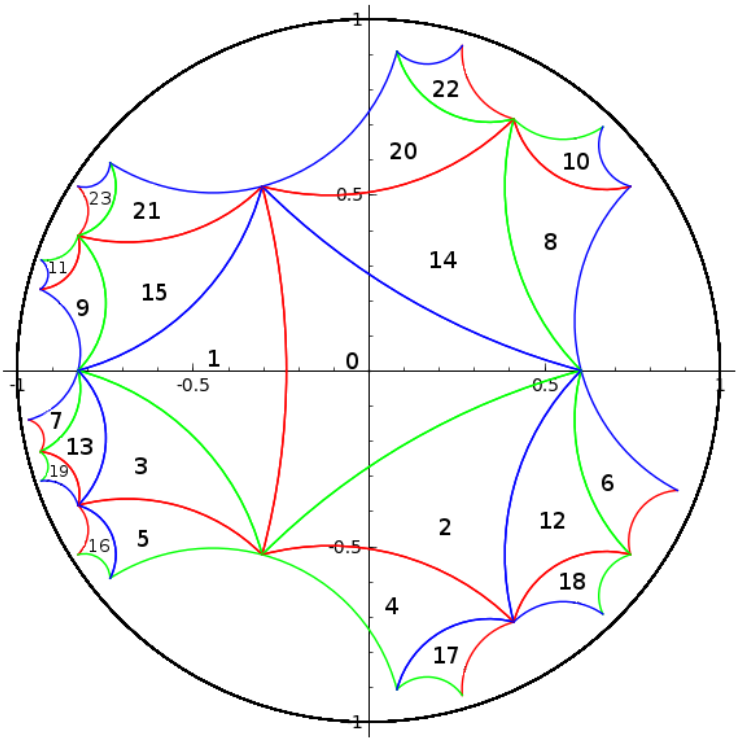


FIGURE 7.4 – Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré pour $G = S_4$

premier triangle	deuxième triangle	couleur du côté identifié	élément de H qui les identifie
4	5	vert	$\sigma_2\sigma_1\sigma_2\sigma_1\sigma_2\sigma_1$
6	7	rouge	$\sigma_1\sigma_2\sigma_3\sigma_2\sigma_1\sigma_2\sigma_3\sigma_2$
6	8	bleu	$\sigma_3\sigma_2\sigma_3\sigma_2\sigma_3\sigma_2$
7	9	bleu	$\sigma_3\sigma_2\sigma_3\sigma_2\sigma_3\sigma_2$
10	16	vert	$\sigma_2\sigma_1\sigma_2\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3$
10	11	bleu	$\sigma_3\sigma_1\sigma_2\sigma_3\sigma_1\sigma_3\sigma_2\sigma_1$
11	17	vert	$\sigma_2\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1$
16	22	rouge	$\sigma_1\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3\sigma_1\sigma_2$
17	23	rouge	$\sigma_1\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3\sigma_1\sigma_2$
18	19	vert	$\sigma_2\sigma_1\sigma_3\sigma_2\sigma_1\sigma_2\sigma_3\sigma_1$
18	23	bleu	$\sigma_3\sigma_1\sigma_3\sigma_2\sigma_1\sigma_3\sigma_1\sigma_2$
19	22	bleu	$\sigma_3\sigma_1\sigma_3\sigma_2\sigma_1\sigma_3\sigma_1\sigma_2$
20	21	bleu	$\sigma_3\sigma_1\sigma_3\sigma_1\sigma_3\sigma_1$

TABLE 7.1 – Identifications des côtés du domaine pour $G = S_4$

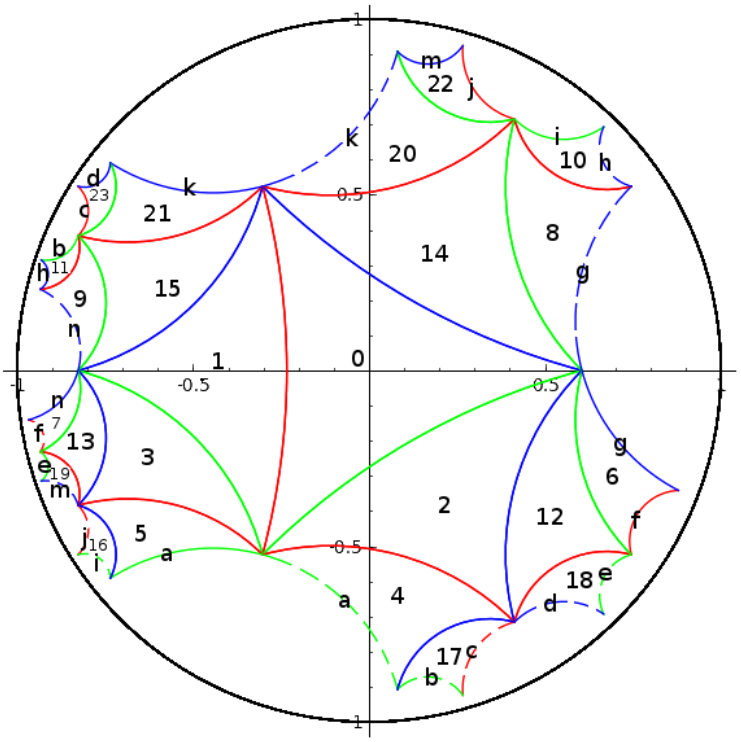


FIGURE 7.5 – Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré avec les identifications des côtés pour $G = S_4$

On recolle les côtés identifiés et on construit ainsi la surface quotient D/H (avec D le disque de Poincaré) triangulée avec 36 arêtes, 12 sommets et 24 faces. Donc la caractéristique d'Euler $\chi = S - A + F = 12 - 36 + 24 = 0$ et le genre de la surface est $g = \frac{2-\chi}{2} = 1$. On a donc un tore triangulé.

Exemple 7.2. *Nous allons également voir un deuxième exemple avec $K = S_5$ et l'application f suivante :*

$$\begin{aligned} f : T &\to S_5 \\ \sigma_1 &\mapsto (45) \\ \sigma_2 &\mapsto (34) \\ \sigma_3 &\mapsto (12)(45) \end{aligned}$$

Ici f n'est pas surjective et si on calcule $Im(f)$ on a :

$$\begin{aligned} Im(f) = \{&Id, (45), (34), (345), (354), (35), (12), \\ &(12)(45), (12)(34), (12)(345), (12)(354), (12)(35)\} \end{aligned}$$

$G = Im(f)$ est de cardinal 12 et est isomorphe à D_6 le groupe diédral d'indice 6. On a $(45)^2 = (34)^2 = ((12)(45))^2 = 1$ et $(45)(34) = (354)$ est d'ordre 3, $(34)(12)(45) = (12)(345)$ est d'ordre 6 et $(12)(45)(45) = (12)$ est d'ordre 2. Donc a sera un multiple de 2, 3 et 6, nous avons choisi $a = 6$. Comme dans le précédent exemple, on dessine

premier triangle	deuxième triangle	couleur du côté identifié	élément de H qui les identifie
4	5	vert	$\sigma_2\sigma_1\sigma_2\sigma_1\sigma_2\sigma_1$
4	8	bleu	$\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3\sigma_2$
5	9	bleu	$\sigma_3\sigma_1\sigma_2\sigma_1\sigma_3\sigma_2$
6	7	rouge	$\sigma_1\sigma_3\sigma_1\sigma_3$
8	10	rouge	$\sigma_1\sigma_2\sigma_3\sigma_1\sigma_2\sigma_3$
9	11	rouge	$\sigma_1\sigma_2\sigma_3\sigma_1\sigma_2\sigma_3$
10	11	vert	$\sigma_2\sigma_3\sigma_2\sigma_1\sigma_2\sigma_3$

TABLE 7.2 – Identifications des côtés du domaine pour $G = D_6$

le domaine fondamental de $H = Ker(f)$ en utilisant les images des générateurs de T. En effet, on a :

0	1	2	3	4	5
Id	σ_1	σ_2	$\sigma_2\sigma_1$	$\sigma_1\sigma_2$	$\sigma_1\sigma_2\sigma_1$
6	7	8	9	10	11
$\sigma_3\sigma_1$	σ_3	$\sigma_2\sigma_3\sigma_1$	$\sigma_2\sigma_3$	$\sigma_3\sigma_2$	$\sigma_3\sigma_2\sigma_1$

On obtient le domaine 7.6. Nous allons identifier les bords du domaine deux à deux.

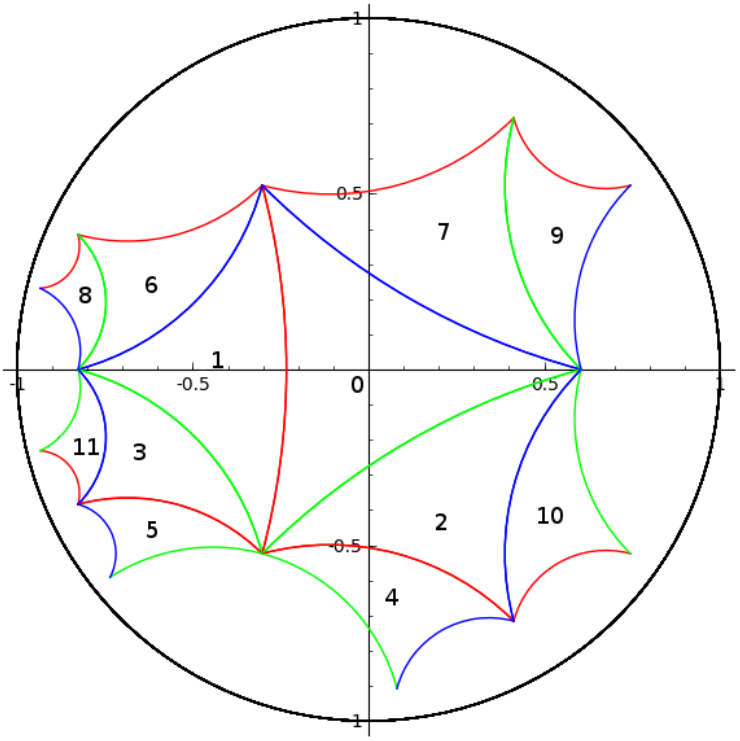


FIGURE 7.6 – Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré pour $G = D_6$

On a le tableau d'identification 7.2.

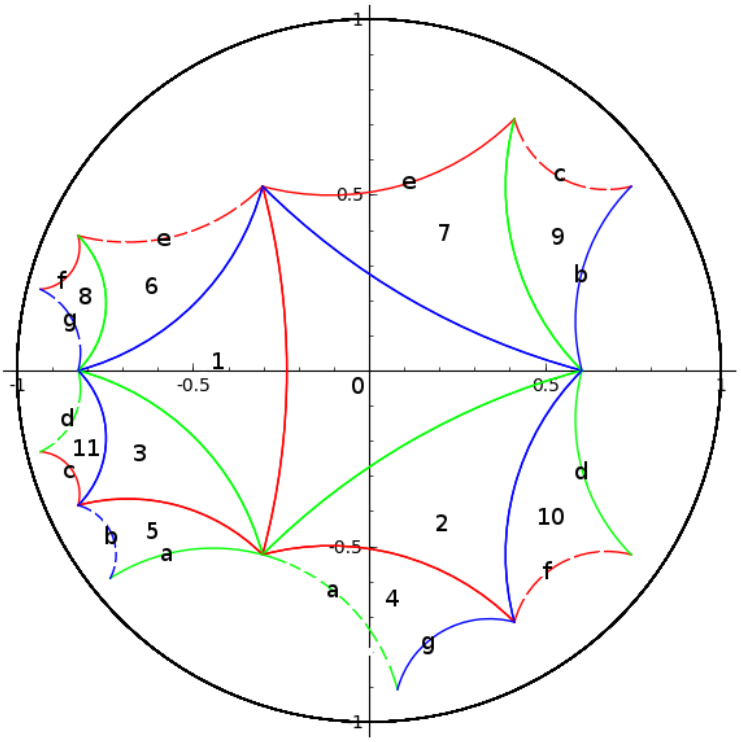


FIGURE 7.7 – Domaine fondamental de $H = Ker(f)$ sur le disque de Poincaré avec les identifications des côtés pour $G = D_6$

Ainsi, on a le domaine de la figure 7.7, avec les côtés identifiés qui sont marqués d'une même lettre.

On recolle les côtés identifiés et on construit ainsi la surface quotient D/H (avec D le disque de Poincaré) triangulée avec 18 *arêtes,* 6 *sommets et* 12 *faces. Donc la caractéristique d'Euler $\chi = S - A + F = 6 - 18 + 12 = 0$ et le genre de la surface est $g = \frac{2-\chi}{2} = 1$. On a donc un tore triangulé.*

Nous allons maintenant voir comment on calcule les intervalles d'encodage sur ces groupes.

7.3 Encodage du message

7.3.1 Intervalles d'encodage

Une géodésique va parcourir la surface associée au domaine fondamental précédemment calculé, dans lequel, à chaque triangle correspond un élément du groupe quotient T/H. Nous avons choisi de faire démarrer la géodésique au point $(0, 0)$ et ainsi avoir une droite de pente choisie. Avec cette géodésique, on parcourt un certain nombre de triangles de la surface. En effet, quand la géodésique arrive sur un côté extérieur du domaine, on utilise l'identification entre côtés extérieurs afin de la prolonger dans le domaine. Ainsi, on calcule le nombre de triangles sans répétitions afin d'avoir des intervalles d'encodage. On obtient le graphique 7.8.

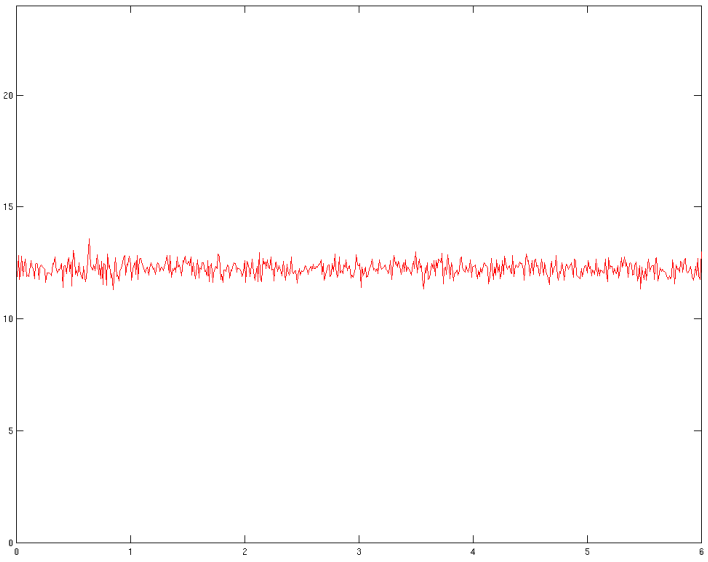


FIGURE 7.8 – Longueur moyenne des intervalles d'encodage en fonction de la pente pour $G = S_4$

On peut constater que la pente n'influe pas sur la longueur des intervalles d'encodage. On a une longueur de 13 en moyenne ce qui donne un rendement à peine supérieur à $\frac{1}{2}$ ce qui n'est pas suffisant. De plus, 13 n'est qu'une moyenne, on a encore beaucoup d'intervalles de petites longueurs. Pour augmenter la longueur de ces intervalles, nous n'allons plus compter comme fin d'intervalle, les répétitions issues de ce qu'on va appeler des auto-intersections génériques.

7.3.2 Auto-intersections génériques

Définition 7.2. *Une* ***auto-intersection générique*** *(AIG) est une répétition de triangles qui n'excède pas deux triangles identiques.*

Dans l'image de gauche de la figure 7.9, la suite de triangles traversés par la géodésique noire (dans n'importe quel sens) est $2, 1, 4$ et la suite de triangles traversés par la géodésique rose est $3, 1, 2$. Les triangles $2, 1$ se répètent mais la troisième case n'est pas la même donc on est en présence d'une auto-intersection générique. Dans la deuxième image, nous avons la même suite de 3 triangles modulo l'ordre des éléments donc l'auto-intersection n'est pas générique.

Nous allons donc, dans la suite de triangles que l'on obtient, ne plus compter comme fin d'intervalle d'encodage, les auto-intersections génériques. On arrête l'intervalle lors d'une intersection non-générique. Nous devons cependant supprimer les doublons dans les intervalles que l'on obtient. Nous calculons de nouveau les longueurs des intervalles d'encodage en fonction de la pente et nous obtenons le graphique 7.10.

On peut voir que la longueur moyenne des intervalles d'encodage a augmenté pour passer à 16 environ. On a cependant encore beaucoup de petits intervalles de longueur 8 ou 9. Nous constatons que ces petits intervalles sont différents, ce n'est pas

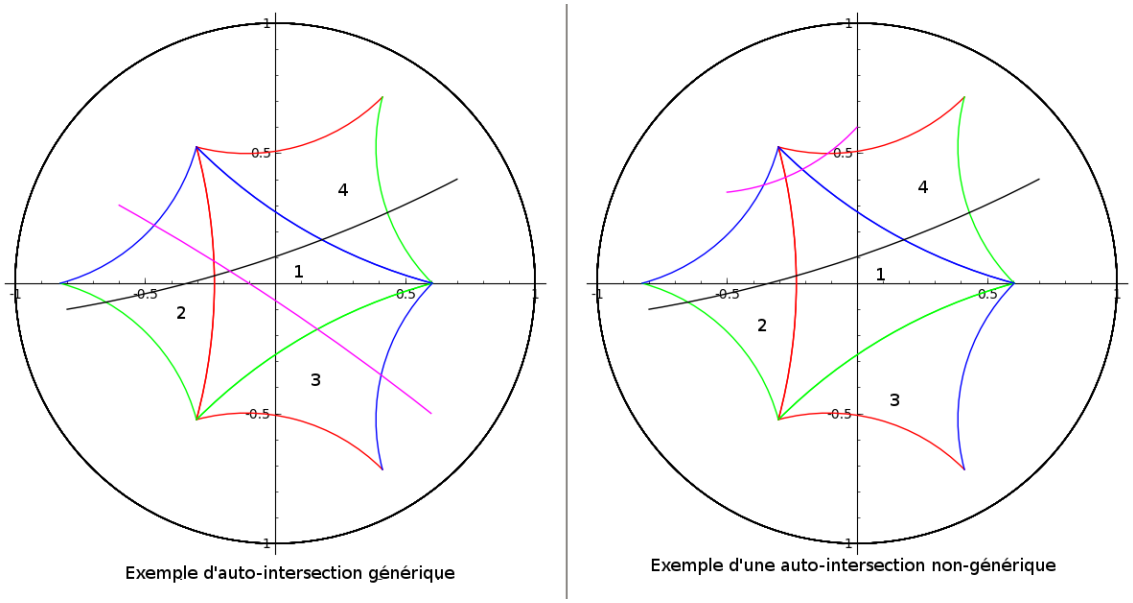


FIGURE 7.9 – Exemple et contre-exemple d'une auto-intersection générique

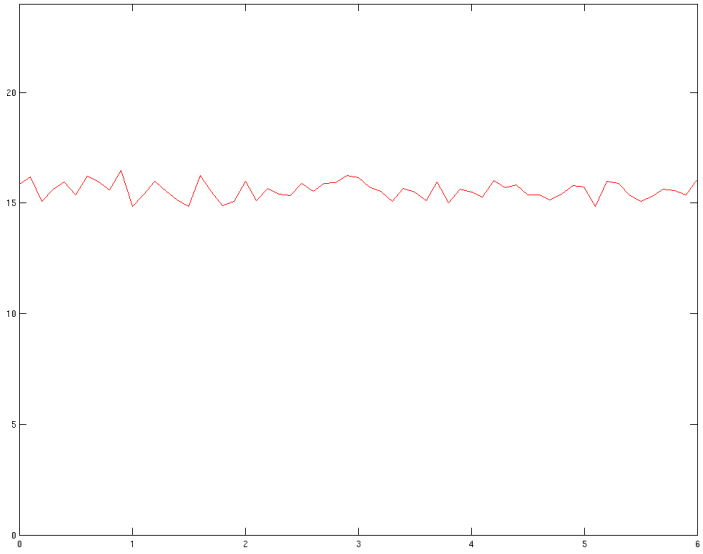


FIGURE 7.10 – Longueur moyenne des intervalles d'encodage en fonction de la pente en ne tenant pas compte des auto-intersections génériques pour $G = S_4$

la même suite d'éléments qui se répète. Pour éviter ces petits intervalles d'encodage qui diminuent le rendement du code, nous allons, comme dans le chapitre 5, choisir k une longueur d'intervalles d'encodage, ignorer les cases traversées si celles-ci sont déjà dans l'intervalle d'encodage que l'on souhaite remplir, et ainsi construire des intervalles d'encodage de longueur k.

7.3.3 Intervalles de longueur k sans répétitions

Pour avoir ces intervalles d'encodage de longueur k sans répétitions, nous allons utiliser l'algorithme 7.1.

Algorithme 7.1 Calcul d'intervalles d'encodage de longueur k sur un groupe fini G

Entrée: *dom* le domaine associé au groupe fini G, $NbInterval$ le nombre d'intervalles, a la pente, k la longueur des intervalles d'encodage
Sortie: $TabInterval$ le tableau contenant tous les intervalles

```
Initialiser un vecteur DansInterval de taille Card(G).
NumeroElt = 1
Tant que longueur(TabInterval) < NbInterval Faire
  Parcourir dom avec la droite partant de (0,0) et ayant pour pente a
  Pour chaque face traversée ft Faire
    Si DansInterval(ft) == 0 Alors
      DansInterval(ft) = NumeroElt
      NumeroElt++
      Si NumeroElt == k+1 Alors
        Initialiser un vecteur Interval de taille k.
        Pour j = 1 à Card(G) Faire
          Si DansInterval(j) ≠ 0 Alors
            Interval(DansInterval(j)) = j
          Fin Si
        Fin Pour
        Ajouter Interval au tableau TabInterval
        NumeroElt = 1
        Réinitialiser le vecteur DansInterval
      Fin Si
    Fin Si
  Fin Pour
Fin Tant que
Retourner TabInterval
```

De même que dans le chapitre 5, cette façon de calculer les intervalles d'encodage va être plus coûteuse en terme de parcours des cases. Dans le tableau 7.3, nous avons calculé le nombre de cases à parcourir pour pouvoir remplir 100 intervalles d'encodage, tout d'abord dans le cas $G = D_6$, puis pour $G = S_4$. On peut voir dans ce tableau que le nombre de cases à parcourir pour remplir ces 100 intervalles d'encodage est au maximum 2.3 fois plus grand que le nombre de cases réellement enregistrées dans ces intervalles. Le coût est encore moins important que dans le

	$G = D_6$	
k	$100k$	nombre de cases parcourues
1	100	100
2	200	200
3	300	300
4	400	400
5	500	512
6	600	620
7	700	769
8	800	931
9	900	1169
10	1000	1407
11	1100	1887

	$G = S_4$	
k	$100k$	nombre de cases parcourues
1	100	100
2	200	200
3	300	300
4	400	400
5	500	500
6	600	600
7	700	704
8	800	808
9	900	927
10	1000	1041
11	1100	1195
12	1200	1322
13	1300	1477
14	1400	1651
15	1500	1836
16	1600	2043
17	1700	2264
18	1800	2489
19	1900	2878
20	2000	3191
21	2100	3638
22	2200	4158
23	2300	5078

TABLE 7.3 – Nombre moyen de cases à parcourir pour avoir 100 intervalles de longueur k pour $G = D_6$ et $G = S_4$

chapitre 5.

Nous allons maintenant tester l'aspect chaotique du parcours, d'après la définition 3.3. Nous allons commencer par tester la sensibilité aux conditions initiales. Pour cela, nous allons donc choisir deux pentes p_1 et p_2 proches l'une de l'autre d'un certain écart ϵ et calculer $\sum \frac{\phi_i(p_1,p_2)}{l}$, avec $l = 2400$. Le résultat se trouve dans le tableau 7.4.

écarts ϵ	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
$\frac{1}{2400}\sum_{i=0}^{2399}\phi_i(p_1,p_2)$	0,0492	0,0408	0,0420	0,0462	0,0479	0,0458	0,0466

TABLE 7.4 – Calcul de $\sum_{i=0}^{2399} \frac{\phi_i(p_1,p_2)}{2400}$ en fonction des écarts entre les pentes des géodésiques

On peut voir que pour chaque ϵ, $\sum_{i=0}^{2399} \frac{\phi_i(p_1,p_2)}{2400}$ est environ égal à $\frac{1}{24} = 0,042$. Le calcul des intervalles d'encodage est donc bien sensible aux conditions initiales.

Testons maintenant la condition d'uniforme répartition des éléments dans le parcours par une géodésique. Pour cela, nous calculons l'occurrence o_g de chaque élément g dans la suite des éléments, que nous divisons par la longueur de cette suite afin de la comparer à $\frac{1}{Card(G)}$. Le résultat se trouve dans le tableau 7.5.

On peut constater que la propriété d'uniforme répartition des éléments est bien respectée. Nous avons donc des intervalles d'encodage chaotiques.

7.3.4 Problèmes de précision

Comme dans le chapitre 5, le parcours des éléments du groupe est chaotique donc sensible aux conditions initiales. Il faut donc que l'expéditeur et le destinataire aient exactement la même pente. Si celle-ci est un nombre approché d'un irrationnel, il faut qu'ils aient donc la même précision pour approcher ce nombre.

De plus, un problème peut survenir lorsque la géodésique intersecte un sommet de triangle. Si la pente est choisie aléatoirement par l'expéditeur, la probabilité est faible, mais cela peut survenir. Dans ce cas, il faut stopper l'algorithme et recommencer avec une nouvelle droite de pente différente.

7.4 Fonctions de transfert et propriétés

Nous pouvons donc maintenant définir le message u sur le groupe fini G choisi. L'opération d'encodage consiste ensuite à convoluer ce message avec une fonction de transfert. Cette dernière doit rendre le codage injectif ce que l'on peut vérifier grâce au critère du théorème 2.1. Nous avons implémenté ce critère en Sage et testé toutes les fonctions de $\mathbb{F}_2^{D_6}$ et $\mathbb{F}_2^{S_4}$. On obtient :

1. 768 fonctions sur $2^{12} = 4096$ passent le test pour $G = D_6$.
2. 3145728 fonctions sur $2^{24} = 16777216$ passent le test pour $G = S_4$.

indice de g	o_g	$\frac{o_g}{2400}$
0	103	0,0429
1	104	0,0433
2	95	0,0396
3	89	0,0371
4	109	0,0454
5	100	0,0417
6	101	0,0421
7	91	0,0379
8	114	0,0475
9	96	0,04
10	103	0,0429
11	101	0,0421
12	93	0,0388
13	86	0,0358
14	109	0,0454
15	101	0,0421
16	98	0,0408
17	106	0,0442
18	93	0,0388
19	91	0,0379
20	111	0,0463
21	107	0,0446
22	100	0,0417
23	99	0,0413

TABLE 7.5 – Calcul des occurrences des éléments g dans une suite de 2400 éléments parcourus sur le groupe S_4

Donc, dans les deux cas, 3 fonctions sur 16 rendent le codage injectif.

Nous pouvons maintenant encoder les messages (ce que l'on verra en détails dans la partie suivante) et calculer la distance minimale des codes obtenus à partir d'un intervalle d'encodage E et d'une fonction de transfert τ. Comme dans la partie 5.6, on peut se limiter aux représentants des orbites des ensembles E et aux représentants des orbites des fonctions de transfert τ par l'action à droite des groupes D_6 ou S_4. Pour D_6, le nombre d'ensembles E à tester est donc de 830 et le nombre de fonctions de transfert τ est ramené à 64. Pour S_4, le nombre de E est estimé à $\frac{2^{24}}{24} \simeq 699051$ et le nombre de τ est de 131072, ce qui donne un nombre de couples à tester bien trop grand, pour calculer en temps et matériel raisonnables. Pour le groupe S_4, on se contentera donc de calculer la distance moyenne approchée des codes, en prenant des ensembles E et des fonctions τ au hasard pour chaque longueur k.

Nous allons tout d'abord étudier la distance minimale pour le groupe D_6. On a $64 \times 830 = 53\ 120$ couples (E, τ) qui permettent de définir chacun un code. Nous les séparons selon la longueur k de E et étudions la distance minimale, ainsi que les fonctions τ qui atteignent cette distance pour chaque E. Tout d'abord, dans le tableau 7.6 se trouve la distance minimale maximale pour ces codes que l'on compare

k	d_{min} codes en bloc	d_{min} codes sur D_6
1	12	11
2	8	7
3	6	6
4	6	5
5	4	4
6	4	4
7	4	3
8	3	2
9	2	2
10	2	2
11	2	1

TABLE 7.6 – Comparaison des distances minimales des codes linéaires en bloc connus avec celles des codes convolutifs sur D_6

k	combien de E atteignent telle distance
1	un $E \rightarrow d = 11$
2	9 $E \rightarrow d = 7$
3	10 $E \rightarrow d = 6$ et 9 $E \rightarrow d = 5$
4	40 $E \rightarrow d = 5$ et 10 $E \rightarrow d = 4$
5	59 $E \rightarrow d = 4$ et 7 $E \rightarrow d = 3$
6	30 $E \rightarrow d = 4$ et 51 $E \rightarrow d = 3$ et 9 $E \rightarrow d = 2$
7	24 $E \rightarrow d = 3$ et 42 $E \rightarrow d = 2$
8	50 $E \rightarrow d = 2$
9	19 $E \rightarrow d = 2$
10	9 $E \rightarrow d = 2$
11	un $E \rightarrow d = 1$

TABLE 7.7 – Nombre d'ensembles E atteignant telle distance sur D_6

avec la distance minimale des meilleurs codes linéaires en bloc [Gra07].

On peut constater que la distance minimale maximale des codes sur D_6 est soit identique à celles des meilleurs codes linéaires en bloc, soit d'un point plus petite. Cependant, tous les ensembles E de longueur k n'atteignent pas cette distance minimale maximale. Le tableau 7.7 montre combien d'ensembles E de longueur k atteignent telle distance.

On peut voir que tous les ensembles E ne permettent pas d'atteindre la distance minimale maximale. Comme, de plus, ces ensembles E vont être choisis de manière chaotique, on ne va pas pouvoir choisir de « bons » ensembles. Seul le choix de la fonction de transfert sera important pour avoir la distance minimale la plus élevée. Pour cela, nous avons calculé, pour chaque fonction de transfert, la moyenne de la distance pour tous les ensembles E de longueur k. Nous avons obtenu les fonctions de transfert optimales pour chaque longueur k qui se trouvent en annexe C. Le tableau 7.8 nous présente quelle distance moyenne atteint chacune de ces fonctions optimales pour chaque longueur k des intervalles d'encodage.

Nous constatons que cette distance moyenne est moins bonne que la distance minimale des codes linéaires en bloc.

k	distance moyenne
2	$5,92$
3	$4,58$
4	$4,1$
5	$3,64$
6	$2,75$
7	$2,26$
8	2
9	2
10	2
11	1

TABLE 7.8 – Distance moyenne atteinte par les fonctions de transfert τ optimales pour tous les ensembles E de longueur k

Pour le groupe S_4, nous allons calculer une distance moyenne approchée pour chaque longueur k. Pour cela, nous allons choisir des pentes et des fonctions de transfert de façon aléatoire et calculer la moyenne des distances associées à ces différents codes. L'algorithme 7.2 résume ce calcul. De plus, nous allons également sauvegarder la distance maximale calculée pour chaque k, afin d'avoir une borne supérieure pour la distance minimale maximale. Tous ces résultats sont regroupés dans le tableau 7.9.

Algorithme 7.2 Calcul de la distance moyenne approchée des codes sur S_4 en fonction de k la longueur des intervalles d'encodage

Entrée: k la longueur des intervalles d'encodage
Sortie: d la distance moyenne
  Pour $nbtau = 1$ à 1000 **Faire**
    Choisir τ au hasard parmi les fonctions de transfert rendant le codage injectif
    Choisir p une pente
    Calculer $tabE$ le tableau contenant 100 intervalles d'encodage à l'aide de l'algorithme 7.1 avec dom le domaine associé au groupe S_4, 100 le nombre d'intervalles, p la pente, k la longueur des intervalles d'encodage
    Pour $i = 1$ à 100 **Faire**
      $E_i = tabE(i)$
      $H_i = MatriceParite(\tau, E_i)$
      $d_i = DistanceCode(H_i)$
      Ajouter d_i au tableau $tabd$
    Fin Pour
  Fin Pour
  Retourner $d = moyenne(tabd)$

Enfin, nous allons également calculer la distance moyenne des codes construits en utilisant la suite de triangles directement associée à la pente, ceux construits en ne prenant pas en compte une, puis deux auto-intersections génériques et ceux ne prenant aucune intersection générique en compte (tableau 7.10).

k	distance moyenne approchée sur S_4	borne inférieure à la distance minimale maximale sur S_4	distance des codes linéaires en bloc
1	12	23	24
2	10.2511	15	16
3	8.8267	13	13
4	7.4505	11	12
5	6.5160	10	12
6	5.7510	10	10
7	5.1079	8	10
8	4.6428	8	8
9	4.1463	7	8
10	3.8693	6	8
11	3.4472	6	8
12	3.2476	5	8
13	2.9713	5	6
14	2.7646	4	6
15	2.3649	4	4
16	2.1519	4	4
17	1.9551	3	4
18	1.7823	3	4
19	1.5895	2	3
20	1.3944	2	2
21	1.0974	2	2
22	1.0246	2	2
23	1	1	2

TABLE 7.9 – Comparaison entre la distance moyenne approchée des codes sur S_4, la borne inférieure de la distance minimale maximale pour les codes sur S_4 et la distance minimale des codes linéaires en bloc

type de parcours	rendement approché	distance moyenne approchée
normal	0.5078	3.3004
autorisant une AIG par intervalle d'encodage	0.5350	3.0819
autorisant deux AIG par intervalle d'encodage	0.6020	2.6518
autorisant toutes les AIG possibles par intervalle d'encodage	0.6483	2.4206

TABLE 7.10 – Distance moyenne et rendement approchés pour chaque type de parcours du domaine de S_4

On peut constater dans le tableau 7.9, que la distance moyenne des codes sur S_4 est bien plus faible que la distance minimale des meilleurs codes linéaires. De plus, nous constatons que la borne inférieure à la distance minimale des codes sur S_4 est égale à la distance minimale des codes linéaires pour $k \in \{3, 6, 8, 15, 16, 20, 21, 22\}$ et inférieure pour les autres valeurs de k.

Si nous utilisons les intervalles d'encodage de taille variable directement calculés à partir du parcours des triangles, nous obtenons un rendement entre $\frac{12}{24}$ et $\frac{13}{24}$ et la distance est meilleure que celle obtenue pour $k = 12$. Seulement, on aura des intervalles de petites longueurs où l'on peut corriger beaucoup d'erreurs et des intervalles de plus grande longueur où l'on ne peut presque pas corriger d'erreurs. Si les erreurs sont équiprobablement réparties, alors on ne pourra pas les corriger à certains endroits.

Si nous ne comptons pas comme fin d'intervalle, une auto-intersection générique par bloc, le rendement augmente tout en restant entre $\frac{12}{24}$ et $\frac{13}{24}$ et la distance diminue, on ne gagne donc pas beaucoup. Pour l'autorisation de deux AIG par bloc, le rendement est compris entre $\frac{14}{24}$ et $\frac{15}{24}$ et la distance se situe entre les distances de $k = 14$ et $k = 15$. Lorsque nous autorisons un nombre illimité d'AIG par bloc, nous obtenons un rendement compris entre $\frac{15}{24}$ et $\frac{16}{24}$ et une distance meilleure que pour $k = 15$. Utiliser de tels intervalles peut être intéressant dans le cas d'un canal de type burst, où les erreurs arrivent par paquet, tout en espérant qu'elles arrivent sur les paquets où k est très petit et où on peut corriger beaucoup d'erreurs.

7.5 Exemples de codage

L'encodage sur ces groupes se fait de la même façon que dans le chapitre 5. En effet, nous voulons encoder un message u de longueur k sur le groupe G. Pour cela, nous calculons un intervalle d'encodage E à l'aide de la pente de la géodésique de départ. Ensuite, nous calculons le message u sur le groupe comme ceci :

$$u = \sum_i u_i E(i)$$

Nous choisissons une fonction de transfert τ rendant le codage injectif. Pour $G = D_6$, on peut la choisir parmi les fonctions optimales données en annexe C. Ensuite, nous convoluons sur le groupe, le message u avec la fonction de transfert τ. Nous obtenons un mot de code $c = \sum_i c_i g_i$ avec $g_i \in G$. Les éléments $c_i \in \mathbb{F}_2$ sont les bits à envoyer sur le canal.

Exemple 7.3. *Soit $G = D_6$ comme défini précédemment. Soit $u = [1, 0, 1, 0, 0, 1, 1]$ de longueur $k = 7$. Soit $a = -\frac{7722793\sqrt{2}}{54608393}$ la pente de la géodésique de départ démarrant à $(0, 0)$ sur le domaine. Nous calculons l'intervalle d'encodage $E = [0, 2, 10, 11, 3, 1, 4]$ de longueur $k = 7$. On rappelle l'ordre des éléments suivants :*

g_0	g_1	g_2	g_3	g_4	g_5
Id	(45)	(34)	(345)	(354)	(35)
g_6	g_7	g_8	g_9	g_{10}	g_{11}
(12)	(12)(45)	(12)(34)	(12)(345)	(12)(354)	(12)(35)

Ainsi le message u sur le groupe est défini comme :

$$\begin{aligned} u &= g_0 + g_{10} + g_1 + g_4 \\ &= Id + (12)(354) + (45) + (354) \end{aligned}$$

Nous choisissons $\tau = [0,0,0,1,1,1,0,1,1,1,0,1,1] = (345) + (354) + (12) + (12)(45) + (12)(34) + (12)(354) + (12)(35)$ *parmi les fonctions optimales pour* $k = 7$*. Nous convoluons* u *et* τ *sur le groupe et nous obtenons :*

$$\begin{aligned} c &= Id + (45) + (345) + (12) + (12)(45) + (12)(34) + (12)(345) + (12)(354) \\ &= g_0 + g_1 + g_3 + g_6 + g_7 + g_8 + g_9 + g_{10} \end{aligned}$$

Ce qui nous donne le mot de code $[1,1,0,1,0,0,1,1,1,1,1,0]$.

De même pour $G = S_4$.

7.6 Décodage

Nous allons décoder de la même façon que dans le chapitre 5, par syndrome.

Exemple 7.4. *Reprenons l'exemple d'encodage précédent et supposons que le destinataire a reçu le mot :*

$$r = [1,1,0,0,0,0,1,1,1,1,1,0]$$

Le destinataire calcule la matrice de parité H *du code associé à la fonction de transfert* τ *et à l'intervalle d'encodage* E *et obtient :*

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

Il multiplie H *et* ${}^t r$ *et obtient le vecteur colonne suivant :*

$$v = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

qui correspond au syndrome d'une erreur sur le $4^{ème}$ *bit. Donc le destinataire corrige* r *et obtient* $c = [1,1,0,1,0,0,1,1,1,1,1,0]$ *qui est bien le mot de code envoyé. Pour revenir au message envoyé, le destinataire calcule l'inverse de la fonction* τ*,* $\tau^{-1} = (1,0,0,1,0,0,0,1,1,1,1,1)$ *qu'il convolue avec le mot de code comme suit :*

$$\begin{aligned} u &= c * \tau^{-1} \\ &= (Id + (45) + (345) + (12) + (12)(45) + (12)(34) + (12)(345) + (12)(354)) \\ &\times (Id + (345) + (12)(45) + (12)(34) + (12)(345) + (12)(354) + (12)(35)) \\ &= Id + (45) + (354) + (12)(354) \\ &= g_0 + g_1 + g_4 + g_{10} \end{aligned}$$

Enfin, il en déduit les bits du message u à l'aide de l'intervalle d'encodage E.

$$E = [0, 2, 10, 11, 3, 1, 4]$$
$$u = [1, 0, 1, 0, 0, 1, 1]$$

7.7 Reconnaissance de code et aspects cryptographiques

De la même façon que dans le chapitre 5, le critère du rang ne va pas permettre de retrouver les paramètres du code. En effet, les bits d'un mot de code sont dépendants entre eux, mais les dépendances sont différentes à chaque mot de code. Donc la matrice construite avec pour lignes, les mots de code, sera de rang plein.

De plus, comme dans le chapitre 5, selon le choix de la pente de la géodésique de départ, on obtient une suite d'intervalles d'encodage totalement différents les uns des autres. Cela est dû au caractère chaotique du parcours d'une géodésique sur le plan hyperbolique. Ces intervalles sont utilisés pour transférer les bits du message u sur le groupe sur lequel on va convoluer et inversement. Ces intervalles sont donc nécessaires pour l'expéditeur et le destinataire du message. Quelqu'un qui ne connaîtrait pas ces intervalles d'encodage pourrait retrouver le message u sur le groupe (à supposer qu'il connaisse τ) mais ne pourrait ensuite pas en déduire son expression binaire de départ. Les intervalles d'encodage étant calculés à partir de la pente de la géodésique, cette dernière pourrait servir de clé à un système cryptographique. En effet, toute personne ne possédant pas cette pente serait incapable de calculer les intervalles d'encodage et ainsi retrouver le message envoyé. De plus, il serait aussi incapable de calculer la matrice de parité du code et ainsi corriger les erreurs. Le protocole d'échange sécurisé des messages se passerait ainsi comme dans la partie 5.6.3 avec pour clé la pente de la géodésique.

7.8 Conclusion

Dans ce chapitre, nous nous sommes placés sur le disque de Poincaré et nous avons considéré le groupe du triangle T avec un triangle équilatéral. Nous avons défini un morphisme f de T dans un groupe symétrique S_n. Le groupe G, sur lequel la convolution est définie, est le quotient de T par $Ker(f)$. Nous avons dessiné le domaine fondamental de $Ker(f)$ sur le disque de Poincaré et nous avons obtenu une surface triangulée où chaque triangle représente un élément de G. Sur cette surface, nous avons fait parcourir une géodésique, que nous avons choisi par convention comme étant un rayon du disque commençant au point $(0, 0)$. Le parcours de cette géodésique est chaotique et nous permet de définir des intervalles d'encodage E. Le parcours simple ne nous permettant pas d'avoir des intervalles de grandes longueurs, nous avons décidé de ne plus compter les auto-intersections génériques. Cela nous a permis d'avoir des intervalles de plus grande longueur mais nous avons toujours des intervalles de petites longueurs. Pour contourner ce problème, nous avons décidé de remplir des intervalles d'encodage de longueur k choisie en ne prenant pas en compte

les triangles dans lequel la géodésique est déjà passée. Nous avons ainsi obtenu des intervalles d'encodage de longueur k chaotiques.

Ces intervalles d'encodage permettent de définir le message u sur l'algèbre du groupe G et ainsi de le convoluer avec une fonction de transfert également sur $\mathbb{F}_2[G]$ afin d'avoir le mot de code. Nous avons étudié la distance minimale sur le groupe D_6 pour chacun des couples (τ, E) et constaté que cette distance était égale ou inférieure à la distance minimale des codes linéaires connus. Nous avons calculé des codes optimaux, qui atteignent la meilleure distance possible pour une longueur k donnée. Ensuite, les ensembles E étant choisis de manière chaotique, nous avons calculé les fonctions de transfert qui ont la meilleure distance moyenne sur tous les ensembles E possibles. Pour le groupe S_4, nous avons calculé une distance moyenne approchée et constaté également que la distance minimale est plus faible que celle des codes linéaires classiques.

La correction d'erreurs de ces codes s'effectue par syndrome, puis le message envoyé est retrouvé à l'aide de la fonction de transfert τ et de l'intervalle d'encodage E. Il est donc nécessaire que le destinataire aient calculé le même intervalle d'encodage que l'expéditeur et aient, pour cela, la même précision lors du parcours de la géodésique sur le plan hyperbolique.

Ces codes sont intéressants pour leurs propriétés cryptographiques, car ils permettent de sécuriser l'échange en considérant la pente de la géodésique comme une clé symétrique. L'aspect chaotique du parcours des triangles rend difficile l'accès aux intervalles d'encodage pour un attaquant ne possédant pas la clé du système.

Ce système offre un compromis entre correction d'erreurs et sécurité.

Conclusion et perspectives

En conclusion, nous avons étudié les codes convolutifs définis sur plusieurs groupes non-commutatifs. L'objectif était d'obtenir des codes avec des propriétés cryptographiques. Tout d'abord, nous avons étudié les codes sur le groupe diédral infini, dont les performances sont identiques aux codes convolutifs classiques. Nous avons montré que la reconnaissance de code n'est pas plus difficile, cependant nous obtenons plus de codes optimaux, qui peuvent permettre de concevoir un émetteur/récepteur dédié propriétaire différent de celui des concurrents. Ces codes n'ont malheureusement pas les propriétés cryptographiques recherchées.

Nous avons donc, dans un second temps, décidé d'étudier des codes convolutifs en bloc définis sur des groupes finis. L'idée est de faire varier l'encodage sur le groupe avec le temps. Cet encodage variable chaotiquement dépend d'un état initial partagé par l'émetteur et le récepteur. Nous avons commencé par étudier le groupe $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$, dont nous avons calculé un domaine fondamental sur le tore. Le parcours des éléments s'effectue avec le système dynamique chaotique appelé « chat d'Arnold ». Nous avons ainsi créé des intervalles d'encodage chaotiques de longueur k à partir d'un point de départ sur le tore. Ce point de départ est la clé du système cryptographique, connu seulement par l'émetteur et le récepteur et garant de la sécurité. Ainsi, chaque message est encodé sur un sous-ensemble différent du groupe avant convolution. Le décodage s'effectue par syndrome. Nous avons étudié la distance minimale de ces codes et montré qu'elle est à peine plus petite que la distance des meilleurs codes linéaires en bloc.

Ces groupes étant de cardinal impair, nous avons décidé d'étudier, dans un troisième temps, des groupes de cardinal pair, comme les sous-groupes de congruence $\Gamma(N)$, $\Gamma'_0(N)$ et $\Gamma'_1(N)$. Ces groupes agissant sur le plan hyperbolique, nous avons construit un domaine fondamental sur le demi-plan de Poincaré. Nous avons parcouru ce domaine avec une géodésique et montré que les intervalles obtenus étaient de petites longueurs et qu'il y avait souvent répétition des mêmes intervalles à cause des sommets de face n'appartenant pas au domaine.

Nous avons donc décidé d'étudier d'autres groupes de cardinal pair, quotients du groupe du triangle. Le groupe du triangle étant également défini sur le plan hyperbolique, nous avons construit un domaine fondamental qui nous permette de parcourir les éléments du groupe avec une géodésique. Cette géodésique est, initialement, un diamètre du disque de Poincaré, dont la pente est la clé du système cryptographique. Nous avons calculé des intervalles d'encodage chaotiques de longueur k qui nous ont

permis de faire varier l'encodage de chaque message. Nous avons étudié en détails les groupes D_6 et S_4 et calculé les distances minimales des codes sur ces groupes en fonction de k. Cette distance est, dans le pire des cas, de quelques points plus petite que la distance minimale des meilleurs codes linéaires en bloc. Le décodage s'effectue aussi par syndrome.

Par conséquent, que ce soit sur $\mathbb{Z}/N\mathbb{Z} \rtimes Z/M\mathbb{Z}$ ou sur les quotients du groupe du triangle, l'utilisateur de tels codes fait un compromis entre correction d'erreurs et sécurité.

Les perspectives de ce travail seraient, tout d'abord, de concevoir un algorithme de décodage rapide de ces codes en bloc, basé sur les algorithmes connus de décodage des codes convolutifs comme Viterbi ou Fano. Ensuite, une cryptanalyse approfondie du système cryptographique induit par ces codes serait indispensable pour s'assurer de la sécurité. Puis, il serait intéressant d'étudier d'autres groupes, tout d'abord, d'autres quotients du groupe du triangle, puis les groupes linéaires sur les corps finis, où l'exponentielle de matrice serait la fonction de parcours des éléments. Enfin, les codes convolutifs sont souvent utilisés comme codes de base des turbo codes. Il serait intéressant d'étudier les propriétés de turbo codes construits avec des codes convolutifs sur des groupes non-commutatifs.

Annexe A

Représentations irréductibles du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

	R_1	R_2	R_3	R_4	R_5
$(0,0)$	1	1	1	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
$(0,1)$	1	β	β^2	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$
$(0,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$
$(1,0)$	1	1	1	$\begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha^2 & 0 \\ 0 & 0 & \alpha^4 \end{pmatrix}$	$\begin{pmatrix} \alpha^3 & 0 & 0 \\ 0 & \alpha^6 & 0 \\ 0 & 0 & \alpha^5 \end{pmatrix}$
$(1,1)$	1	β	β^2	$\begin{pmatrix} 0 & \alpha & 0 \\ 0 & 0 & \alpha^2 \\ \alpha^4 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \alpha^3 & 0 \\ 0 & 0 & \alpha^6 \\ \alpha^5 & 0 & 0 \end{pmatrix}$
$(1,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & \alpha \\ \alpha^2 & 0 & 0 \\ 0 & \alpha^4 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \alpha^3 \\ \alpha^6 & 0 & 0 \\ 0 & \alpha^5 & 0 \end{pmatrix}$
$(2,0)$	1	1	1	$\begin{pmatrix} \alpha^2 & 0 & 0 \\ 0 & \alpha^4 & 0 \\ 0 & 0 & \alpha \end{pmatrix}$	$\begin{pmatrix} \alpha^6 & 0 & 0 \\ 0 & \alpha^5 & 0 \\ 0 & 0 & \alpha^3 \end{pmatrix}$
$(2,1)$	1	β	β^2	$\begin{pmatrix} 0 & \alpha^2 & 0 \\ 0 & 0 & \alpha^4 \\ \alpha & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \alpha^6 & 0 \\ 0 & 0 & \alpha^5 \\ \alpha^3 & 0 & 0 \end{pmatrix}$
$(2,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & \alpha^2 \\ \alpha^4 & 0 & 0 \\ 0 & \alpha & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \alpha^6 \\ \alpha^5 & 0 & 0 \\ 0 & \alpha^3 & 0 \end{pmatrix}$

$(3,0)$	1	1	1	$\begin{pmatrix} \alpha^3 & 0 & 0 \\ 0 & \alpha^6 & 0 \\ 0 & 0 & \alpha^5 \end{pmatrix}$	$\begin{pmatrix} \alpha^2 & 0 & 0 \\ 0 & \alpha^4 & 0 \\ 0 & 0 & \alpha \end{pmatrix}$
$(3,1)$	1	β	β^2	$\begin{pmatrix} 0 & \alpha^3 & 0 \\ 0 & 0 & \alpha^6 \\ \alpha^5 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \alpha^2 & 0 \\ 0 & 0 & \alpha^4 \\ \alpha & 0 & 0 \end{pmatrix}$
$(3,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & \alpha^3 \\ \alpha^6 & 0 & 0 \\ 0 & \alpha^5 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \alpha^2 \\ \alpha^4 & 0 & 0 \\ 0 & \alpha & 0 \end{pmatrix}$
$(4,0)$	1	1	1	$\begin{pmatrix} \alpha^4 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \alpha^2 \end{pmatrix}$	$\begin{pmatrix} \alpha^5 & 0 & 0 \\ 0 & \alpha^3 & 0 \\ 0 & 0 & \alpha^6 \end{pmatrix}$
$(4,1)$	1	β	β^2	$\begin{pmatrix} 0 & \alpha^4 & 0 \\ 0 & 0 & \alpha \\ \alpha^2 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \alpha^5 & 0 \\ 0 & 0 & \alpha^3 \\ \alpha^6 & 0 & 0 \end{pmatrix}$
$(4,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & \alpha^4 \\ \alpha & 0 & 0 \\ 0 & \alpha^2 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \alpha^5 \\ \alpha^3 & 0 & 0 \\ 0 & \alpha^6 & 0 \end{pmatrix}$
$(5,0)$	1	1	1	$\begin{pmatrix} \alpha^5 & 0 & 0 \\ 0 & \alpha^3 & 0 \\ 0 & 0 & \alpha^6 \end{pmatrix}$	$\begin{pmatrix} \alpha & 0 & 0 \\ 0 & \alpha^2 & 0 \\ 0 & 0 & \alpha^4 \end{pmatrix}$
$(5,1)$	1	β	β^2	$\begin{pmatrix} 0 & \alpha^5 & 0 \\ 0 & 0 & \alpha^3 \\ \alpha^6 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \alpha & 0 \\ 0 & 0 & \alpha^2 \\ \alpha^4 & 0 & 0 \end{pmatrix}$
$(5,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & \alpha^5 \\ \alpha^3 & 0 & 0 \\ 0 & \alpha^6 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \alpha \\ \alpha^2 & 0 & 0 \\ 0 & \alpha^4 & 0 \end{pmatrix}$
$(6,0)$	1	1	1	$\begin{pmatrix} \alpha^6 & 0 & 0 \\ 0 & \alpha^5 & 0 \\ 0 & 0 & \alpha^3 \end{pmatrix}$	$\begin{pmatrix} \alpha^4 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & \alpha^2 \end{pmatrix}$
$(6,1)$	1	β	β^2	$\begin{pmatrix} 0 & \alpha^6 & 0 \\ 0 & 0 & \alpha^5 \\ \alpha^3 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \alpha^4 & 0 \\ 0 & 0 & \alpha \\ \alpha^2 & 0 & 0 \end{pmatrix}$
$(6,2)$	1	β^2	β	$\begin{pmatrix} 0 & 0 & \alpha^6 \\ \alpha^5 & 0 & 0 \\ 0 & \alpha^3 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \alpha^4 \\ \alpha & 0 & 0 \\ 0 & \alpha^2 & 0 \end{pmatrix}$

TABLE A.1 – Représentations irréductibles du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

	R_1	R_2	R_3	R_4	R_5
$(0,0)$	1	1	1	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
$(0,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$
$(0,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$
$(1,0)$	1	1	1	$\begin{pmatrix} \gamma^{9} & 0 & 0 \\ 0 & \gamma^{18} & 0 \\ 0 & 0 & \gamma^{36} \end{pmatrix}$	$\begin{pmatrix} \gamma^{27} & 0 & 0 \\ 0 & \gamma^{54} & 0 \\ 0 & 0 & \gamma^{45} \end{pmatrix}$
$(1,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & \gamma^{9} & 0 \\ 0 & 0 & \gamma^{18} \\ \gamma^{36} & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \gamma^{27} & 0 \\ 0 & 0 & \gamma^{54} \\ \gamma^{45} & 0 & 0 \end{pmatrix}$
$(1,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & \gamma^{9} \\ \gamma^{18} & 0 & 0 \\ 0 & \gamma^{36} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \gamma^{27} \\ \gamma^{54} & 0 & 0 \\ 0 & \gamma^{45} & 0 \end{pmatrix}$
$(2,0)$	1	1	1	$\begin{pmatrix} \gamma^{18} & 0 & 0 \\ 0 & \gamma^{36} & 0 \\ 0 & 0 & \gamma^{9} \end{pmatrix}$	$\begin{pmatrix} \gamma^{54} & 0 & 0 \\ 0 & \gamma^{45} & 0 \\ 0 & 0 & \gamma^{27} \end{pmatrix}$
$(2,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & \gamma^{18} & 0 \\ 0 & 0 & \gamma^{36} \\ \gamma^{9} & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \gamma^{54} & 0 \\ 0 & 0 & \gamma^{45} \\ \gamma^{27} & 0 & 0 \end{pmatrix}$
$(2,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & \gamma^{18} \\ \gamma^{36} & 0 & 0 \\ 0 & \gamma^{9} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \gamma^{54} \\ \gamma^{45} & 0 & 0 \\ 0 & \gamma^{27} & 0 \end{pmatrix}$
$(3,0)$	1	1	1	$\begin{pmatrix} \gamma^{27} & 0 & 0 \\ 0 & \gamma^{54} & 0 \\ 0 & 0 & \gamma^{45} \end{pmatrix}$	$\begin{pmatrix} \gamma^{18} & 0 & 0 \\ 0 & \gamma^{36} & 0 \\ 0 & 0 & \gamma^{9} \end{pmatrix}$
$(3,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & \gamma^{27} & 0 \\ 0 & 0 & \gamma^{54} \\ \gamma^{45} & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \gamma^{18} & 0 \\ 0 & 0 & \gamma^{36} \\ \gamma^{9} & 0 & 0 \end{pmatrix}$
$(3,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & \gamma^{27} \\ \gamma^{54} & 0 & 0 \\ 0 & \gamma^{45} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \gamma^{18} \\ \gamma^{36} & 0 & 0 \\ 0 & \gamma^{9} & 0 \end{pmatrix}$
$(4,0)$	1	1	1	$\begin{pmatrix} \gamma^{36} & 0 & 0 \\ 0 & \gamma^{9} & 0 \\ 0 & 0 & \gamma^{18} \end{pmatrix}$	$\begin{pmatrix} \gamma^{45} & 0 & 0 \\ 0 & \gamma^{27} & 0 \\ 0 & 0 & \gamma^{54} \end{pmatrix}$
$(4,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & \gamma^{36} & 0 \\ 0 & 0 & \gamma^{9} \\ \gamma^{18} & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \gamma^{45} & 0 \\ 0 & 0 & \gamma^{27} \\ \gamma^{54} & 0 & 0 \end{pmatrix}$

$(4,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & \gamma^{36} \\ \gamma^{9} & 0 & 0 \\ 0 & \gamma^{18} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \gamma^{45} \\ \gamma^{27} & 0 & 0 \\ 0 & \gamma^{54} & 0 \end{pmatrix}$
$(5,0)$	1	1	1	$\begin{pmatrix} \gamma^{45} & 0 & 0 \\ 0 & \gamma^{27} & 0 \\ 0 & 0 & \gamma^{54} \end{pmatrix}$	$\begin{pmatrix} \gamma^{9} & 0 & 0 \\ 0 & \gamma^{18} & 0 \\ 0 & 0 & \gamma^{36} \end{pmatrix}$
$(5,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & \gamma^{45} & 0 \\ 0 & 0 & \gamma^{27} \\ \gamma^{54} & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \gamma^{9} & 0 \\ 0 & 0 & \gamma^{18} \\ \gamma^{36} & 0 & 0 \end{pmatrix}$
$(5,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & \gamma^{45} \\ \gamma^{27} & 0 & 0 \\ 0 & \gamma^{54} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \gamma^{9} \\ \gamma^{18} & 0 & 0 \\ 0 & \gamma^{36} & 0 \end{pmatrix}$
$(6,0)$	1	1	1	$\begin{pmatrix} \gamma^{54} & 0 & 0 \\ 0 & \gamma^{45} & 0 \\ 0 & 0 & \gamma^{27} \end{pmatrix}$	$\begin{pmatrix} \gamma^{36} & 0 & 0 \\ 0 & \gamma^{9} & 0 \\ 0 & 0 & \gamma^{18} \end{pmatrix}$
$(6,1)$	1	γ^{21}	γ^{42}	$\begin{pmatrix} 0 & \gamma^{54} & 0 \\ 0 & 0 & \gamma^{45} \\ \gamma^{27} & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & \gamma^{36} & 0 \\ 0 & 0 & \gamma^{9} \\ \gamma^{18} & 0 & 0 \end{pmatrix}$
$(6,2)$	1	γ^{42}	γ^{21}	$\begin{pmatrix} 0 & 0 & \gamma^{54} \\ \gamma^{45} & 0 & 0 \\ 0 & \gamma^{27} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & \gamma^{36} \\ \gamma^{9} & 0 & 0 \\ 0 & \gamma^{18} & 0 \end{pmatrix}$

TABLE A.2 – Représentations irréductibles du groupe $\mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$ sur $\mathbb{F}_{64}$

Annexe B

Fonctions de transfert optimales pour $G = \mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$

Pour $k \in \{4, 6, 13, 14, 16, 17, 18, 19\}$, on obtient la même distance minimale maximale que celle des meilleurs codes linéaires en bloc. On donne quelques exemples de couples (τ, E) qui réalisent cette distance. A chaque (τ, E), le couple $(\tau.g, E.g')$, $\forall g, g' \in G$ réalise aussi cette distance.

— $k = 4 \rightarrow d_{min} = 10$

$$(\tau, E) = ((0,0,0,0,1,0,1,1,0,1,1,0,0,1,1,1,1,0,1,0,1), [0,1,3,4])$$
$$(\tau, E) = ((0,0,1,1,0,0,1,1,1,1,1,1,0,1,1,1,0,1,0,1,0), [0,1,5,6])$$
$$(\tau, E) = ((0,0,1,1,0,0,0,0,1,1,1,0,1,0,1,1,0,1,1,1,0), [0,4,9,10])$$
$$(\tau, E) = ((0,0,0,0,1,1,0,1,0,1,1,0,1,1,0,1,1,0,1,0,1), [0,4,5,6])$$
$$(\tau, E) = ((0,0,0,0,0,0,0,1,1,0,1,1,1,1,1,0,1,1,1,0,1), [5,7,8,12])$$
$$(\tau, E) = ((0,0,1,0,0,1,1,1,0,1,1,1,0,0,1,0,1,0,0,1,1), [2,6,11,12])$$

— $k = 6 \rightarrow d_{min} = 8$

$$(\tau, E) = ((0,0,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0), [0,1,3,5,6,7])$$
$$(\tau, E) = ((0,0,1,1,0,1,1,1,1,1,1,1,0,1,1,0,0,1,1,1,1), [0,1,2,4,5,9])$$
$$(\tau, E) = ((0,0,0,1,0,0,1,1,0,1,0,0,1,0,1,1,1,0,1,0,0), [4,9,10,14,15,16])$$
$$(\tau, E) = ((0,0,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,0,1), [1,2,6,9,12,15])$$
$$(\tau, E) = ((0,0,0,0,1,0,0,1,1,1,0,0,1,1,0,1,1,0,0,0,1), [5,7,10,11,12,13])$$
$$(\tau, E) = ((0,0,1,1,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0), [1,2,4,7,12,13])$$

— $k = 13 \rightarrow d_{min} = 4$

$$(\tau, E) = ((0,0,1,0,0,1,1,1,1,1,1,1,1,0,1,0,1,1,0,1,0),$$
$$[0,1,2,3,4,5,6,7,8,9,10,11,12])$$
$$(\tau, E) = ((0,1,1,0,1,1,1,0,1,0,1,1,1,0,1,1,0,1,1,1,1),$$
$$[1,3,4,5,8,9,10,11,12,13,15,16,17])$$
$$(\tau, E) = ((0,0,1,1,1,0,1,0,0,1,1,0,1,1,1,0,0,1,1,1,1),$$
$$[0,1,4,5,6,7,9,10,11,12,13,16,18])$$
$$(\tau, E) = ((0,0,1,0,0,1,0,1,1,1,1,0,0,1,0,1,0,0,1,1,1),$$
$$[1,2,3,4,6,7,10,11,12,14,15,16,18])$$
$$(\tau, E) = ((0,0,1,0,1,0,0,1,0,1,0,0,0,1,1,0,0,1,0,1,1),$$
$$[1,3,4,7,8,9,10,11,12,13,14,17,18])$$
$$(\tau, E) = ((0,0,1,0,1,0,0,0,1,1,0,0,1,0,1,1,0,1,1,0,0),$$
$$[3,4,5,9,10,11,12,13,14,15,16,17,18])$$

— $k = 14 \rightarrow d_{min} = 4$

$$(\tau, E) = ((0,0,0,0,0,1,0,1,0,0,1,1,0,0,1,0,1,0,1,1,1),$$
$$[0,1,2,3,4,5,7,8,9,10,11,12,13,15])$$
$$(\tau, E) = ((0,0,1,0,1,1,0,0,1,0,1,1,1,0,0,0,1,0,0,1,0),$$
$$[0,1,2,3,4,5,6,7,8,11,13,14,16,18])$$
$$(\tau, E) = ((0,0,1,0,1,1,0,0,1,0,1,1,1,0,0,0,1,0,0,1,0),$$
$$[1,2,3,5,7,8,9,10,11,12,14,15,16,18])$$
$$(\tau, E) = ((0,0,1,1,0,1,1,1,1,1,1,1,0,1,1,0,0,1,1,1,1),$$
$$[0,1,2,6,7,9,10,11,12,13,14,15,17,18])$$
$$(\tau, E) = ((0,0,1,0,0,1,1,0,0,0,1,0,0,1,1,0,1,1,0,1,0),$$
$$[0,1,2,5,7,9,10,11,12,13,14,16,17,18])$$
$$(\tau, E) = ((0,0,1,0,0,1,0,1,0,0,1,1,1,0,0,0,1,1,0,1,0),$$
$$[1,2,3,4,6,8,9,11,12,14,15,16,17,18])$$

— $k = 16 \rightarrow d_{min} = 3$

$$(\tau, E) = ((0,0,0,0,1,1,0,1,0,1,0,0,1,1,1,1,1,0,1,0,1),$$
$$[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])$$
$$(\tau, E) = ((0,0,0,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0),$$
$$[0,1,3,4,5,6,7,8,9,10,11,12,13,15,16,19])$$
$$(\tau, E) = ((0,0,0,0,0,1,0,1,1,1,0,0,1,0,1,1,1,0,1,1,1),$$
$$[0,1,2,3,5,6,7,8,9,10,12,13,14,15,17,18])$$
$$(\tau, E) = ((0,0,0,0,1,0,1,1,0,1,0,1,0,1,1,1,0,1,1,1,0),$$
$$[1,2,3,4,6,7,8,9,10,11,12,13,14,16,17,18])$$
$$(\tau, E) = ((0,0,0,0,0,0,0,0,0,1,0,0,1,0,1,1,1,0,0,1,1),$$
$$[0,1,2,3,4,6,7,8,9,10,12,13,14,15,18,19])$$
$$(\tau, E) = ((0,0,0,0,1,1,1,1,0,1,0,1,1,0,1,1,1,0,0,0,1),$$
$$[0,1,2,3,5,6,7,8,11,12,13,14,15,17,18,19])$$

— $k = 17 \rightarrow d_{min} = 2$

$$(\tau, E) = ((0,0,0,0,0,0,0,0,1,0,1,1,1,1,1,0,1,1,0,1,0),$$
$$[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16])$$
$$(\tau, E) = ((0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,1,1,0),$$
$$[0,1,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17])$$
$$(\tau, E) = ((0,0,0,1,0,1,1,1,1,1,0,1,1,0,1,0,1,0,1,0,0),$$
$$[0,2,3,4,5,6,7,8,9,10,12,13,14,15,16,17,18])$$
$$(\tau, E) = ((0,0,0,0,0,1,0,0,1,0,1,0,0,1,1,1,1,1,1,1,1),$$
$$[0,1,2,4,5,6,7,8,9,10,11,12,13,14,16,18,19])$$
$$(\tau, E) = ((0,1,1,0,1,1,1,0,1,1,0,1,1,1,1,1,1,0,1,0,1),$$
$$[0,1,2,3,4,5,6,7,8,9,10,12,13,14,17,18,19])$$
$$(\tau, E) = ((0,0,0,0,0,0,0,0,1,1,1,0,1,1,0,1,1,0,0,1,1),$$
$$[1,2,3,4,5,7,8,9,10,11,12,13,14,16,17,18,19])$$

— $k = 18 \to d_{min} = 2$

$$
\begin{aligned}
(\tau, E) = ((&0,0,0,0,0,0,0,0,0,1,0,0,1,0,1,1,1,0,1,1,0),\\
&[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17])\\
(\tau, E) = ((&0,0,0,1,0,1,0,1,1,0,1,0,1,0,1,0,1,1,1,1,0),\\
&[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,16,18,19])\\
(\tau, E) = ((&0,0,0,0,0,0,1,0,0,0,0,1,0,1,0,0,0,1,0,1,0),\\
&[0,1,2,3,4,5,6,7,8,10,11,12,13,14,15,16,18,19])\\
(\tau, E) = ((&0,0,1,0,1,1,1,1,0,1,0,0,1,1,0,1,1,0,1,0,0),\\
&[0,2,3,4,5,6,7,8,9,10,11,12,13,14,15,17,18,19])\\
(\tau, E) = ((&0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,1,1,0),\\
&[0,2,3,4,5,6,7,8,9,10,11,12,13,15,16,17,18,19])\\
(\tau, E) = ((&0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,1,0,1),\\
&[0,1,2,3,5,7,8,9,10,11,12,13,14,15,16,17,18,19])
\end{aligned}
$$

— $k = 19 \to d_{min} = 2$

$$
\begin{aligned}
(\tau, E) = ((&0,0,0,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1,0,1,0),\\
&[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18])\\
(\tau, E) = ((&0,0,0,0,0,0,0,0,1,0,1,1,0,0,1,1,0,1,0,1,0),\\
&[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,18,19])\\
(\tau, E) = ((&0,0,1,0,0,1,1,0,0,1,0,0,0,1,0,0,1,1,0,1,1),\\
&[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,16,17,18,19])\\
(\tau, E) = ((&0,1,1,0,1,1,1,1,0,0,1,1,1,1,0,1,1,0,1,1,1),\\
&[0,1,2,3,4,5,6,7,8,9,10,11,12,14,15,16,17,18,19])\\
(\tau, E) = ((&0,0,1,0,1,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1),\\
&[0,1,2,3,4,5,7,8,9,10,11,12,13,14,15,16,17,18,19])\\
(\tau, E) = ((&0,0,0,0,0,0,0,0,1,0,1,1,1,0,0,0,1,1,0,0,1),\\
&[0,1,2,3,4,5,6,7,8,9,10,12,13,14,15,16,17,18,19])
\end{aligned}
$$

Pour chaque longueur d'encodage k, on a les fonctions de transfert τ optimales suivantes (ainsi que tous les $\tau.g$ avec $g \in \mathbb{Z}/7\mathbb{Z} \rtimes \mathbb{Z}/3\mathbb{Z}$), c'est-à-dire qui réalisent la meilleure distance moyenne pour tous les ensembles E de longueur k :

— $k = 2 \to d_{moy} = 10,4$

$$\begin{aligned}
\tau &= (0,0,0,0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,0,0,1,1,1,0,1,1,1,1,0,0,1,1,1,0,1)\\
&= (0,0,0,0,0,1,0,1,1,1,0,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,0,1,0,1,1,1,1,0,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,0,1,0,1,0,1,1,0,0,1,1,1,1,1)\\
&= (0,0,0,0,0,1,1,1,0,1,0,1,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,0,1,1,0,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,0,1,1,0,1,1,1,1,0,1,1,1,1)\\
&= (0,0,0,0,0,1,1,1,0,1,1,1,1,0,1,0,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,0,1,1,1,1,1,1,0,0,1,1,1,1)\\
&= (0,0,0,0,0,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0)\\
&= (0,0,0,0,0,1,1,1,1,1,0,0,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,1,1,1,0,1,0,1,0,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,0)\\
&= (0,0,0,0,1,1,0,0,0,1,0,1,1,1,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,0,0,0,1,1,1,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,0,1,1,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,1,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,1,1,0,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,1,1,1,0,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,1,1,1,0,1,1,1,0,0,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,1,1,1,1,1,1,1,0,0,1)\\
&= (0,0,0,0,1,1,0,1,1,0,0,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,1,1,0,1,1,0,0,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,1,1,0,1,1,0,0,0,1,1,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,1,0,1,1,0,1,1,1,1,1,1,1,0)\\
&= (0,0,0,0,1,1,0,1,1,1,0,0,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,1,1,0,1,1,1,0,1,0,1,1,1,0,1,0,1,1)\\
&= (0,0,0,0,1,1,0,1,1,1,1,0,1,0,1,1,1,1,1,1,1)\\
&= (0,0,0,0,1,1,1,0,0,1,0,1,1,1,0,1,1,1,0,0,1)\\
&= (0,0,0,0,1,1,1,0,0,1,1,0,1,1,1,0,1,1,1,1,1)\\
&= (0,0,0,0,1,1,1,0,1,0,1,0,1,1,1,1,1,1,1,0,1)\\
&= (0,0,0,0,1,1,1,0,1,0,1,1,0,0,1,1,1,0,1,1,0)\\
&= (0,0,0,0,1,1,1,0,1,0,1,1,0,1,1,1,0,0,1,1,0)
\end{aligned}$$

$$= (0,0,0,0,1,1,1,0,1,1,0,1,0,1,1,1,0,0,1,1,0)$$
$$= (0,0,0,0,1,1,1,0,1,1,1,1,0,0,1,1,0,0,1,1,0)$$
$$= (0,0,0,0,1,1,1,0,1,1,1,1,0,1,1,0,0,0,1,1,0)$$
$$= (0,0,0,0,1,1,1,1,0,0,1,1,1,0,1,1,0,1,1,1,1)$$
$$= (0,0,0,0,1,1,1,1,0,0,1,1,1,0,1,1,1,1,1,1,0)$$
$$= (0,0,0,0,1,1,1,1,0,0,1,1,1,1,1,0,1,1,0,1,1)$$
$$= (0,0,0,0,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,0)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,0,1,0,1,1,1,0,0,1,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,0,0,1,1,0,1,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,0,1,1,0,0,1,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,0,1,1,1,0,0,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,1,0,1,0,0,1,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,1,1,1,0,0,0,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,0,1,1,0,0,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,0,0,0,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,1,0,0,0)$$
$$= (0,0,0,0,1,1,1,1,0,1,1,0,1,1,1,0,1,1,1,0,1)$$
$$= (0,0,0,0,1,1,1,1,0,1,1,1,1,0,1,1,0,1,1,1,0)$$
$$= (0,0,0,0,1,1,1,1,1,0,1,0,1,0,1,1,0,1,1,1,1)$$
$$= (0,0,0,0,1,1,1,1,1,0,1,1,1,1,0,1,0,0,1,1,1)$$
$$= (0,0,0,0,1,1,1,1,1,1,0,1,1,0,1,1,0,1,1,0,1)$$
$$= (0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,0,1,1,1,0,1)$$
$$= (0,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0)$$
$$= (0,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,0,0)$$
$$= (0,0,0,1,1,1,0,0,1,0,0,1,1,1,0,1,1,1,1,1,1)$$
$$= (0,0,0,1,1,1,0,0,1,1,0,1,1,1,0,1,1,1,1,1,0)$$
$$= (0,0,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,0,0,1)$$
$$= (0,0,0,1,1,1,0,1,1,0,1,1,1,0,0,1,1,1,1,1,0)$$
$$= (0,0,0,1,1,1,0,1,1,0,1,1,1,1,0,0,1,1,0,1,1)$$
$$= (0,0,0,1,1,1,0,1,1,0,1,1,1,1,0,1,0,1,1,1,0)$$
$$= (0,0,0,1,1,1,0,1,1,1,1,0,0,0,0,1,0,1,1,0,1)$$
$$= (0,0,0,1,1,1,0,1,1,1,1,0,0,1,0,0,0,1,1,0,1)$$
$$= (0,0,0,1,1,1,0,1,1,1,1,0,0,1,0,1,0,1,1,0,0)$$
$$= (0,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,0,1,0,1,0)$$
$$= (0,0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,0,0,0,1,1)$$
$$= (0,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1,0,0,1,1,1)$$
$$= (0,0,1,0,0,0,1,1,0,1,0,1,1,1,0,1,1,1,1,1,1)$$

$$= (0,0,1,0,0,0,1,1,0,1,1,1,1,1,1,0,0,1,1,1,1)$$
$$= (0,0,1,0,0,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0)$$
$$= (0,0,1,0,0,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,0)$$
$$= (0,0,1,0,0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,1,1)$$
$$= (0,0,1,0,0,1,0,1,0,1,1,1,1,0,1,1,0,1,1,1,1)$$
$$= (0,0,1,0,0,1,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1)$$
$$= (0,0,1,0,0,1,0,1,1,1,1,1,1,0,0,1,1,1,0,1,1)$$
$$= (0,0,1,0,0,1,1,0,0,0,1,1,1,1,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,0,1,1,0,0,1,0,1,1,1,1,1,1,1,1,0,1)$$
$$= (0,0,1,0,0,1,1,0,1,1,0,1,1,0,1,1,1,0,1,1,1)$$
$$= (0,0,1,0,0,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0,0)$$
$$= (0,0,1,0,0,1,1,1,0,0,0,1,1,1,0,1,1,1,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,0,0,0,1,1,1,1,1,1,0,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,0,0,1,0,1,1,1,1,1,0,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,0,1,1,1,1,1,0,0,1,0,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,0,1,1,1,1,1,1,0,0,0,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,0)$$
$$= (0,0,1,0,0,1,1,1,1,0,0,1,1,1,0,1,1,0,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,1,0,1,0,1,1,0,1,1,1,1,1,0)$$
$$= (0,0,1,0,0,1,1,1,1,0,1,1,1,1,1,0,1,1,1,0,0)$$
$$= (0,0,1,0,0,1,1,1,1,1,0,1,0,1,0,1,0,1,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,0)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,0,1,0,1,1,0,1,1,0,1)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,1,0,1,1,0,0,0,1,1,1,1)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1,1,1,0)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,1,0,1,1,0)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,1,1,1,0,0,1,0,1,0,1,1)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,1,1,1,0,1,0,1,1,0,1,0)$$
$$= (0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0)$$
$$= (0,0,1,0,1,0,0,1,0,1,1,1,1,1,0,1,1,1,1,1,0)$$
$$= (0,0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,1,1,0,0,1)$$
$$= (0,0,1,0,1,0,0,1,1,1,1,1,1,1,0,1,1,1,0,1,0)$$
$$= (0,0,1,0,1,0,1,0,0,0,1,1,1,1,1,1,1,1,1,0,1)$$
$$= (0,0,1,0,1,0,1,0,0,1,0,1,1,1,1,1,1,1,1,1,0)$$
$$= (0,0,1,0,1,0,1,0,1,0,1,1,1,1,1,1,1,1,1,0,0)$$
$$= (0,0,1,0,1,0,1,0,1,1,1,1,0,0,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,0,1,0,1,1,1,1,1,0,0,1,1,1,1,1,0,1)$$
$$= (0,0,1,0,1,0,1,1,0,0,0,1,1,1,0,1,1,1,1,1,1)$$

$$= (0,0,1,0,1,0,1,1,0,1,1,0,0,1,0,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,0,1,1,0,1,1,1,1,1,1,0,0,1,1,1,0)$$
$$= (0,0,1,0,1,0,1,1,0,1,1,1,1,1,1,0,1,1,1,0,0)$$
$$= (0,0,1,0,1,0,1,1,1,0,0,1,1,1,1,1,1,0,1,1,0)$$
$$= (0,0,1,0,1,0,1,1,1,0,1,1,0,1,0,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,0,1,1,1,1,0,1,1,0,1,1,1,1,0,0,1)$$
$$= (0,0,1,0,1,0,1,1,1,1,1,0,1,0,0,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,0,1,1,1,1,1,0,1,0,0,1,1,1,1,0,1)$$
$$= (0,0,1,0,1,0,1,1,1,1,1,0,1,0,1,0,1,1,1,0,1)$$
$$= (0,0,1,0,1,0,1,1,1,1,1,0,1,1,0,0,0,1,1,1,1)$$
$$= (0,0,1,0,1,0,1,1,1,1,1,0,1,1,1,0,1,1,0,1,0)$$
$$= (0,0,1,0,1,1,0,1,0,0,1,1,1,1,0,1,1,1,1,1,0)$$
$$= (0,0,1,0,1,1,0,1,1,0,0,1,1,0,0,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,0,1,1,0,0,1,1,1,1,1,1,1,1,0,0)$$
$$= (0,0,1,0,1,1,0,1,1,0,1,0,1,1,1,0,1,0,1,1,1)$$
$$= (0,0,1,0,1,1,0,1,1,1,0,1,1,0,1,1,1,1,1,0,0)$$
$$= (0,0,1,0,1,1,1,0,0,0,1,1,0,0,1,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,0,0)$$
$$= (0,0,1,0,1,1,1,1,0,0,1,0,0,1,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,0,1,0,1,1,1,1,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,0,1,0,1,1,1,1,1,1,0,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,0,1,0,0,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,0,0,0,1,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,0,0,1,0,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,0,1,0,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,0,1,1,0,1,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,0,1,1,1,0,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,0,0,1,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,0,1,0,1,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,1,0,0,1,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,1,0,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,1,1,1,0,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,0,1,1)$$
$$= (0,0,1,0,1,1,1,1,1,0,0,1,0,1,0,1,1,0,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,1,0,0,1,0,1,1,0,1,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,1,0,1,1,1,1,0,1,0,1,1,1,0)$$
$$= (0,0,1,0,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,1,1,0,0,0,1,1,0,0,1,1,1,1)$$

$$= (0,0,1,0,1,1,1,1,1,1,1,0,1,0,0,1,0,1,1,1,0)$$
$$= (0,0,1,0,1,1,1,1,1,1,1,1,0,1,1,0,1,0,0,1,0)$$
$$= (0,0,1,0,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0,0)$$
$$= (0,0,1,1,0,0,0,0,1,1,0,1,1,1,1,0,1,1,1,1,1)$$
$$= (0,0,1,1,0,0,0,0,1,1,1,1,0,1,1,1,1,1,1,0,1)$$
$$= (0,0,1,1,0,0,0,1,1,0,1,1,1,1,1,1,0,0,1,1,1)$$
$$= (0,0,1,1,0,0,0,1,1,1,0,0,1,1,1,1,1,1,0,1,1)$$
$$= (0,0,1,1,0,0,0,1,1,1,0,1,1,1,1,1,1,0,0,1,1)$$
$$= (0,0,1,1,0,0,0,1,1,1,0,1,1,1,1,1,1,0,1,1,0)$$
$$= (0,0,1,1,0,0,0,1,1,1,1,1,0,1,1,1,1,1,0,0,1)$$
$$= (0,0,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,1,0)$$
$$= (0,0,1,1,0,0,1,1,0,1,0,1,1,1,0,0,1,1,1,1,1)$$
$$= (0,0,1,1,0,0,1,1,0,1,1,1,1,0,0,1,1,1,1,1,0)$$
$$= (0,0,1,1,0,0,1,1,1,1,1,0,1,0,1,1,1,1,0,1,1,0)$$
$$= (0,0,1,1,0,1,0,0,1,0,1,1,1,1,1,0,1,0,1,1,1)$$
$$= (0,0,1,1,0,1,0,0,1,1,0,1,1,1,1,1,0,1,1,1,0)$$
$$= (0,0,1,1,0,1,0,0,1,1,1,0,1,0,1,1,1,1,1,0,1)$$
$$= (0,0,1,1,0,1,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1)$$
$$= (0,0,1,1,0,1,0,1,0,0,1,1,1,1,1,1,0,1,1,1,0)$$
$$= (0,0,1,1,0,1,0,1,0,1,1,1,1,1,1,0,0,1,1,0,1)$$
$$= (0,0,1,1,0,1,0,1,1,0,1,0,1,1,1,1,0,0,1,1,1)$$
$$= (0,0,1,1,0,1,0,1,1,0,1,1,1,1,0,1,0,0,1,1,1)$$
$$= (0,0,1,1,0,1,0,1,1,1,0,1,1,1,0,1,1,1,0,1,0)$$
$$= (0,0,1,1,0,1,1,0,1,0,0,1,1,1,1,0,1,0,1,1,1)$$
$$= (0,0,1,1,0,1,1,0,1,1,0,0,1,1,1,0,0,1,1,1,1)$$
$$= (0,0,1,1,0,1,1,0,1,1,1,1,0,0,1,1,0,1,1,1,0)$$
$$= (0,0,1,1,0,1,1,1,0,0,0,1,1,0,1,1,0,1,1,1,1)$$
$$= (0,0,1,1,0,1,1,1,0,0,1,1,1,1,0,1,0,0,1,1,1)$$
$$= (0,0,1,1,0,1,1,1,1,0,0,1,1,1,0,1,1,1,1,1,0)$$
$$= (0,0,1,1,0,1,1,1,1,0,0,0,1,1,1,1,1,0,0,0,1)$$
$$= (0,0,1,1,0,1,1,1,1,1,0,0,0,1,1,1,1,1,0,1,0)$$
$$= (0,0,1,1,0,1,1,1,1,1,0,1,1,1,0,0,0,1,1,0,1)$$
$$= (0,0,1,1,0,1,1,1,1,1,0,1,1,1,0,1,0,0,1,1,0)$$
$$= (0,0,1,1,0,1,1,1,1,1,1,0,0,1,1,1,0,0,1,1,0)$$
$$= (0,0,1,1,0,1,1,1,1,1,1,0,1,1,1,0,0,0,1,1,0)$$
$$= (0,0,1,1,1,0,0,0,1,1,1,1,1,1,1,0,1,1,1,0,0,1)$$
$$= (0,0,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,1,0)$$

$$
\begin{aligned}
&= (0,0,1,1,1,0,0,1,1,1,1,1,0,1,0,1,1,0,0,1,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,1,0,0,0,1,1,1,0,0,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,1,1,0,0,1,1,1,1,0,0,0)
\end{aligned}
$$

— $k = 3 \to d_{moy} = 8,9$

$$
\begin{aligned}
\tau &= (0,0,0,0,0,1,0,1,0,0,0,1,1,1,1,1,1,1,1,1,0)\\
&= (0,0,0,0,0,1,0,1,1,1,1,1,1,1,0,1,0,1,0,1,0)\\
&= (0,0,0,0,0,1,1,1,0,0,0,1,1,0,1,1,1,1,1,1,0)\\
&= (0,0,0,0,0,1,1,1,0,0,0,1,1,1,0,1,1,1,1,1,0)\\
&= (0,0,0,0,0,1,1,1,0,0,1,0,1,1,1,0,0,1,1,1,1)\\
&= (0,0,0,0,0,1,1,1,0,0,1,1,1,1,1,1,0,1,1,0,0)\\
&= (0,0,0,0,0,1,1,1,0,1,1,0,1,1,1,0,0,1,1,0,1)\\
&= (0,0,0,0,0,1,1,1,0,1,1,0,1,1,1,0,0,1,1,1,0)\\
&= (0,0,0,0,0,1,1,1,1,0,1,1,1,0,0,1,0,0,1,1,1)\\
&= (0,0,0,0,1,1,0,1,0,1,1,1,1,0,1,1,0,1,0,1,0)\\
&= (0,0,0,0,1,1,0,1,1,0,1,1,1,0,0,1,0,0,1,1,1)\\
&= (0,0,0,0,1,1,1,0,0,1,1,0,1,0,1,0,0,1,1,1,1)\\
&= (0,0,0,0,1,1,1,0,0,1,1,1,0,1,1,0,1,1,1,0,0)\\
&= (0,0,0,0,1,1,1,1,0,0,1,0,0,0,1,1,1,1,1,0,1)\\
&= (0,0,0,0,1,1,1,1,0,0,1,1,1,0,0,1,0,0,1,1,1)\\
&= (0,0,0,0,1,1,1,1,1,0,1,0,1,0,0,1,0,1,1,1,0)\\
&= (0,0,0,0,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,0,0)\\
&= (0,0,0,0,1,1,1,1,1,1,1,0,1,0,0,0,1,1,1,0,0)\\
&= (0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,1,1,0,0)\\
&= (0,0,0,1,1,1,0,0,1,0,0,1,1,1,0,1,0,1,1,1,0)\\
&= (0,0,0,1,1,1,0,0,1,0,0,1,1,1,0,1,1,0,1,1,0)\\
&= (0,0,0,1,1,1,0,0,1,0,0,1,1,1,0,1,1,1,0,1,0)\\
&= (0,0,0,1,1,1,0,0,1,1,0,1,1,1,0,1,0,0,0,1,1)\\
&= (0,0,0,1,1,1,0,0,1,1,1,1,0,1,0,1,1,0,0,0,1)\\
&= (0,0,1,0,0,0,0,1,0,1,0,1,1,1,0,1,1,1,0,1,1)\\
&= (0,0,1,0,0,0,1,0,1,0,0,1,1,1,1,1,1,1,0,1,1,0)\\
&= (0,0,1,0,0,0,1,1,0,0,0,1,1,1,1,1,1,1,0,1,1,0)\\
&= (0,0,1,0,0,0,1,1,0,1,1,1,1,0,1,0,0,1,1,1,0)
\end{aligned}
$$

$$
\begin{aligned}
&= (0,0,1,0,0,0,1,1,0,1,1,1,1,1,0,0,0,1,1,1,0)\\
&= (0,0,1,0,0,0,1,1,0,1,1,1,1,1,1,0,0,1,0,1,0)\\
&= (0,0,1,0,0,0,1,1,1,1,0,0,1,0,0,0,1,1,1,1,1)\\
&= (0,0,1,0,0,1,0,1,0,0,1,1,1,1,0,1,0,0,1,1,1)\\
&= (0,0,1,0,0,1,1,1,0,1,0,1,1,1,0,0,0,0,1,1,1)\\
&= (0,0,1,0,0,1,1,1,0,1,1,0,1,1,0,0,0,0,1,1,1)\\
&= (0,0,1,0,0,1,1,1,1,0,0,0,0,1,0,1,1,1,1,1,0)\\
&= (0,0,1,0,0,1,1,1,1,0,0,0,1,1,0,1,0,1,1,1,0)\\
&= (0,0,1,0,0,1,1,1,1,0,0,0,1,1,0,1,1,0,1,1,0)\\
&= (0,0,1,0,1,0,1,0,0,1,1,1,1,0,1,0,1,1,1,0,0)\\
&= (0,0,1,0,1,0,1,1,0,0,0,1,1,0,0,1,1,1,1,0,1)\\
&= (0,0,1,0,1,0,1,1,0,1,1,1,0,0,1,0,1,1,1,0,0)\\
&= (0,0,1,0,1,1,1,0,0,0,0,0,1,1,1,1,0,0,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,0,1,0,1,0,0,0,0,1,1,1,0)
\end{aligned}
$$

— $k = 4 \rightarrow d_{moy} = 7,7895$

$$
\begin{aligned}
\tau &= (0,0,0,0,0,0,0,0,1,1,0,1,1,1,1,1,0,1,0,1,0)\\
&= (0,0,0,0,0,0,0,0,1,1,1,0,1,1,1,1,1,0,1,0,0)\\
&= (0,0,0,0,0,1,0,0,1,1,1,1,0,1,0,1,1,0,0,1,0)\\
&= (0,0,0,0,0,1,0,1,0,0,1,0,0,0,1,1,1,1,1,1,0)\\
&= (0,0,0,0,0,1,0,1,0,1,1,0,1,1,0,1,0,0,1,0,1)\\
&= (0,0,0,0,0,1,0,1,1,1,1,0,1,0,0,0,1,1,0,1,0)\\
&= (0,0,0,0,0,1,1,0,0,0,1,1,1,0,0,0,0,1,1,1,1)\\
&= (0,0,0,0,0,1,1,0,1,0,1,0,0,1,1,1,0,1,1,0,0)\\
&= (0,0,0,0,0,1,1,0,1,1,1,1,1,1,0,1,0,1,0,0,0)\\
&= (0,0,0,0,0,1,1,1,0,1,1,1,1,1,0,1,0,0,0,0,0)\\
&= (0,0,0,0,0,1,1,1,1,0,1,0,0,0,0,1,0,1,1,0,1)\\
&= (0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,0,1,1,0)\\
&= (0,0,0,0,1,1,0,0,1,1,0,1,1,0,1,1,0,0,0,1,0)\\
&= (0,0,0,0,1,1,0,1,0,1,0,0,0,1,1,0,0,0,1,1,1)\\
&= (0,0,0,0,1,1,0,1,1,0,0,0,0,1,0,1,1,1,1,0,0)\\
&= (0,0,0,0,1,1,0,1,1,0,0,0,1,0,0,1,1,1,0,1,0)\\
&= (0,0,0,0,1,1,1,0,0,0,1,0,0,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0,0,1,1,1)\\
&= (0,0,0,0,1,1,1,1,0,0,0,1,0,1,0,1,0,0,0,1,1)\\
&= (0,0,0,1,1,1,0,0,0,0,1,1,0,1,0,1,0,0,0,1,1)\\
&= (0,0,0,1,1,1,0,0,0,0,1,1,1,0,0,0,1,0,0,1,1)
\end{aligned}
$$

— $k = 5 \to d_{moy} = 7,0268$

$$\begin{aligned}
\tau &= (0,0,1,0,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1)\\
&= (0,0,1,0,0,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,0,1)\\
&= (0,0,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,0,1)\\
&= (0,0,1,0,1,1,1,1,1,1,1,1,1,0,1,0,0,1,1,1,1)\\
&= (0,0,1,1,0,1,0,1,1,0,0,1,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,1,0,1,1,1,1,1,0,1,0,1,1,1,1,1,0,1,1)\\
&= (0,0,1,1,1,1,0,0,1,1,0,1,1,1,1,1,1,1,0,1,1)\\
&= (0,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1,1,1,0,0,1)\\
&= (0,0,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,0,0,1)\\
&= (0,0,1,1,1,1,1,1,1,1,1,1,0,0,1,0,1,1,1,0,1)\\
&= (0,1,1,0,0,1,1,1,1,0,0,1,1,0,1,1,1,1,1,1,1)\\
&= (0,1,1,0,0,1,1,1,1,1,1,1,1,1,1,0,0,1,1,0,1)
\end{aligned}$$

— $k = 6 \to d_{moy} = 6,0517$

$$\begin{aligned}
\tau &= (0,0,1,0,0,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1)\\
&= (0,0,1,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0)\\
&= (0,0,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0)\\
&= (0,0,1,0,1,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,0,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1)\\
&= (0,0,1,0,1,1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,0,1)\\
&= (0,0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0,0,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,0,0)\\
&= (0,0,1,0,1,1,1,1,1,1,1,1,0,0,1,0,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,0)\\
&= (0,0,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1)\\
&= (0,0,1,1,0,0,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1)\\
&= (0,0,1,1,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1)\\
&= (0,0,1,1,0,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,1)
\end{aligned}$$

$$
\begin{aligned}
&= (0,0,1,1,0,1,0,0,1,1,1,1,1,1,0,1,1,1,1,1,1)\\
&= (0,0,1,1,0,1,0,0,1,1,1,1,1,1,1,1,1,0,1,1,1)\\
&= (0,0,1,1,0,1,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1)\\
&= (0,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0)\\
&= (0,0,1,1,0,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,1)\\
&= (0,0,1,1,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,0)\\
&= (0,0,1,1,1,0,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1)\\
&= (0,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1)\\
&= (0,0,1,1,1,0,1,0,0,1,1,0,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,0,1,0,1,1,1,1,0,0,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,0,1,1,1,1,0,0,1,1,1,1,1,1,0,1,1,1)\\
&= (0,0,1,1,1,0,1,1,1,1,0,1,1,1,0,1,1,1,1,1,0)\\
&= (0,0,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,0,0,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1,0)\\
&= (0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,1,1,0,1,1,1)\\
&= (0,0,1,1,1,1,0,1,0,1,1,1,1,1,1,0,1,1,1,1,0)\\
&= (0,0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0,0)\\
&= (0,0,1,1,1,1,1,0,0,0,1,1,1,0,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,1,1,0,1,1,1,0,0,0,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,1,1,0,1,1,1,1,0,1,0,1,0,1,1,1,1)\\
&= (0,0,1,1,1,1,1,1,1,0,1,1,0,1,1,1,1,1,1,1,0,0)\\
&= (0,0,1,1,1,1,1,1,1,0,1,1,1,1,0,0,1,1,1,1,0)\\
&= (0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,0,1,1,0,1)\\
&= (0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,0,1,1,1,1)\\
&= (0,0,1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,0,0,1,1,1)\\
&= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,0,1,1,0)
\end{aligned}
$$

— $k = 7 \to d_{moy} = 5,3374$

$$
\begin{aligned}
\tau &= (0,0,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1)\\
&= (0,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0)\\
&= (0,0,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1)
\end{aligned}
$$

$$= (0,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1)$$
$$= (0,0,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0)$$
$$= (0,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,0,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,0)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1)$$
$$= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1)$$
$$= (0,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1)$$
$$= (0,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0)$$
$$= (0,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1)$$
$$= (0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0)$$
$$= (0,1,1,1,0,0,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1)$$
$$= (0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0)$$
$$= (0,1,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1)$$
$$= (0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1)$$
$$= (0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1)$$
$$= (0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0)$$
$$= (0,1,1,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1)$$
$$= (0,1,1,1,1,0,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1)$$

— $k = 8 \to d_{moy} = 4,8088$

$$\begin{aligned}
\tau &= (0,0,1,0,1,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1)\\
&= (0,0,1,0,1,1,0,1,1,0,1,1,1,1,0,1,1,1,1,1,1)\\
&= (0,0,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0,1,0)\\
&= (0,0,1,0,1,1,1,1,0,1,1,1,0,1,1,0,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,0,1,0,0,1,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,0)\\
&= (0,0,1,1,0,0,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1)\\
&= (0,0,1,1,0,1,1,1,0,1,1,1,1,1,1,1,0,1,1,0,1)\\
&= (0,0,1,1,0,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1)\\
&= (0,0,1,1,0,1,1,1,1,1,1,1,1,0,0,1,0,1,1,1,1)\\
&= (0,0,1,1,1,0,0,1,1,1,1,1,0,1,1,1,1,1,0,1,1)\\
&= (0,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,0,1,1,0)\\
&= (0,0,1,1,1,0,1,1,1,1,0,1,1,0,1,1,1,1,1,0,1)\\
&= (0,0,1,1,1,0,1,1,1,1,1,0,1,1,0,1,1,1,1,1,0)\\
&= (0,0,1,1,1,1,0,1,0,0,1,1,1,1,1,0,1,1,1,1,1)\\
&= (0,0,1,1,1,1,0,1,1,1,1,1,1,0,1,1,0,1,0,1,1,1)\\
&= (0,0,1,1,1,1,1,1,0,0,1,1,1,1,0,1,1,1,1,1,0,1)\\
&= (0,0,1,1,1,1,1,1,0,1,1,0,0,1,1,1,1,1,1,1,0,1)\\
&= (0,0,1,1,1,1,1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,1)\\
&= (0,0,1,1,1,1,1,1,1,1,1,0,1,1,0,1,0,1,1,1,0,1,1)\\
&= (0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,0,1,1,0,0)
\end{aligned}$$

— $k = 9 \to d_{moy} = 4,3097$

$$\begin{aligned}
\tau &= (0,0,1,0,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0)\\
&= (0,0,1,1,0,1,1,1,1,1,1,1,1,1,0,0,1,0,1,1,1,1)
\end{aligned}$$

— $k = 10 \to d_{moy} = 3,9243$

$$\tau = (0,0,1,0,1,1,1,1,0,0,0,1,0,1,0,1,1,1,0,1,0)$$

— $k = 11 \to d_{moy} = 3,6456$

$$\tau = (0,0,0,0,0,0,0,1,1,1,1,1,0,0,1,0,1,0,1,1,0)$$

— $k = 12 \to d_{moy} = 3,1897$

$$\begin{aligned}
\tau &= (0,0,0,0,0,1,0,0,0,0,1,0,0,1,1,1,1,0,1,1,1)\\
&= (0,0,0,0,0,1,0,1,0,0,1,1,0,1,0,1,1,1,1,0,0)
\end{aligned}$$

— $k = 13 \to d_{moy} = 2,8278$

$$\begin{aligned}
\tau &= (0,0,0,0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,0,0,1,1,1,0,1,1,1,1,0,0,1,1,1,0,1)\\
&= (0,0,0,0,0,1,0,1,1,1,1,0,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,0,1,1,1,1,1,1,0,1,0,1,1,0,0,0,1,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,0,1,0,1,1,1,0,1,1,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,0,1,1,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,0,1,0,1,0,1,1,1,1,1,1,0,0,0,1)\\
&= (0,0,0,0,1,1,1,0,1,0,1,1,0,1,1,1,0,0,1,1,0)\\
&= (0,0,0,0,1,1,1,0,1,1,1,1,0,0,1,1,0,0,1,1,0)\\
&= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,0,1,1,1,0,0,1)\\
&= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,1,1,1,0,0,0,1)
\end{aligned}$$

— $k = 14 \to d_{moy} = 2,5845$

$$\begin{aligned}
\tau &= (0,0,0,0,0,1,1,1,0,1,1,0,1,0,1,1,1,0,0,1,1)\\
&= (0,0,0,0,1,1,1,1,0,1,0,1,1,1,0,1,1,0,0,0,1)
\end{aligned}$$

— $k = 15 \to d_{moy} = 2,1840$

$$\begin{aligned}
\tau &= (0,0,0,0,1,1,0,0,0,1,0,1,1,1,1,1,1,0,1,0,1)\\
&= (0,0,0,0,1,1,1,1,0,1,0,1,1,0,0,1,1,1,0,0,1)
\end{aligned}$$

— $k = 16 \to d_{moy} = 2$ et $k = 17 \to d_{moy} = 2$ et $k = 18 \to d_{moy} = 1,9365$

$$\begin{aligned}
\tau &= (0,0,0,0,0,1,0,1,0,1,1,1,1,1,1,1,1,0,1,1,1)\\
&= (0,0,0,0,0,1,1,1,0,1,1,1,1,1,0,1,1,1,0,1,1)\\
&= (0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,1)\\
&= (0,0,0,0,1,1,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1)\\
&= (0,0,0,0,1,1,1,0,0,1,1,1,0,1,1,1,0,1,1,1,1)\\
&= (0,0,0,0,1,1,1,1,1,1,0,0,1,1,1,1,1,1,0,0,1)\\
&= (0,0,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,0,1)\\
&= (0,0,0,1,1,1,0,0,1,1,1,0,0,1,0,1,1,1,1,1,1)\\
&= (0,0,0,1,1,1,0,1,1,1,1,0,1,1,1,0,0,1,1,1,0)\\
&= (0,0,0,1,1,1,0,1,1,1,1,1,1,1,1,0,0,1,1,0,0)\\
&= (0,0,0,1,1,1,1,1,1,0,0,1,0,1,1,1,0,0,1,1,1)\\
&= (0,0,0,1,1,1,1,1,1,0,0,1,0,1,1,1,1,0,1,1,0)\\
&= (0,0,1,0,0,0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,0)\\
&= (0,0,1,0,0,0,1,1,1,1,1,0,1,1,1,1,1,1,0,1,0)\\
&= (0,0,1,0,0,1,0,1,0,1,1,1,1,0,1,1,1,0,1,1,1)\\
&= (0,0,1,0,0,1,1,1,1,1,1,1,0,1,0,1,1,1,1,0,1,0)\\
&= (0,0,1,0,1,0,0,1,1,1,0,1,1,1,1,1,1,0,1,0,1)\\
&= (0,0,1,0,1,0,0,1,1,1,1,1,0,1,1,1,1,1,0,1,0,1)\\
&= (0,0,1,0,1,0,1,0,1,1,1,1,1,0,0,1,1,1,1,1,1,0)\\
&= (0,0,1,0,1,0,1,1,0,1,0,0,0,1,1,1,1,1,1,1,1)\\
&= (0,0,1,0,1,0,1,1,0,1,0,0,1,1,1,1,0,1,1,1,1)\\
&= (0,0,1,0,1,0,1,1,1,1,0,0,0,1,1,1,0,1,1,1,1)\\
&= (0,0,1,0,1,0,1,1,1,1,0,1,1,0,1,1,1,0,0,1,1)\\
&= (0,0,1,0,1,0,1,1,1,1,0,1,1,1,0,1,1,1,0,0,1)\\
&= (0,0,1,0,1,0,1,1,1,1,1,1,1,1,0,1,1,1,0,0,0)\\
&= (0,0,1,0,1,1,0,1,0,1,0,1,1,1,1,1,1,1,1,0,0)\\
&= (0,0,1,0,1,1,0,1,0,1,1,1,1,1,0,1,1,1,1,0,0)\\
&= (0,0,1,0,1,1,0,1,1,1,0,0,1,0,1,1,1,1,1,1,0)\\
&= (0,0,1,0,1,1,1,0,1,1,1,0,0,1,1,1,0,0,1,1,1)\\
&= (0,0,1,0,1,1,1,1,1,1,0,1,0,1,0,1,0,1,1,1,0)\\
&= (0,0,1,1,0,0,1,0,1,0,1,1,1,1,0,1,1,0,1,1,1)\\
&= (0,0,1,1,0,0,1,1,0,0,1,1,1,1,1,0,1,1,0,1,1)\\
&= (0,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,1,0,1,1,1)\\
&= (0,0,1,1,0,0,1,1,1,1,1,0,1,1,1,0,1,0,0,1,1)\\
&= (0,0,1,1,0,0,1,1,1,1,1,1,1,0,1,0,1,0,0,1,1)\\
&= (0,0,1,1,0,1,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0)\\
&= (0,0,1,1,0,1,1,0,1,0,1,1,1,1,0,0,1,0,1,1,1)
\end{aligned}$$

$$= (0,0,1,1,0,1,1,1,0,1,1,1,1,0,0,0,1,1,1,1,0)$$
$$= (0,0,1,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,1,1,1)$$
$$= (0,0,1,1,1,1,1,0,0,0,1,1,1,1,1,1,0,1,0,1,1,0)$$
$$= (0,0,1,1,1,1,1,0,1,0,1,0,1,1,1,0,1,1,1,1,0,0)$$
$$= (0,0,1,1,1,1,1,1,0,0,0,1,0,1,1,1,0,1,1,1,1,0)$$

— $k = 19 \to d_{moy} = 1,44$

$$\tau = (0,0,0,0,1,1,1,0,0,1,1,1,0,1,1,1,0,1,1,1,1)$$
$$= (0,0,0,1,1,1,0,0,1,1,1,0,0,1,0,1,1,1,1,1,1)$$
$$= (0,0,0,1,1,1,0,1,1,1,1,0,1,1,1,0,0,1,1,1,0)$$
$$= (0,0,0,1,1,1,1,1,1,0,0,1,0,1,1,1,0,0,1,1,1)$$
$$= (0,0,1,0,1,0,1,0,1,1,1,1,0,0,1,1,1,1,1,1,0)$$
$$= (0,0,1,0,1,0,1,1,1,1,0,0,0,1,1,1,0,1,1,1,1)$$
$$= (0,0,1,0,1,1,1,0,1,1,1,0,0,1,1,1,0,0,1,1,1)$$
$$= (0,0,1,0,1,1,1,1,1,1,0,1,0,1,0,1,0,1,1,1,0)$$
$$= (0,0,1,1,0,0,1,1,1,1,1,0,1,0,1,0,1,0,1,1,1)$$
$$= (0,0,1,1,0,1,1,1,0,1,1,1,1,0,0,0,1,1,1,1,0)$$
$$= (0,0,1,1,1,0,0,1,1,1,1,1,0,1,0,1,0,0,1,1,1)$$
$$= (0,0,1,1,1,1,0,0,0,1,1,1,1,1,1,0,1,0,1,1,0)$$
$$= (0,0,1,1,1,1,0,1,0,1,0,1,1,1,0,1,1,1,1,0,0)$$
$$= (0,0,1,1,1,1,1,0,0,0,1,0,1,1,1,0,1,1,1,1,0)$$

— $k = 20 \to d_{moy} = 1$, toutes les fonctions τ injectives

Annexe C

Fonctions de transfert optimales pour $G = D_6$

Pour $k \in \{3, 5, 6, 9, 10\}$, on obtient la même distance minimale maximale que celle des meilleurs codes linéaires en bloc. On donne ici quelques exemples de couples (τ, E) qui réalisent cette distance. A chaque (τ, E), le couple $(\tau.g, E.g')$, $\forall g, g' \in G$ réalise aussi cette distance.

— $k = 3 \rightarrow d_{min} = 6$

$$(\tau, E) = ((0,0,1,1,0,0,1,1,0,1,1,1), [0,1,2])$$
$$(\tau, E) = ((0,0,0,0,1,1,1,1,1,1,0,1), [0,1,3])$$
$$(\tau, E) = ((0,0,0,1,0,1,1,1,1,0,1,1), [0,2,4])$$
$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,0,0,1), [1,2,6])$$
$$(\tau, E) = ((0,0,0,1,1,0,1,1,1,1,0,1), [1,3,6])$$
$$(\tau, E) = ((0,0,0,1,1,0,1,1,1,0,1,1), [1,4,6])$$

— $k = 5 \rightarrow d_{min} = 4$

$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,0,0,1), [0,1,2,3,4])$$
$$(\tau, E) = ((0,1,0,0,1,1,0,1,1,0,1,1), [1,2,3,4,6])$$
$$(\tau, E) = ((0,0,0,0,0,1,1,0,1,1,1,0), [0,2,3,5,6])$$
$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,0,0,1), [0,2,4,6,7])$$
$$(\tau, E) = ((0,0,0,1,0,1,1,1,1,0,1,1), [1,2,4,6,8])$$
$$(\tau, E) = ((0,1,1,0,1,1,1,1,1,0,1,1), [0,4,5,7,9])$$

— $k = 6 \rightarrow d_{min} = 4$

$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,0,0,1), [0,1,2,3,4,5])$$
$$(\tau, E) = ((0,0,0,0,0,1,1,0,1,1,1,0), [0,1,3,4,5,6])$$
$$(\tau, E) = ((0,0,0,0,0,1,1,1,0,1,1,0), [0,1,3,5,6,7])$$
$$(\tau, E) = ((0,1,1,0,1,1,1,1,1,0,1,1), [2,3,4,5,6,7])$$
$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,0,0,1), [0,1,2,3,6,8])$$
$$(\tau, E) = ((0,1,0,0,1,1,0,1,1,0,1,1), [1,2,5,6,9,10])$$

— $k = 9 \rightarrow d_{min} = 2$

$$(\tau, E) = ((0,0,0,0,1,1,0,0,0,1,1,1), [0,1,2,3,4,5,6,7,8])$$
$$(\tau, E) = ((0,0,1,1,0,0,1,1,0,1,1,1), [0,1,2,3,4,6,7,9,10])$$
$$(\tau, E) = ((0,0,1,1,0,0,1,1,0,1,1,1), [0,1,3,4,5,6,7,9,10])$$
$$(\tau, E) = ((0,1,1,1,1,1,1,1,1,1,1,1), [1,2,3,4,5,6,7,9,10])$$
$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,0,0,1), [0,1,2,3,5,7,8,9,10])$$
$$(\tau, E) = ((0,1,1,0,1,1,1,1,1,0,1,1), [0,1,2,4,5,7,8,9,10])$$

— $k = 10 \rightarrow d_{min} = 2$

$$(\tau, E) = ((0,0,0,1,1,1,0,0,1,1,1,1), [0,1,2,3,4,5,6,7,8,9])$$
$$(\tau, E) = ((0,1,0,1,1,1,0,1,1,1,1,1), [0,1,2,3,4,5,6,7,9,10])$$
$$(\tau, E) = ((0,0,1,1,1,1,0,1,1,1,1,1), [0,1,3,4,5,6,7,8,9,10])$$
$$(\tau, E) = ((0,0,0,0,0,1,1,1,1,1,1,1), [0,1,2,4,5,6,7,8,9,10])$$
$$(\tau, E) = ((0,1,0,1,1,1,0,1,1,1,1,1), [1,2,3,4,5,6,7,8,9,10])$$
$$(\tau, E) = ((0,0,0,1,1,1,1,1,1,1,0,0), [0,1,2,3,4,5,6,7,8,10])$$

Pour chaque longueur d'encodage k, on a les fonctions de transfert τ optimales suivantes (ainsi que tous les $\tau.g$ avec $g \in D_6$), c'est-à-dire qui réalisent la meilleure distance moyenne pour tous les ensembles E de longueur k :

— $k = 2 \rightarrow d_{moy} = 5,92$

$$\tau = (0,0,0,1,1,0,1,1,1,0,1,1)$$

— $k = 3 \rightarrow d_{moy} = 4,58$

$$\tau = (0,0,0,1,1,1,1,1,1,0,0,1)$$

— $k = 4 \to d_{moy} = 4,1$ et $k = 5 \to d_{moy} = 3,64$

$$\begin{aligned}\tau &= (0,0,0,0,0,1,1,0,1,1,1,0)\\ &= (0,0,0,0,0,1,1,1,0,1,1,0)\end{aligned}$$

— $k = 6 \to d_{moy} = 2,75$

$$\tau = (0,0,0,0,0,1,1,0,1,1,1,0)$$

— $k = 7 \to d_{moy} = 2,26$

$$\begin{aligned}\tau &= (0,0,0,1,1,0,1,1,1,0,1,1)\\ &= (0,0,0,1,1,0,1,1,1,1,0,1)\end{aligned}$$

— $k = 8 \to d_{moy} = 2$

$$\begin{aligned}\tau &= (0,0,1,1,1,1,0,1,1,1,1,1)\\ &= (0,1,0,1,1,1,0,1,1,1,1,1)\\ &= (0,1,1,0,1,1,0,1,1,1,1,1)\\ &= (0,1,1,0,1,1,1,1,1,0,1,1)\\ &= (0,1,1,1,1,0,0,1,1,1,1,1)\\ &= (0,1,1,1,1,1,1,1,1,1,1,1)\end{aligned}$$

— $k = 9 \to d_{moy} = 2$ et $k = 10 \to d_{moy} = 2$

$$\tau = (0,1,1,1,1,1,1,1,1,1,1,1)$$

— $k = 11$, toutes les fonctions τ injectives

Communications et publications

Communications dans une conférence ou un séminaire

Marion Candau, Roland Gautier et Johannes Huisman : « Non-commutative convolutional codes over the infinite dihedral group », au congrès *Non-commutative rings and their applications*, Lens, France (1-4 Juillet 2013)

Marion Candau, Roland Gautier et Johannes Huisman : « Codes convolutifs non-commutatifs », invitée au Séminaire Calcul Formel et Complexité de l'IRMAR, Rennes, France (27 juin 2014)

Articles dans une revue à comité de lecture

Marion Candau, Roland Gautier et Johannes Huisman : « Non-commutative convolutional codes over the infinite dihedral group », International Journal of Information and Coding Theory, accepté (octobre 2014)

Articles en préparation

Marion Candau, Roland Gautier et Johannes Huisman : « Convolutional block codes over the semi-direct product $\mathbb{Z}/N\mathbb{Z} \rtimes \mathbb{Z}/M\mathbb{Z}$ », en préparation pour être soumis à une revue

Marion Candau, Roland Gautier et Johannes Huisman : « Convolutional block codes over quotients of the triangle group », en préparation pour être soumis à une revue

Bibliographie

[Bar05] Johann Barbier : Reconstruction of turbo-code encoders. *In Proceedings of SPIE Security and Defence, Space Communication Technologies Symposium*, volume 5819, pages 463–473, Orlando, FL, USA, mars 2005.

[BCJR74] Lalit R. Bahl, John Cocke, Frederik Jelinek et Josef Raviv : Optimal decoding of linear codes for minimizing symbol error rate (corresp.). *Information Theory, IEEE Transactions on*, 20(2):284–287, Mar 1974.

[BG03] Gilles Burel et Roland Gautier : Blind estimation of encoder and interleaver characteristics in a non cooperative context. *In IASTED International Conference on Communications, Internet and Information Technology*, Scottsdale, AZ, USA, novembre 2003.

[Bol32] János Bolyai : Appendix, scientiam spatii absolute veram exhibens. *Tentamen Juventutem studiosam in elementa Matheseos purae*, 1832.

[BSH06] Johann Barbier, Guillaume Sicot et Sébastien Houcke : Algebraic approch for the reconstruction of linear and convolutional error correcting codes. *International journal of applied mathematics and computer sciences*, 2(3):113–118, 2006.

[CF09] Mathieu Cluzeau et Matthieu Finiasz : Recovering a code's length and synchronization from a noisy intercepted bitstream. *In Proceedings of the 2009 IEEE International Conference on Symposium on Information Theory - Volume 4*, ISIT'09, pages 2737–2741, Piscataway, NJ, USA, 2009. IEEE Press.

[Clu04] Mathieu Cluzeau : Reconstruction of a linear scrambler. *In IEEE International Symposium on Information Theory (ISIT)*, page 230, Chicago, USA, 2004.

[Clu06] Mathieu Cluzeau : *Reconnaissance d'un schéma de codage*. Thèse de doctorat, Ecole Polytechnique, 2006.

[CS09] Maxime Côte et Nicolas Sendrier : Reconstruction of convolutional codes from noisy observation. *In Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, pages 546–550, Seoul, Korea, 2009.

[Dev89] Robert L. Devaney : *An Introduction to Chaotic Dynamical Systems*. Addison-Wesley, 2nd édition, 1989.

[DF92] Freeman J. Dyson et Harold Falk : Period of a discrete cat mapping. *The American Mathematical Monthly*, 99(7):pp. 603–614, 1992.

[DH07] Janis Dingel et Joachim Hagenauer : Parameter estimation of a convolutional encoder from noisy observation. *In IEEE International Symposium on Information Theory (ISIT)*, pages 1776–1780, Nice, France, Juin 2007.

[Eli55] Peter Elias : Coding for noisy channels. *In IRE International Convention Record*, mars 1955.

[EPI66] Euclide, François Peyrard et Jean Itard : *Les œuvres d'Euclide*. A. Blanchard, Paris, [reprod. en fac-sim.] édition, 1966. Fac-sim. de l'édition de Paris : chez C.-F. Patris, 1819.

[Fan63] Robert Fano : A heuristic discussion of probabilistic decoding. *Information Theory, IEEE Transactions on*, 9(2):64–74, April 1963.

[Fil97] Eric Filiol : Reconstruction of convolutional encoders over gf(q). *In Proceedings of the 6th IMA International Conference on Cryptography and Coding*, pages 101–109, London, UK, 1997. Springer-Verlag.

[For70] G. David Jr. Forney : Convolutional codes i : Algebraic structure. *Information Theory, IEEE Transactions on*, 16(6):720–738, nov 1970.

[Gop70] Valery D. Goppa : A new class of linear error-correcting codes. *Problemy Peredachi Informatsii*, 1970.

[Gra07] Markus Grassl : Bounds on the minimum distance of linear codes and quantum codes. Online available at http://www.codetables.de, 2007. Accessed on 2014-07-21.

[LDW94] Yuan Xing Li, R.H. Deng et Xin Mei Wang : On the equivalence of mceliece's and niederreiter's public-key cryptosystems. *IEEE Transactions on Information Theory*, 1994.

[LLWC13] Wenwen Li, Jing Lei, Lei Wen et Bin Chen : An improved method of blind recognition of rs code based on matrix transformation. *In Communication Technology (ICCT), 2013 15th IEEE International Conference on*, pages 196–200, Nov 2013.

[Lob30] Nikolai I. Lobachevsky : On the principles of geometry. *Kazanskij Vestnik*, pages 25–28, 1829-1830.

[Mal48] Anatoly I. Malcev : On the embedding of group algebras in division algebras. *Doklady Akad. Nauk SSSR (N.S.)*, 60:1499–1501, 1948.

[Mar09] Mélanie Marazin : *Reconnaissance en aveugle de codeur à base de code convolutif : Contribution à la mise en oeuvre d'un récepteur intelligent*. Thèse de doctorat, Université de Bretagne Occidentale, 2009.

[McE78] Robert J. McEliece : A public-key cryptosystem based on algebraic coding theory. *JPL DSN Progress Report 4244*, pages 114–116, 1978.

[MGB11] Melanie Marazin, Roland Gautier et Gilles Burel : Blind recovery of k/n rate convolutional encoders in a noisy environment. *EURASIP Journal on Wireless Communications and Networking*, page 168, 2011.

[Moo05] Todd K. Moon : Convolutional codes. *In Error Correction Coding : Mathematical Methods and Algorithms*, chapitre 12, pages 452–580. Wiley-Interscience, 2005.

[Neu49] Bernhard H. Neumann : On ordered division rings. *Trans. Amer. Math. Soc.*, 66:202–252, 1949.

[Neu07] Andre Neubauer : Convolutional codes. *In Coding Theory : Algorithms, Architectures and Applications*, chapitre 3, pages 112–177. Wiley-Interscience, 2007.

[Nie86] Harald Niederreiter : Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 1986.

[Rat06] John G. Ratcliffe : *Foundations of hyperbolic manifolds.* Graduate Texts in Mathematics 149. Springer New York Springer e-books, New York, NY, seconde édition, 2006.

[Ric95] B. Rice : Determining parameters of a rate 1/n convolutional encoder over gf(q). *In Proceedings of 3-rd International Conference on Finite Fields and Applications*, Glasgow, 1995.

[RS10] David Renard et Laurent Schwartz : *Groupes et représentations.* Éd. de l'École Polytechnique, 2010.

[Ser71] Jean-Pierre Serre : *Représentations linéaires des groupes finis.* Hermann, 1971.

[Sta11] Alexander Stanoyevitch : *Introduction to cryptography with mathematical foundations and computer implementations.* Discrete mathematics and its applications. CRC press, Boca Raton (FL) London New York (NY), 2011.

[Ulm12] Felix Ulmer : *Théorie des groupes.* Ellipses, Paris, 2012.

[Val00] Antoine Valembois : *Décodage, détection et reconnaissance des codes linéaires binaires.* Thèse de doctorat, Université de Limoges, 2000. Thèse doctorat : Mathématiques et Informatique.

[Val01] Antoine Valembois : Detection and recognition of a binary linear code. *Discrete Applied Mathematics*, 111(1–2):199 – 218, 2001. Coding and Cryptology.

[Vit67] Andrew Viterbi : Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, avril 1967.

[WHZ07] Fenghua Wang, Zhitao Huang et Yiyu Zhou : A method for blind recognition of convolution code based on euclidean algorithm. *In Proceedings of International Conference on Wireless Communications, Networking and Mobile Computing*, pages 1414–1417, sept 2007.

[Woz57] John M.R. Wozencraft : *Sequential decoding for reliable communication.* Technical report (Massachusetts Institute of Technology. Research Laboratory of Electronics). Massachusetts Institute of Technology., 1957.

[ZHL+13] Jing Zhou, Zhiping Huang, Chunwu Liu, Shaojing Su et Yimeng Zhang : Information-dispersion-entropy-based blind recognition of binary bch codes in soft decision situations. *Entropy*, 15:1705–1725, 2013.

[ZHSY13] Jing Zhou, Zhiping Huang, Shaojing Su et Shaowu Yang : Blind recognition of binary cyclic codes. *EURASIP Journal on Wireless Communications and Networking*, page 218, 2013.

Résumé

Un code correcteur d'erreur ajoute de la redondance à un message afin de pouvoir corriger celui-ci lorsque des erreurs se sont introduites pendant la transmission. Les codes convolutifs sont des codes performants, et par conséquent, souvent utilisés. Le principe d'un code convolutif consiste à se fixer une fonction de transfert définie sur le groupe des entiers relatifs et à effectuer la convolution d'un message avec cette fonction de transfert. Ces codes ne protègent pas le message d'une interception par une tierce personne. C'est pourquoi nous proposons dans cette thèse, des codes convolutifs avec des propriétés cryptographiques, définis sur des groupes non-commutatifs. Nous avons tout d'abord étudié les codes définis sur le groupe diédral infini, qui, malgré de bonnes performances, n'ont pas les propriétés cryptographiques recherchées. Nous avons donc étudié ensuite des codes convolutifs en bloc sur des groupes finis, avec un encodage variable dans le temps. Nous avons encodé chaque message sur un sous-ensemble du groupe différent à chaque encodage. Ces sous-ensembles sont générés de façon chaotique à partir d'un état initial, qui est la clé du cryptosystème symétrique induit par le code. Nous avons étudié plusieurs groupes et plusieurs méthodes pour définir ces sous-ensembles chaotiques. Nous avons étudié la distance minimale des codes que nous avons conçu et montré qu'elle est légèrement plus petite que la distance minimale des codes en blocs linéaires. Cependant, nous avons, en plus, un cryptosystème symétrique associé à ces codes. Ces codes convolutifs non-commutatifs sont donc un compromis entre correction d'erreur et sécurité.

Abstract

An error correcting code adds redundancy to a message in order to correct it when errors occur during transmission. Convolutional codes are powerful ones, and therefore, often used. The principle of a convolutional code is to perform a convolution product between a message and a transfer function, both defined over the group of integers. These codes do not protect the message if it is intercepted by a third party. That is why we propose in this thesis, convolutional codes with cryptographic properties defined over non-commutative groups. We first studied codes over the infinite dihedral group, which despite good performance, do not have the desired cryptographic properties. Consequently, we studied convolutional block codes over finite groups with a time-varying encoding. Every time a message needs to be encoded, the process uses a different subset of the group. These subsets are chaotically generated from an initial state. This initial state is considered as the symmetric key of the code-induced cryptosystem. We studied many groups and many methods to define these chaotic subsets. We examined the minimum distance of the codes we conceived and we showed that it is slightly smaller than the minimum distance of the linear block codes. Nevertheless, our codes have, in addition, cryptographic properties that the others do not have. These non-commutative convolutional codes are then a compromise between error correction and security.

université de bretagne occidentale

Université de Bretagne Occidentale
3 rue des Archives - CS93837 - 292238 Brest Cedex 3 - France
Téléphone : (33) 02 98 01 60 00 - Fax (33) 02 98 01 60 01