Département de formation doctorale en informatique **École doctorale de Paris VI**
UFR Informatique

Décodage des codes algébriques et cryptographie

Mémoire d'habilitation à diriger des recherches

présentée et soutenue publiquement le 7 juin 2007

pour l'obtention du

Diplôme d'habilitation à diriger des recherches de l'université Pierre et Marie Curie – Paris VI

(spécialité informatique)

par

Daniel Augot

Composition du jury

Rapporteurs :	Patrick	Fitzpatrick
	Daniel	Lazard
	Amin	Shokrollahi
Examinateurs :	Philippe	Flajolet
	François	Morain
	Nicolas	Sendrier
	Annick	Valibouze

Mis en page avec la classe thloria.

Table des matières

Remerciements

Je tiens tout d'abord à remercier mes trois rapporteurs. D'abord les étrangers. Merci à Patrick Fitzpatrick, qui connait bien mon travail, d'avoir bien voulu être rapporteur et d'avoir fait le déplacement depuis Cork, en Irlande.

Merci aussi à Amin Shokrollahi, pour l'intérêt qu'il porte à mon travail. Je tiens aussi à le remercier de m'avoir associé au jury de son étudiant Andrew Brown.

Enfin, merci à Daniel Lazard, pour l'exigence qu'il a montré vis-à-vis de ce document. Cet effort qu'il m'a demandé a été bien nécessaire.

Merci à Annick Valibouze d'avoir bien voulu m'éclairer sur quelques propriétés des fonctions symétriques élémentaires. Sa patiente relecture du document a également permis de l'améliorer.

Je suis très reconnaissant à François Morain, pour l'accueil qu'il me fait à l'école polytechnique.

Dois-je remercier Nicolas Sendrier, collègue de tous les jours, prêt à résoudre quasi instantanément tous les problèmes ?

Je suis aussi très honoré de la présence de Philippe Flajolet, plus qu'éminent, qui déjà avait montré de l'intérêt à ma thèse, et qui est encore présent maintenant.

Maintenant il me reste à remercier tous les collègues. Il y a tant de monde... Il y a d'abord Pascale Charpin qui toujours m'a aidé et soutenu, dans des moments difficiles, et encore Nicolas Sendrier, mais aussi Anne Canteaut, qui les ont compris.

Quant au projet Codes, serais-je capable de remercier tous ces gens qui m'ont tant apporté ? Je vais surtout énumérer les thésards que j'ai co-dirigé, avec d'autres directeurs eux habilités alors que je ne l'étais pas : Lancelot Pecquet, Cédric Tavernier, Magali Bardet et Raghav Bhaskar. Je suis content et fier de leur succès respectifs, dans différents domaines, dans différents lieux.

Enfin, salut à tous les étudiants, présents ou passés, du projet, pour l'ambiance sympathique qu'ils contribuent à instaurer dans cette équipe si chaleureuse. Je suis vraiment heureux de partager la vie quotidienne avec tous.

Chapitre 1

Introduction générale

La théorie des codes correcteurs est une discipline de l'*artefact*, qui cherche à *construire* des objets mathématiques particuliers, utilisables dans les schémas de communication, principalement pour protéger les messages contre le bruit. Un code définit une manière d'ajouter de la redondance à une suite de symboles, et le tout est transmis sur le canal de transmission. On espère que cette redondance permettra de reconstruire le message, même si des perturbations se sont produites.

La construction historique des codes correcteurs est algébrique : les grandes familles des codes cycliques, codes de Reed-Solomon, codes BCH, codes de Reed-Muller, reposent sur les polynômes univariés ou multivariés sur les corps finis. Les propriétés bien connues de ces objets mathématiques permettent de donner les propriétés des codes ainsi construits.

À ces objets mathématiques que sont les codes sont associés des *algorithmes de décodage*, qui permettent d'éliminer le bruit qui a perturbé la transmission. Il y a des algorithmes de décodage *génériques*, qui décodent tout code, et des algorithmes dédiés à chaque famille de code construite. Les algorithmes génériques sont peu efficaces, alors que pour un code donné, on peut tirer parti de sa structure pour construire l'algorithme de décodage. Il n'est en général pas facile de construire un algorithme de décodage efficace associé à un code donné. Étant donné la forte structure mathématique des codes que je considère, les algorithmes historiques de décodage utilisent, souvent sans le savoir, les méthodes de calcul formel : algorithme d'Euclide, algorithme de Berlekamp-Massey etc.

Mes travaux de recherche se situent donc à l'intersection de la discipline des codes correcteurs d'erreurs et du calcul formel, avec une intention d'applications des algorithmes du calcul formel au codage. Ma thèse portait sur la famille des codes cycliques : j'ai mis une *mise en équation* algébrique du problème de déterminer la distance minimale des codes cycliques [Aug93]. En collaboration avec Paul Camion, j'y ai aussi donné un algorithme de construction de base normale dans un corps fini, reposant sur le calcul de vecteurs cycliques et de la forme de Frobenius d'une matrice. Mais je n'ai pas développé par la suite cet axe de recherche précis (algèbre linéaire sur les corps finis).

1.1 Travaux en codage

Après ma thèse, je me suis intéressé au problème du décodage des codes cycliques, que j'ai essayé de mettre en équations. J'ai utilisé principalement une mise en équation induite par les *identités de Newton*. Le problème est l'étude de la variété associée à cette mise en équation : il n'est pas évident que les solutions obtenues par résolution du système correspondent aux solutions du problème de décodage asso-

cié. La ligne de mire de ces travaux est le décodage d'un sous ensemble des codes cycliques, les codes à résidus quadratiques, qui sont de bons codes, mais qu'on ne sait pas décoder de manière efficace. Ces mises en équations sont ensuite résolues avec les logiciels de calcul de base de Gröbner de Jean-Charles Faugère, qui, à l'époque de ma thèse, étaient déjà les plus performants. Ces problèmes de décodage des codes cycliques ont formé une partie de la thèse de Magali Bardet en 2004, que j'ai co-encadrée [Bar04], avec Jean-Charles Faugère.

Une autre famille de codes, plus restreinte encore que celle des codes cycliques, mais très importante, est celle des codes de Reed-Solomon. Ces codes sont optimaux relativement à certaines bornes, et présentent la propriété de bien se décoder. Ils sont ainsi largement employés, dans de nombreuses situations.

En 1996, une percée remarquable a été obtenue par Madhu Sudan. En relâchant quelque peu la problématique de la correction d'erreurs, il a obtenu, pour les codes de Reed-Solomon, un algorithme de correction capable de supprimer beaucoup plus de bruit que les algorithmes classiques. Ce résultat a valu a Sudan le prix Nevanlinna 2002 (et aussi pour d'autres travaux, en théorie de la complexité notamment). Cet algorithme, conceptuellement très simple, est très algébrique. Il n'a pas été complètement décrit par Sudan. En effet, il procède en deux étapes : une étape d'interpolation bivariée, et une étape de factorisation bivariée. Sudan s'est contenté de signaler que ces deux étapes peuvent se faire de manière « polynomiale ». La porte était ouverte à la recherche de bons algorithmes pour résoudre ces deux problèmes, et aussi à la généralisation de cet algorithme à d'autres codes que les simples codes de Reed-Solomon.

Au premier rang des codes candidats à être décodés par des variantes de l'algorithme de Sudan figurent les *codes géométriques*. J'ai co-encadré la thèse de Lancelot Pecquet sur ces sujets : formulation de l'algorithme de Sudan pour les codes géométriques, et optimisation des méthodes de calcul formel pour l'algorithme de Sudan, pour les codes de Reed-Solomon et pour les codes géométriques [Pec01].

L'algorithme de Sudan a ouvert la brèche du décodage d'un grand nombre d'erreurs. Dans ce domaine, j'ai aussi co-encadré, avec Pascale Charpin, la thèse de Cédric Tavernier sur le décodage des codes de Reed-Muller. Ces codes, moins « bons » que les codes BCH ou les codes de Reed-Solomon, sont toutefois des objets très naturels, qui trouvent de nombreuses applications en cryptographie. Les codes de Reed-Muller sont construits en utilisant des polynômes multivariés de petit degré sur le corps $\mathbb{F}_2$: le problème de décodage de ces codes est un problème d'approximation de points par un polynôme de bas degré. Cédric Tavernier a trouvé et implémenté un bon algorithme pour trouver les meilleures approximations d'un ensemble de points, et a ensuite appliqué son algorithme pour trouver les meilleures approximations de certaines sorties de l'algorithme de chiffrement DES [Tav04].

Ces approximations peuvent être utilisées pour monter une attaque particulière, la *cryptanalyse linéaire* de Matsui [Mat94b]. Ce n'est pas mon sujet de mener cette cryptanalyse complètement, et l'algorithme de Cédric Tavernier permet de faire automatiquement le travail préalable de trouver les approximations, travail que le cryptanalyste devait auparavant faire à la main.

1.2 Travaux en cryptographie

En ce domaine de la cryptographie, j'ai aussi travaillé sur l'exploitation des résultats négatifs en codage. Pour beaucoup de codes, le problème décisionnel associé au décodage est NP-complet, et surtout il est effectivement réputé difficile dans la communauté. De nombreux cryptosystèmes à clé publique ont été construits sur cet aspect négatif de la théorie des codes, dont le système fondateur de McEliece [McE78]. Avec Matthieu Finiasz, j'ai proposé un nouvel algorithme de chif-

frement à clé publique (hélas « cassé »), et surtout une fonction de hachage avec réduction de sécurité. Ces travaux ont fait l'objet d'une partie de la thèse de Matthieu Finiasz [Fin04].

Toujours en cryptographie, j'ai mené une activité beaucoup plus proches des applications, sur des fondements très différents. Dans ces travaux-là, je n'ai plus utilisé la théorie des codes correcteurs d'erreurs, et j'ai utilisé à chaque fois le protocole fondateur de la cryptographie à clé publique : celui de Diffie-Hellman [DH76] ou ses variantes. Ce protocole permet à deux utilisateurs de se mettre d'accord sur une clé secrète en échangeant uniquement des données publiques au su et au vu de l'attaquant. Cette clé secrète peut ensuite être utilisée pour sécuriser des échanges ultérieurs, en utilisant la cryptographie à clé secrète.

Presque tout de suite après ma thèse, avec Caroline Fontaine, j'ai travaillé au problème de la protection des droits d'auteurs dans un projet d'intégration et de diffusion des ressources patrimoniales européennes. Nous avons proposé un système particulier de gestion des clés, et nous sommes allés jusqu'à la réalisation d'un prototype, qui a fait l'objet d'une démonstration devant des représentants de la Commission Européenne. Ces travaux constituent la moitié de la thèse de Caroline Fontaine [Fon98].

Enfin, dans le cadre d'une collaboration avec les projets Arles (architectures logicielles et systèmes distribués) et Hipercom (réseaux sans fil) de l'INRIA, j'ai eu aussi à m'intéresser à quelques aspects de la sécurisation des réseaux sans fil. Une des solutions envisagées a été de mettre en place dans le réseau une clé secrète, en utilisant une version à n utilisateurs du protocole de Diffie-Hellman. Une telle généralisation n'existe pas de manière directe ou canonique. Avec Raghav Bhaskar, nous avons conçu une variante du protocole de Diffie-Hellman, qui présente des caractéristiques intéressantes : il n'y a pas besoin d'une forte structuration des participants ; une défaillance d'un des participants n'empêche pas les autres membres d'obtenir la clé secrète. Ces caractéristiques rendent ce protocole très adapté aux réseaux avec perte de message, et nous pensons pousser assez loin la mise en œuvre pratique de ce protocole. Ces travaux étaient le cœur de la thèse de Raghav Bhaskar [Bha06], que j'ai co-encadrée avec Valérie Issarny.

Structure du document Ce document présente mes travaux de recherche depuis la soutenance de ma thèse, en prétendant à l'exhaustivité. Il y a cependant certains de mes travaux que j'ai choisi d'exposer avec un certain niveau de détails, et d'autres de manière plus succincte. C'est pour cela que le document est structuré en deux principaux volets : un volet de survol de mes travaux, en codage et en cryptographie ; un volet de présentation précise de mes travaux en codage.

Le premier volet est en deux parties : codage, cryptographie. En codage, j'ai estimé nécessaire de faire une présentation générale du domaine, pour amener les problématiques qui m'ont principalement intéressé : le décodage des codes cycliques, et le décodage des codes d'évaluation. En cryptographie, les problématiques dont je parle sont plus simples à aborder : je présente dans un premier chapitre mes constructions de systèmes à clé publique reposant sur la théorie des codes correcteurs d'erreurs, et dans un deuxième chapitre mes travaux applicatifs reposant sur le protocole de Diffie-Hellman (protection des droits d'auteurs, réseaux sans fil).

Le deuxième volet ne parle que du codage et détaille les constructions des codes considérés, leurs définitions et les algorithmes de décodage. La partie sur le décodage des codes cycliques avec les bases de Gröbner contient un certain nombre de résultats nouveaux sur la mise en équation que j'ai étudiée, résultats non publiés. La présentation est assez abrupte, et essentiellement technique, avec toutes les preuves. La partie sur le décodage à la Sudan détaille des généralisations multivariées de Sudan, en mettant l'accent sur les difficultés rencontrées, qui sont celles, générales, du

passage des polynômes univariés aux polynômes multivariés.

Enfin, une partie « administrative », récapitule les étudiants que j'ai co-encadrés, mes activités d'enseignement, donne la liste classée de mes publications, et la liste des contrats de recherche dans lesquels j'ai été impliqué.

Première partie

Codage

Chapitre 2

Introduction au codage algébrique

Dans ce chapitre, je présente à grands traits la problématique du décodage, et les notions de la théorie des codes correcteurs, de manière à introduire le sujet de mes travaux. En effet la problématique du codage et du décodage n'est pas connue en général par les non-spécialistes, aussi m'a-t'il semblé nécessaire de bien faire ces rappels. Il faut de plus introduire une certaine « botanique », un peu déroutante, de codes correcteurs d'erreurs (codes cycliques, BCH, à résidus quadratiques etc), parmi les plus classiques.

Il faut comprendre qu'en grande partie, la théorie des codes correcteurs produit des artefacts : on construit des objets très particuliers. Ces objets sont certains codes et leurs algorithmes de décodage, pour traiter un problème, celui de la transmission fiable des données.

Mes travaux en codage sont situés dans le domaine des algorithmes de décodage des codes ayant une définition fortement algébrique. La définition de ces codes repose sur les propriétés des polynômes univariés ou multivariés sur un corps fini. Les algorithmes pour les décoder utilisent les outils du calcul formel. Notamment, j'utilise les bases de Gröbner pour décoder tous les codes cycliques. Les méthodes issues du calcul formel sont aussi employées pour améliorer l'algorithme de Guruswami-Sudan, pour décoder les codes de Reed-Solomon.

2.1 Problématique du codage et théorème de Shannon

Le problème à l'origine de l'émergence de la théorie des codes correcteurs est celui de la transmission fiable d'information sur des canaux de communication soumis à des perturbations. Dans un tel canal, le signal reçu après propagation dans le canal bruité peut être différent que celui émis. Le problème dépend du canal de communication, et de la manière dont celui-ci affecte les signaux émis. Un des modèles les plus étudiés est celui du *canal binaire symétrique* : chaque bit 0 ou 1 émis par la source peut être transformé en son complémentaire avec une probabilité $p < 1/2$.

L'idée naturelle pour améliorer la transmission est d'ajouter de la *redondance* au message émis. À la réception, on espère ensuite d'être capable d'utiliser cette redondance pour retrouver le message émis.

Le système le plus simple est de répéter chaque bit émis, par exemple $2t + 1$ fois. Ainsi, pour $t = 1$, on transmet 000 pour 0, et 111 pour 1. La règle de décodage

est celle du décodage majoritaire : on décode en 0 si le mot reçu a une majorité de 0 et en 1 sinon. Le calcul des probabilités permet de montrer que le taux d'erreur après correction tend vers zéro quand t croît. Le défaut de cette méthode est que le nombre de bits transmis par bit utile est grand quand t croît : on consomme beaucoup trop de bande passante pour transmettre un bit.

De manière plus générale, on considère les symboles à émettre dans un corps fini $\mathbb{F}_q$. On va les regrouper par paquets de k symboles, avant de les *encoder* en *mots* de longueur n, avec $n > k$. En termes mathématiques, on a une fonction de codage, linéaire, injective :

$$\begin{array}{rccc} \phi: & \mathbb{F}_q^k & \to & \mathbb{F}_q^n \\ & m & \mapsto & \phi(m) = c \end{array}$$

et on dit que m est le *message*, et c est le *mot de code* associé. Le *code* C est l'image de ϕ : $C = \phi(\mathbb{F}_q^k)$. On dit que C est un code de longueur n et de dimension k. Le *taux de transmission* (ou *rendement*) du code est le rapport k/n, noté R. C'est une quantité qu'on cherche à optimiser.

On émet le mot de code c. Le canal transforme le mot émis $c \in C \subset \mathbb{F}_q^n$ en un mot *bruité* $c' \in \mathbb{F}_q^n$, qui n'est plus nécessairement dans le codes C. Du coté de la réception, on essaye de retrouver c à partir de c'. Un *algorithme de décodage associé à* C est un algorithme D :

$$\begin{array}{rccc} D: & \mathbb{F}_q^n & \to & C \cup \{\epsilon\} \\ & c' & \mapsto & D(c') \end{array}$$

tel que $D(c') = c$, si c' n'est pas trop différent de c, ou $D(c') = \epsilon$, si l'algorithme échoue (s'il y a trop d'erreurs, par exemple).

Le *théorème de codage de canal*, démontré par Shannon en 1948, justifie cette approche, qui consiste à ajouter de la redondance. Pour chaque canal, il existe un taux limite R_0, tel que, pour tout taux de transmission $R < R_0$, il existe un code C de taux transmission R, et un algorithme de décodage associé, tel que le taux d'erreur après décodage tend vers zéro, quand la longueur de C croît.

Ce théorème célèbre a une preuve probabiliste, et il est non constructif. Il ne dit pas quels codes choisir, et ni comment construire les algorithmes de décodage associés. On peut dire que l'objectif majeur du codage est la construction effective de codes et de leurs algorithmes associés, permettant de réaliser les promesses du théorème de Shannon.

2.2 La distance de Hamming

Dans le cas des canaux dits *q-aires symétriques*, cette problématique se traduit en termes de mathématiques discrètes, comme l'a montré Hamming en 1950.

L'*espace de Hamming* est $\mathbb{F}_q^n$ muni de la *distance de Hamming* :

$$d(x, y) = \{i \in \{1, \ldots, n\} |\, x_i \neq y_i\},$$

et on définit aussi le poids d'un mot $x \in \mathbb{F}_q^n$

$$w(x) = \{i \in \{1, \ldots, n\} |\, x_i \neq 0\} = d(x, 0).$$

Pour un code $C \subset \mathbb{F}_q^n$ de dimension k, on définit la *distance minimale* de C, $d(C)$:

$$d(C) = \min_{x,y \in C, x \neq y} d(x, y).$$

On parle de code $[n, k, d]_q$ qui est la notation consacrée.

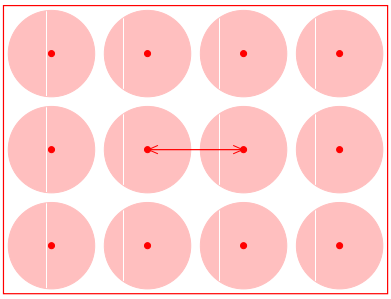

FIG. 2.1 – Pour un code de distance minimale d, les sphères centrées sur les mots du code de rayon $t = \lfloor (d-1)/2 \rfloor$ sont disjointes.

Pour les canaux en question, on connaît l'algorithme de décodage qui minimise la probabilité d'erreur. C'est celui qui retourne le mot de code le plus proche du mot reçu, pour la distance de Hamming. C'est le *décodage à maximum de vraisemblance*. On montre que la probabilité d'erreur après décodage à maximum de vraisemblance est d'autant plus faible que la distance minimale est grande. Mais on ne connaît *aucun* code pour lequel il existe un algorithme de décodage à maximum de vraisemblance vraiment meilleur que la recherche exhaustive sur tous les mots du code. Le décodage à maximum de vraisemblance est donc un objectif trop ambitieux, et on se contente de corriger un nombre limité d'erreurs.

On note traditionnellement $t = \lfloor (d-1)/2 \rfloor$, qui est la *capacité de correction* de C. Si moins de t erreurs se sont produites, alors il y a unicité du mot de code le plus proche (voir la figure 2.1). On dit que le code C est t-correcteur.

Dans ce modèle là, les problèmes principaux de la théorie deviennent :

1. la construction : trouver des « bons » codes, ayant une bonne dimension k et une bonne distance minimale d. Ces deux objectifs sont antagonistes.

2. le décodage : trouver les algorithmes de décodage associés.

2.3 Obstacles

En ce qui concerne le problème de la construction de bons codes, un code linéaire de longueur n et de dimension k pris au hasard aura une bonne distance minimale, avec une très forte probabilité. Mais nous avons un résultat de complexité dû à Vardy [Var97], qui montre que le problème décisionnel associé au calcul de la distance minimale est un problème NP-complet. La théorie de la complexité donne seulement une indication sur le pire cas, pour des algorithmes déterministes. Mais la pratique montre aussi que déterminer la distance minimale d'un code est difficile pour les instances aléatoires. Par la pratique, je pense aux algorithmes déterministes aussi bien qu'aux algorithmes probabilistes.

En ce qui concerne le décodage, on a aussi que le problème décisionnel associé au décodage est lui aussi NP-complet [BMvT78]. Comme précédemment, la pratique montre que les instances aléatoires sont tout aussi difficiles. Ce résultat a été renforcé par Bruck et Naor, qui montrent que le problème du décodage, même en autorisant un temps de précalcul aussi long que l'on désire, reste un problème difficile [BN90]. Cette dernière considération a beaucoup de sens et est bien réelle, car dans un schéma de communication, le code est fixé, et on décode répétitivement des mots bruités : on peut donc s'autoriser un précalcul énorme. On aurait donc pu espérer gagner beaucoup en faisant un précalcul sur le code, mais ce n'est pas le cas.

Donc, pour résumer, en reprenant dans l'ordre les points 1 et 2 du paragraphe précédent :

1. construction : il est difficile de connaître la distance minimale d'un code donné d, donc sa capacité de correction t (sauf cas exceptionnel) ;
2. décodage : il est difficile de décoder un code donné (sauf cas exceptionnel).

On cherche à construire des codes dont on puisse prouver quelque chose sur la distance minimale, et pour lesquels on puisse construire des algorithmes de décodage associés efficaces. Ce seront des codes très particuliers : on va du cas général des codes linéaires vers des codes particuliers, voire exceptionnels. Plusieurs voies sont possibles pour obtenir ces constructions, la mienne est dans la tradition du *codage algébrique*, qui utilise des outils issus de l'algèbre pour élaborer ces constructions. Je vais parler principalement des codes cycliques, des codes BCH, des codes à résidus quadratiques, des codes de Reed-Solomon, et des codes géométriques, voir figure 2.2.

2.4 Les codes cycliques

Les *codes cycliques* sont une famille très restreinte de codes, au premier rang desquels on trouve les codes BCH (Bose-Chauduri-Hocquenghem).

Les codes BCH C'est une construction qui, étant donnée une distance minimale d_0 et une longueur n, donne un code de longueur n et distance minimale d, $d \geq d_0$. De plus, pour $t_0 = \lfloor (d_0 - 1)/2 \rfloor$, les codes obtenus ont un algorithme de décodage associé corrigeant t_0 erreurs, de faible complexité.

Toutefois, il se peut que la distance minimale d d'un code BCH soit plus grande que d_0, et que sa capacité de correction $t = \lfloor (d - 1)/2 \rfloor$ soit supérieure à t_0. Le problème de la détermination de la vraie distance minimale des codes BCH se pose. Ensuite se pose aussi le problème de la correction d'erreurs jusqu'à la vraie capacité de correction.

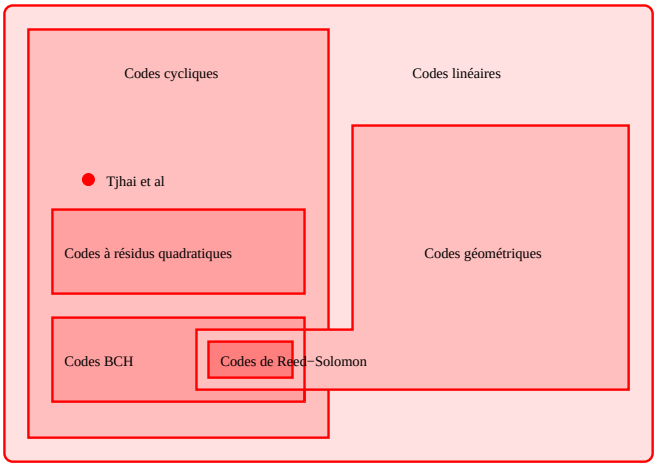

FIG. 2.2 – Inclusion des principales familles de codes algébriques. Les surfaces ne rendent pas compte des réelles proportions respectives de ces familles de codes. Par exemple, la famille des codes cycliques est très petite par rapport à celle des codes linéaires.

Dans ma thèse [Aug93], j'ai montré comment il est possible de déterminer la distance minimale d d'un code BCH en faisant des calculs de bases de Gröbner. Ensuite, j'ai réalisé que des approches à base de base de Gröbner peuvent être utilisées pour le problème du décodage jusqu'à la vraie capacité de correction t. Cette dernière veine a été développée avec Jean-Charles Faugère, en encadrant Magali Bardet pour une partie de sa thèse [Bar04].

Ces résultats permettent en réalité de calculer la distance minimale de *tout* code cyclique [Aug93, ACS90, Aug96a, Aug94], et de décoder *tout* code cyclique jusqu'à sa vraie capacité de correction [Aug98]. Dans la classe des codes cycliques, on peut donc chercher de meilleurs codes que les codes BCH, car les codes BCH ont le défaut de ne pas avoir une très bonne dimension, quand la distance minimale est grande.

Les codes à résidus quadratiques Parmi les codes cycliques, les *codes à résidus quadratiques* ont une bonne distance minimale, pour un bon taux de transmission 1/2. Mais ce sont des codes pour lesquels aucun algorithme efficace de décodage n'existe. Voir figure 2.2 pour les classes principales de codes. Le problème de décoder les codes à résidus quadratiques est si difficile à aborder que différents auteurs traitent au cas par cas le problème du décodage de ces codes. Pour chaque instance de petite longueur ($n = 23$, 31, 41, 47 etc) de ces codes, ces auteurs vont concevoir un algorithme de décodage associé. Ces algorithmes sont en fait des *formules*, précalculées, que l'on évalue pour chaque mot à décoder. Le résultat de cette évaluation donne le mot de code le plus proche. Les résultats que j'ai obtenus avec Magali Bardet et Jean-Charles Faugère [Bar04, ABF02, ABF03, ABF05] montrent que ces formules peuvent être obtenues automatiquement, par la théorie de l'élimination et les bases de Gröbner. Si d'autres travaux ont déjà proposé d'utiliser les bases de Gröbner pour décoder les codes cycliques, notre mise en équation sapparaît comme plus facile à résoudre en pratique.

Plutôt que de chercher des formules, on peut aussi résoudre le système d'équations en ligne, pour chaque mot reçu : il apparaît en réalité que les formules deviennent vite gigantesques (pour les codes à résidus quadratiques), alors que le calcul de la base de Gröbner, une fois les paramètres instanciés, se déroule rapidement.

2.5 Le décodage en liste et l'algorithme de Sudan

On peut relâcher l'hypothèse que l'algorithme de décodage retourne *un seul* mot de code, et chercher à décoder $\tau > t$ erreurs. Dans le cas le pire, il y a plusieurs solutions au problème de corriger τ erreurs (voir figure 2.3). On exige que l'algorithme de décodage retourne *tous* les mots de codes à distance τ du mot reçu. On appelle cela le *décodage en liste*.

Le notion du décodage en liste a été introduite par Elias et Wozencraft à la fin des années 1950 [Eli57, Woz58]. Ils l'ont introduite de manière théorique, en étudiant les bénéfices du décodage en liste pour la qualité de la transmission, sans donner d'algorithmes le réalisant. Cette situation a perduré juqu'au milieu des années 1990. Les premiers algorithmes de décodage en liste sont dûs à Sudan et à Guruswami, à la fin des années 1990.

On peut se demander quel mot choisir lorsque l'algorithme de décodage en liste retourne plusieurs solutions. C'est en fait une situation du *cas le pire*. Lorsque le rayon de décodage τ n'est pas trop grand, alors il y a dans *le cas moyen* un seul mot à distance τ du mot reçu, voir figure 2.3. On peut aussi lever l'ambiguïté en se reposant sur une éventuelle redondance naturelle des mots émis (par exemple, ils appartiennent au français), ou ne retenir que le mot de code le plus vraisemblable dans la liste renvoyée par l'algorithme.

La problématique centrale du décodage en liste est de savoir dans quelle mesure τ peut augmenter. Quand τ devient trop grand, le nombre de mots de codes dans

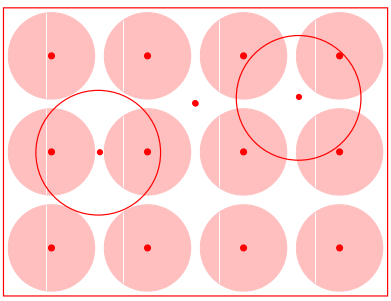

FIG. 2.3 – Décodage en liste : le point A est à égale distance de deux mots de code. Le point B a un seul mot de code à distance $\tau > t$. Le point A correspond au cas le pire, le point B au cas moyen.

une boule de rayon τ devient exponentiel en la longueur n. Il ne peut donc pas y avoir d'algorithme efficace pour corriger beaucoup d'erreurs, et la sortie d'un tel algorithme ne serait pas utilisable, à cause de la taille de la liste. Pour avoir des algorithmes raisonnables de décodage en liste, on doit donc faire croître τ sous la contrainte que la taille de liste retournée reste petite.

Comme précédemment pour le décodage classique, on ne connait pas d'algorithme de décodage en liste des codes quelconques. Les algorithmes de Sudan et Guruswami sont les premiers exemples d'algorithmes de décodage en liste, mais seulement pour la classe très restreinte des *codes de Reed-Solomon*, voir figure 2.2.

2.6 Les codes de Reed-Solomon

Une classe tout-à-fait exceptionnelle de codes est la famille des codes de Reed-Solomon, qui est un cas particuler des codes BCH. Ce sont des codes définis sur un gros alphabet $\mathbb{F}_q$, non binaire, et qui sont de longueur $n \leq q$. Ils sont donc de longueur bornée à taille d'alphabet q fixée. Ce sont des *codes optimaux*, ils vérifient l'égalité $d = n - k + 1$, où k est la dimension et d la distance minimale (la *borne de Singleton* affirme que la distance minimale de tout code $[n, k, d]_q$ vérifie $d \leq n-k+1$). On peut donc corriger jusqu'à

$$t = \lfloor (n-k)/2 \rfloor$$

erreurs. Avec un décodage unique, c'est le meilleur taux de correction qu'on puisse obtenir. Ces codes sont très employés en pratique, avec un spectre large d'applications : stockage (CD, DVD), transmissions (ADSL, TNT), etc.

Le problème du décodage des codes de Reed-Solomon se reformule simplement de la manière suivante.

> Étant donnés n points $(x_i, y_i) \in \mathbb{F}_q^2$, et un degré $k < n$, trouver les polynômes $f(X)$ de degré au plus k, tel que $f(x_i) = y_i$ pour le plus d'indices i possible.

Ce problème est quelquefois appelé le problème de la reconstruction de polynômes. En utilisant cette formulation du problème, Sudan et Guruswami [Sud97, GS99] ont conçu un algorithme permettant de décoder jusqu'à

$$\tau = \lfloor n - \sqrt{kn} \rfloor$$

erreurs. Si l'on rapporte le rayon de décodage à la longueur en formant la quantité τ/n, et qu'on l'étudie par rapport au taux de transmission $R = k/n$, on obient

$$\frac{\tau}{n} \leq 1 - \sqrt{R},$$

alors que le rayon classique est

$$\frac{t}{n} \leq \frac{1-R}{2}.$$

Ces deux rayons de décodage sont comparés dans la figure 2.4.

Cet algorithme est fondamental pour les raisons suivantes :

- il décode beaucoup plus d'erreurs que le décodage unique ;
- il permet de corriger des taux d'erreurs proche de 1, alors que les algorithmes classiques ne permettent de décoder que jusqu'à un taux d'erreurs de $1/2$;
- le rayon obtenu, $1-\sqrt{R}$, semble être optimal. On sait qu'il l'est pour les codes généraux, on ne sait pas s'il l'est pour les codes de Reed-Solomon.

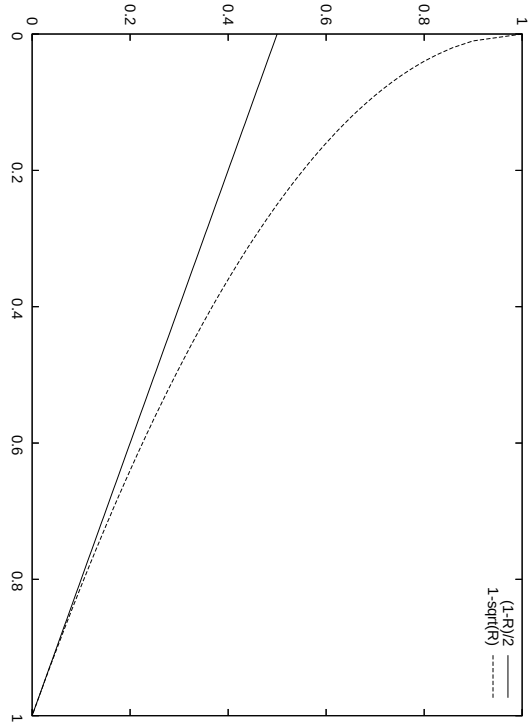

FIG. 2.4 – Rayon de décodage de Guruswami-Sudan $1 - \sqrt{R}$, comparé au rayon de décodage classique $(1 - R)/2$.

Il est aussi fondamental car il s'est révélé contenir des principes généraux, permettant de décoder d'autres sortes de codes, et sur des canaux de transmissions plus généraux que le canal q-aire symétrique. Madhu Sudan a obtenu le prix Nevanlinna 2002, pour l'ensemble de ses travaux, dont le décodage en liste des codes de Reed-Solomon.

Lancelot Pecquet, sous ma direction, a contribué à la généralisation de cet algorithme à la classe des *codes géométriques* [Pec01]. Cette classe de codes est importante pour la principale raison suivante. Alors que les codes de Reed-Solomon sont de longueur bornée, à alphabet $\mathbb{F}_q$ fixé, la longueur des codes géométriques peut croître, tout en restant de « bons » codes.

Du point de vue de son implantation, l'algorithme de Guruswami-Sudan est très proche du calcul formel. Il utilise comme ingrédients de base

1. une interpolation bivariée : il faut trouver un polynôme $Q(X, Y)$ passant par les points (x_i, y_i), avec une certaine multiplicité ;
2. une recherche de racine : il faut trouver tous les polynômes $f(X)$ tel que $Q(X, f(X)) = 0$.

Guruswami et Sudan ont simplement considéré que ces étapes pouvaient se faire en temps polynomial, ce qui n'est pas suffisant, et cet algorithme est souvent considéré comme trop « lourd »pour être utilisé.

J'ai contribué avec Lancelot Pecquet à améliorer la deuxième étape de cet algorithme, dans le cas de codes de Reed-Solomon, et dans le cas des codes géométriques [AP98b, AP00a]. On peut considérer maintenant que cette dernière étape peut-être efficacement réalisée avec une bonne complexité, et le point d'obstruction est le premier point, l'interpolation.

J'ai aussi étudié la généralisation de l'algorithme de Guruswami-Sudan aux *codes de Reed-Muller*, qui sont une généralisation multivariée des codes de Reed-Solomon [AEKM+06]. J'obtiens un meilleur rayon de décodage que Pellikaan et Wu, qui ont aussi abordé ce problème [PW04a, PW04b].

2.7 Conclusion

On peut décoder *tout* code cyclique jusqu'à sa vraie capacité de correction, en utilisant une mise en équation du problème du décodage. Il faut écrire la mise en équation pour chaque code considéré, et cela s'oppose aux algorithmes génériques, par exemple pour le décodage des codes BCH. Avec Magali Bardet et Jean-Charles Faugère, nous avons étudié une nouvelle mise en équation, plus efficace que celles précédemment considérées. Nous pouvons soit obtenir des formules en fonction des paramètres avec un précalcul de base de Gröbner, soit effectuer le calcul de la base de Gröbner en ligne, pour chaque mot reçu : cette dernière approche est la plus efficace. Nous avons étudié aussi le problème de décoder au delà de la capacité classique de correction. Mais nous n'avons gagné que quelques unités par rapport au rayon classique de décodage.

Le décodage en liste des codes de Reed-Solomon permet décoder bien au delà du rayon de décodage classique. C'est la percée fondamentale de Sudan et Guruswami. Avec Lancelot Pecquet, j'ai étudié le volet implémentation de cet algorithme, et la généralisation de cet algorithme aux codes géométriques, et aux codes de Reed-Muller.

Chapitre 3

Décodage des codes cycliques généraux

Dans ce chapitre, je rappelle le principe du décodage de *tout* code cyclique avec des bases de Gröbner, jusqu'à la *vraie* capacité de correction. Cette idée a déjà été considérée par différents auteurs. Ce décodage repose sur la notion de syndrome, et de décodage par syndrome, qui est assez technique à introduire pour les codes cycliques. Il y a plusieurs mises en équations du problème, et celle que j'ai étudiée repose sur les identités de Newton.

Il y a de plus deux manières de décoder avec les bases de Gröbner. Soit faire un précalcul de base de Gröbner, pour obtenir des *formules* pour le décodage, soit faire un calcul de base de Gröbner pour chaque mot reçu. Nous verrons que la deuxième solution est la plus efficace, et permet de décoder les codes BCH en grande longueur, et les codes à résidus quadratiques en longueur moyenne.

3.1 Définition des codes cycliques avec la transformée de Fourier

Les codes cycliques binaires sont des codes de longueur n, définis sur $\mathbb{F}_2$, avec n impair. À tout mot $c = (c_0, \ldots, c_{n-1})$ de longueur n, on lui associe le polynôme $c(X) = c_0 + c_1 + \cdots + c_{n-1}X^{n-1}$.

Soit α une racine primitive n-ième de l'unité, dans une extension de degré m de $\mathbb{F}_2$: $\alpha \in \mathbb{F}_{2^m}$. La transformée de Fourier discrète de c est le vecteur $S = (S_0, \ldots, S_{n-1})$, avec

$$S_i = c(\alpha^i), \quad i \in \{0, \ldots, n-1\}.$$

On peut passer de mot S au mot c, par une opération similaire.

Pour définir un code cyclique C, il faut se donner un *ensemble de définition* $Q \subset \{0, \ldots, n-1\}$. Le code cyclique d'ensemble de définition $Q = \{i_1, \ldots, i_l\}$ est l'ensemble des mots c dont leur transformée de Fourier $S = (S_0, \ldots, S_{n-1})$ vérifie :

$$S_{i_1} = \cdots = S_{i_l} = 0.$$

Les *codes BCH au sens strict de distance construite* δ sont des codes cycliques tels que l'ensemble de définition contient l'intervalle $\{1, \ldots, \delta - 1\}$. Leur distance minimale est alors supérieure ou égale à δ. On a de plus un algorithme efficace qui décode jusqu'à $\lfloor (\delta - 1)/2 \rfloor$ erreurs. En réalité leur distance minimale peut être supérieure à δ, et on ne sait pas comment étendre les algorithmes existant pour décoder plus d'erreurs.

Une autre classe de code cyclique est la classe de *codes à résidus quadratiques*, dont l'ensemble de définition Q est l'ensemble des résidus quadratiques modulo n. On peut montrer que la distance minimale de ces codes est supérieure à $\lfloor\sqrt{n}\rfloor$. Les tables montrent que pour les petites longueurs, la distance minimale d de ces codes est bien meilleure que la borne $\lfloor\sqrt{n}\rfloor$. Ce sont des codes pour lesquels on ne connaît pas d'algorithme efficace pour les décoder jusqu'à $\lfloor(\sqrt{n}-1)/2\rfloor$, encore moins jusqu'à $\lfloor(d-1)/2\rfloor$. Berlekamp les a qualifiés de « bons codes difficiles à décoder » [Ber68].

3.2 Erreur et syndromes

Soit C un code cyclique de longueur n et d'ensemble de définition $Q \subset \{1,\dots,n-1\}$. Décoder un mot $y \in \mathbb{F}_2^n$ donné revient à trouver le mot $c \in C$ le plus proche de y. On écrit $y = c + e$, où e et c sont inconnus, et où e est l'erreur. Si on trouve l'erreur e, alors on retrouve le mot de code, en faisant $c = y - e$.

Soit S la transformée de Fourier, connue, de y, alors

$$S_i = y(\alpha^i) = c(\alpha^i) + e(\alpha^i), \quad i \in \{0,\dots,n-1\}.$$

Or, pour $i \in Q$, on a $c(\alpha^i) = 0$, et donc

$$S_i = e(\alpha^i), \quad i \in Q. \tag{3.1}$$

Donc si $Q = \{i_1,\dots,i_l\}$, les coefficients $S_{i_1},\dots,S_{i_l}$ de la transformée de Fourier de e sont connus. On les appelle les *syndromes* de e. Si on connaissait tous les coefficients de la transformée de Fourier de e, alors e serait déterminé, et on pourrait retrouver le mot de code c. Le problème est qu'on ne connaît que les syndromes.

Rappelons que si d est la distance minimale du code C, alors $t = \lfloor(d-1)/2\rfloor$ est sa capacité de correction. On cherche donc l'erreur e de poids au plus t dont les coefficients $S_{i_1},\dots,S_{i_l}$ de sa transformée de Fourier sont connus.

3.3 Polynôme localisateur et identités de Newton

Si l'erreur e est de poids w, soient $j_1,\dots,j_w$ les indices des composantes non nulles de e, qui sont forcément égales à 1, car nous sommes dans le cas binaire (mots définis dur $\mathbb{F}_2$). On encode ces positions $j_1,\dots,j_w$ dans le *polynôme localisateur* $\sigma(Z)$ comme suit :

$$\sigma(Z) = \prod_{i=1}^{w}(1-\alpha^{j_i}Z) = \sum_{i=0}^{w}\sigma_i Z^i, \tag{3.2}$$

où les σ_i sont les *fonctions symétriques élémentaires* de $\alpha^{j_1},\dots,\alpha^{j_w}$, que l'on note $Z_1,\dots,Z_w$: ce sont les localisateurs de l'erreur. Le problème de décodage est alors équivalent à celui de trouver $\sigma(Z)$. À partir de $\sigma(Z)$, on trouve ses racines, puis les indices $j_1,\dots j_w$, donc e. On considère le problème résolu quand on a trouvé $\sigma(Z)$.

Soient tous les coefficients $S_0,\dots,S_{n-1}$ de la transformée de Fourier de e, alors on a les *identités de Newton* entre les coefficients S_i et les coefficients σ_i :

$$\begin{cases} S_i + \displaystyle\sum_{j=1}^{i-1}\sigma_j S_{i-j} + i\sigma_i = 0, & i \le w, \\ S_i + \displaystyle\sum_{j=1}^{w}\sigma_j S_{i-j} = 0, & i > w. \end{cases} \tag{3.3}$$

On note I_N l'idéal engendré par ces identités. C'est à dire que je considère les membres gauches des égalités précédentes comme des polynômes dans l'anneau

$$\mathbb{F}_2[\sigma_w, \ldots, \sigma_1, S_{n-1}, \ldots, S_0],$$

et I_N est l'idéal engendré par tous ces polynômes. De manière générale, j'ommettrai le membre droit « =0 » des équations, pour que ne considérer que les parties gauches qui sont des polynômes.

Nous considérons ce système comme un *système linéaire* en les σ_i, dont les coefficients sont les S_i. Plusieurs problèmes se posent, quand on utilise ces équations pour décoder un code cyclique d'ensemble de définition Q :

1. contrairement au cas de la caractéristique nulle, le système n'est pas triangulaire en les σ_i : quand i est pair, le terme $i\sigma_i$ disparaît ;
2. on ne connait pas à priori le poids $w = 1$ de l'erreur e : on ne sait donc pas comment écrire les équations ;
3. une partie seulement des S_i sont connus : les S_i pour $i \in Q$; les autres S_i, $i \notin Q$ sont inconnus.

Il faut donc éliminer du système les S_i, $i \notin Q$, et résoudre en les σ_i.

3.4 Un exemple

Je redonne le décodage fait dans [RYT90], qui peut être exposé rapidement. Il s'agit du code à résidus quadratiques de longueur 31, binaire. Son ensemble de définition est l'ensemble des carrés non nuls modulo 31 :

$$Q = \{1, 2, 4, 5, 7, 8, 9, 10, 14, 16, 18, 19, 20, 25, 28\}$$

C'est un code $[n = 31, k = 16, d = 7]_2$ qui est donc théoriquement capable de corriger $t = \lfloor (d-1)/2 \rfloor = 3$ erreurs. Soit $y = c + e$ le mot reçu, c le mot de code, que l'on cherche, et e l'erreur. On connaît les syndromes de l'erreur, S_i, $i \in Q$, et on cherche les fonctions symétriques $\sigma_1, \sigma_2, \sigma_3$ en fonction des syndromes $(S_i)_{i \in Q}$. Les identités de Newton contiennent, entre autres, les équations :

$$\begin{aligned}
S_1 + \sigma_1 &= 0 \\
\mathbf{S_3} + S_2\sigma_1 + S_1\sigma_2 + 3\sigma_3 &= 0 \\
S_5 + S_4\sigma_1 + \mathbf{S_3}\sigma_2 + S_2\sigma_3 &= 0 \\
\mathbf{S_6} + S_5\sigma_1 + S_4\sigma_2 + \mathbf{S_3}\sigma_3 &= 0 \\
S_7 + \mathbf{S_6}\sigma_1 + S_5\sigma_2 + S_4\sigma_3 &= 0 \\
S_9 + S_8\sigma_1 + S_7\sigma_2 + \mathbf{S_6}\sigma_3 &= 0 \\
S_{10} + S_9\sigma_1 + S_8\sigma_2 + S_7\sigma_3 &= 0
\end{aligned}$$

Dans ces équations les syndromes $\mathbf{S_3}$ et $\mathbf{S_6}$ ne sont pas connus, car 3 et 6 ne sont pas dans l'ensemble de définition de C. Il faut donc déterminer $\sigma_1, \sigma_2, \sigma_3$ en fonction de $S_1, S_2, S_4, S_5, S_7, S_8, S_9, S_{10}$, en éliminant $\mathbf{S_3}$ et $\mathbf{S_6}$.

Les auteurs [RYT90] établissent, à la main, les points suivants :

1. si $S_1^5 = S_5$ et $S_1^7 = S_7$, alors $\sigma_2 = \sigma_3 = 0$ et l'erreur est de poids 1, et son polynôme localisateur n'a qu'un terme $\sigma_1 = S_1$.
2. si $S_1^5 \neq S_5$ et si

$$S_1^{15} + S_1^8 S_7 + S_1^5 S_5^2 + S_1^3 S_5 S_7 + S_1 S_7^2 + S_5^3 = 0, \tag{3.4}$$

alors $\sigma_3 = 0$, l'erreur est de poids 2 et

$$\sigma_2 = \frac{S_7 + S_5 S_1^2}{S_1^5 + S_5}. \tag{3.5}$$

3. sinon l'erreur est de poids 3, et

$$\sigma_2 = \frac{S_7\left((S_7 + S_5S_1^2)S_7 + (S_5 + S_1^5)S_9\right)}{S_1^2(S_7 + S_1^7)(S_5^2 + S_1S_9 + S_1^{10})} + \frac{S_5^2 + S_1S_9}{S_1(S_7 + S_1^7)}, \tag{3.6}$$

et

$$\sigma_3 = \frac{S_5^2 + S_1S_9 + \sigma_2S_1^8}{S_7}. \tag{3.7}$$

Ces formules sont assez fastidieuses à obtenir, sujettes à erreur. Elles sont aussi dépendantes du code : ainsi les auteurs Chen, Reed, Truong et d'autres [RYT90, RYTH90, RTCY92, CRT94, LWCL95, HRTC01, CTR+03, TCCL05], construisent à chaque fois des nouvelles formules pour les codes à résidus quadratiques de longueurs 31, 23, 41, 73, 47, 71, 79, 97, et enfin 103 et 113.

Il semble naturel d'utiliser les outils du calcul formel pour automatiser l'obtention des formules. Les questions sont alors : peut-on obtenir des formules de degré 1 en les σ_i en éliminant les S_i qui ne sont pas des syndromes ? Enfin, le problème de l'efficacité se pose : il faudrait pouvoir déterminer la taille des formules.

3.5 Historique des travaux utilisant les bases de Gröbner

Il y a eu un certain nombre de travaux où les bases de Gröbner sont utilisées pour décoder les codes cycliques. On peut les classer suivant deux axes : choix de la mise en équation, décodage en un coup ou décodage en ligne.

Une autre mise en équation Il y a plusieurs manières de mettre en équation le problème du décodage des codes cycliques. Chacune de ces mises en équations relie algébriquement les σ_i aux syndromes S_i. La mise en équation historique, avec l'utilisation des bases de Gröbner remonte à Cooper [CI90, CI91b, CI91a], pour décoder les codes BCH jusqu'à leur vraie capacité de correction. La mise en équation de Cooper utilise directement la définition de la transformée de Fourier, plutôt que les identités de Newton. Elle a l'inconvénient d'avoir un degré élevé, et un grand nombre de solutions parasites. Cette mise en équation a été étudiée par Loustaunau et von York [LY97], et Caboara et Mora [CM02], pour les codes cycliques généraux, et ils ont donné des preuves correctes des affirmations de Cooper.

Décodage en un coup et décodage en ligne L principe que j'ai présenté sur l'exemple précédent est qualifié de décodage en un coup (one-step decoding) : on fait un précalcul pour déterminer symboliquement les σ_i en fonction des syndromes. On utilise ces formules en ligne, en substituant les valeurs des syndromes des mots reçus. Ce décodage a été considéré pour les codes cycliques quelconques par Chen, Helleseth, Reed et Truong [CRHT94b].

Chen, Helleseth, Reed, Truong [CRHT94d, CRHT94a] ont aussi utilisé la mise en équation de Cooper, avec les *syndromes instanciés*. Pour chaque mot reçu, on écrit le système avec les syndromes instanciés $S_i^* \in \mathbb{F}_{2^m}$, plutôt qu'avec les indéterminées S_i. On fait le calcul de base de Gröbner *en ligne*, pour chaque mot reçu. Ils montrent que l'on peut ainsi obtenir directement les coefficients du polynôme localisateur.

3.6 Contributions théoriques

Équations de corps Pour un code de longueur n, les localisateurs Z_i, qui sont des racines n-ièmes de l'unité, vérifient les équations « de corps » $X_i^{n+1} - X_i = 0$, donc $X_i^{2^m} - X_i = 0$, où $\mathbb{F}_{2^m}$ est le corps de décomposition de $X^n - 1$. Les fonctions

symétriques élémentaires et les coefficients de la transformée de Fourier, qui sont des fonctions algébriques des localisateurs, vérifient donc aussi $\sigma_i^{2^m} - \sigma_i = 0$, $S_i^{2^m} - S_i = 0$.

Tous les auteurs précédents ont toujours considéré des idéaux contenant ces équations de corps (c'est-à-dire leur termes gauche), pour chacune des variables. Premièrement cela permet de ne considérer que les solutions dans $\mathbb{F}_{2^m}$ plutôt que dans la clôture algébrique de $\mathbb{F}_2$. Deuxièmement, avec ces équations de corps, les idéaux sont radicaux et de dimension zéro, et il est plus facile de démontrer qu'on va bien obtenir des formules de degré un pour les σ_i.

L'inconvénient est que les polynômes $\sigma_i^{2^m} - \sigma_i$, $S_i^{2^m} - S_i$ peuvent être de degré élevé, même en longueur n petite. Par exemple, pour le code à résidus quadratiques de longueur 41, le corps de décomposition sur $\mathbb{F}_2$ de $X^{41} - 1$ est $\mathbb{F}_{2^{20}}$, ce qui oblige à rajouter les polynômes $\sigma_i^{2^{20}} - \sigma_i$ à l'idéal. De tels degrés ne sont pas utilisables en pratique. Il est donc tentant d'enlever les équations de corps. C'est ce que nous avons étudié, dans le cas du décodage en ligne, et dans le cas du décodage en un coup. Les difficultés rencontrées sont que les idéaux ne sont plus de dimension zéro, et que nous n'avons pas pu prouver la radicalité des ces idéaux.

Nous avons obtenu les résultats suivants.

1. Nous avons prouvé que cette approche est correcte dans le cas du décodage en ligne. Soit $S_{i_1}^*, \ldots, S_{i_l}^*$ les syndromes du mot reçu y. Alors l'idéal I_N des identités de Newton, avec instanciation des syndromes $S_i \mapsto S_i^*$, $i \in Q$, engendrent un idéal dans $\mathbb{F}_{2^m}[(S_i)_{i \notin Q}, \sigma_1, \ldots, \sigma_w]$. Nous avons montré que quand on élimine les « syndromes inconnus » S_i, $i \notin Q$, l'idéal d'élimination $I_N \cap \mathbb{F}_{2^m}[\sigma_1, \ldots, \sigma_w]$ contient les polynômes $\sigma_i - \sigma_i^*$, où les σ_i^* sont les coefficients du polynôme localisateur de l'erreur.

2. Dans le cas du décodage en un coup, lorsque les syndromes sont « formels », l'idéal I_N des relations de Newton est dans l'anneau de polynômes

$$\mathbb{F}_2[(S_i)_{i \in Q}, (S_i)_{i \notin Q}, \sigma_1, \ldots, \sigma_w].$$

   Quand on élimine les $(S_i)_{i \notin Q}$, l'idéal $I_N \cap \mathbb{F}_2[(S_i)_{i \in Q}, \sigma_1, \ldots, \sigma_w]$ contient des relations de degré 1 en les σ_i, dont les coefficients sont des polynômes en les S_i, $i \in Q$. Toutefois, on ne sait pas dire si les initiaux s'annulent quand on les spécialise en les syndromes, ce qui pose des problèmes de division pour trouver les σ_i, quand on spécialise.

3. Dans le contexte du décodage en un coup, j'ai construit une mise en équation, telle que le calcul de la base de Gröbner donne des formules du type $\sigma_j - F_j((S_i)_{i \in Q}) = 0$, voir appendice 8. Il n'y a ainsi plus de problèmes de division par zéro à la spécialisation.

Mise en équation avec les formules de Waring L'inconvénient de l'idéal I_N des identités de Newton, pour le décodage, est qu'il contient des indéterminées S_i, $i \notin Q$, qu'il faut éliminer. On peut écrire une autre mise en équation, où n'interviennent que les syndromes S_i, $i \in Q$.

En utilisant la partie triangulaire des identités de Newton, puis les identités suivantes, on voit qu'il existe des relations directes générales qui lient les S_i aux σ_i, de la forme :

$$S_i = W_i(\sigma_1, \ldots, \sigma_w), \quad i \in \{0, \ldots, n-1\}.$$

Ce sont les *formules de Waring* dont on connait une expression explicite [LN96]. Dans le contexte du décodage, on ne connait que les S_i, $i \in Q$, et on considère l'idéal

$$I_W : \{S_i - W_i(\sigma_1, \ldots, \sigma_w), \quad i \in Q\}.$$

En calculant une base de Gröbner de I_W pour l'ordre lexicographique $\sigma_w > \ldots, \sigma_1 > (S_i)_{i \in Q}$, on obtient des formules de degré 1 pour les σ_i, en fonction des S_i, $i \in Q$. On peut aussi calculer la base de Gröbner avec les syndromes instanciés, au décodage en ligne, pour chaque mot reçu. Avec Magali Bardet et Jean-Charles Faugère, nous avons considéré cette mise en équation [Bar04].

Par rapport à l'idéal des identités de Newton, cette mise en équation présente l'intérêt que les S_i, $i \notin Q$ sont déjà éliminés. En revanche, les polynômes W_i sont de degré d'autant plus élevé en $\sigma_1, \ldots, \sigma_w$ que l'indice i augmente. Dans le cas des codes à résidus quadratiques, les indices $i \in Q$, sont éparpillés dans tout $\{1, \ldots, n\}$. Il y aura donc de grands indices $i \in Q$, donc des équations de haut degré, ce qui pose un problème d'efficacité.

3.7 Contributions pratiques

Décodage en un coup Pour le décodage en un coup, les articles d'origine ne traitent que des exemples de petites longueur, par exemple $n = 16$ dans Caboara et Mora, voire 23 dans Loustaunau et von York [CM02, LY97]. La principale raison est l'emploi des équations de corps, et peut-être aussi la mauvaise efficacité des logiciels utilisés.

En utilisant son logiciel Fgb et les idéaux sans les équations de corps, Jean-Charles Faugère et moi avons pu pousser un peu les calculs. Nous avons calculé par exemple, la base de Gröbner pour le code à résidus quadratiques de longueur 41, voir figure 3.1. On voit que les calculs deviennent très vite impraticables. Il semblerait que sommes nous confrontés au théorème de Bruck et Naor [BN90], qui indique que le décodage d'un code linéaire général est difficile, même avec précalcul.

Décodage en ligne Le principe du décodage en ligne est d'instancier les syndromes du mot reçu, en faisant $S_i \mapsto S_i^*$, dans le système des identités de Newton, puis de calculer la base de Gröbner de cet idéal. En pratique, c'est beaucoup plus efficace que de calculer la base de Gröbner symbolique, puis de substituer ensuite : c'est la comparaison entre une formule explicite et un algorithme. L'algorithme est plus rapide que l'évaluation de la formule. Même plus, la formule est impossible à obtenir.

En ce qui concerne les codes BCH, Magali Bardet a ainsi pu décoder 15 erreurs par rapport au code BCH de paramètres $[n = 127, k = 43, d = 31]$, obtenu avec une distance construite de $\delta = 29$. Avec les algorithmes classiques, on décode seulement $\lfloor \delta - 1 \rfloor / 2 = 14$ erreurs. De même, en longueur 511, elle a pu décoder plusieurs codes BCH jusqu'à leur vraie distance minimale.

Magali Bardet a aussi pu décoder des codes à résidus quadratiques de moyenne longueur, voir figure 3.2.

Décodage au delà de la capacité de correction de correction du code Nous avons prouvé que l'idéal I_N des identités de Newton, relativement au décodage en ligne, est une bonne mise en équation. La propriété fondamentale pour établir l'existence de polynômes de degré 1 en les σ_i dans I_N est que la variété associée à l'idéal I_N est de cardinal 1. En termes de codage, cela signifie qu'il y a unicité du mot de code à distance w. Cette unicité se produit pour un nombre d'erreurs $w \leq t$ où t est la capacité de correction du code.

L'unicité du mot de code le plus proche peut aussi se produire pour un nombre d'erreurs $\tau > t$, pour certains mots reçus y. Dans ce cas, la variété associée à I_N, spécialisée sur les syndromes de y est de dimension 0. On aura bien des formules de degré 1 en les σ_i. Magali Bardet a ainsi décodé des codes BCH de longueur 511, pour un nombre d'erreurs supérieur de quelques unités à la capacité de correction de ces codes. Pour tous les essais effectués (10 000 décodages à la distance 52 pour le code BCH 47-correcteur), il y avait une unique solution.

$\boldsymbol{\sigma_4} S_{23} + \ldots,$
$\boldsymbol{\sigma_4} S_9 + \ldots,$
$\boldsymbol{\sigma_4} S_5 + \ldots,$
$\boldsymbol{\sigma_4} S_1 + \ldots,$
$\boldsymbol{\sigma_3} + \sigma_2 S_1 + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9^2 S_1 + \cdots + S_5^2 S_1^{32} + S_5 S_1^{37} + S_1) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_5 S_1^{13} + S_{23} S_1^{59} + \cdots + S_5^3 S_1^{26} + S_5^2 S_1^{31} + S_5 S_1^{36} + S_1^{41}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_5 + \cdots + S_9 S_1^{60} + S_9 S_1^{19}) + \ldots,$
$\boldsymbol{\sigma_2}(S_9^3 + S_9 S_5^2 S_1^8 + S_9 S_1^{59} + S_9 S_1^{18} + S_5^2 S_1^{58} + S_1^{27}) + \ldots,$
$\boldsymbol{\sigma_2}(S_9^3 S_1^{38} + \cdots + S_9 S_1^{15} + S_5^3 S_1^{50}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9 S_1^{32} + \cdots + S_5^2 S_1^{54} + S_5 S_1^{59}) + \ldots,$
$\boldsymbol{\sigma_2}(S_9^3 S_1^{33} + \cdots + S_9 S_1^{51} + S_5^3 S_1^{45} + S_5 S_1^{55} + S_1^{60}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_5 S_1^{31} + S_{23} S_1^{36} + S_9^2 S_5^2 S_1^{31} + S_9^2 + S_9 S_5^2 S_1^{40} + S_9 S_1^{50} + S_9 S_1^9 + S_5 S_1^{54}) + \ldots,$
$\boldsymbol{\sigma_2}(S_9^4 S_1^{20} + \cdots + S_5^2 S_1^5 + S_5 S_1^{51} + S_5 S_1^{10}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_1^{32} + S_9^2 S_1^{37} + \cdots + S_9 S_1^5 + S_5^2 S_1^{45} + S_5 S_1^{50} + S_1^{55}) + \ldots,$
$\boldsymbol{\sigma_2}(S_9^4 S_5 S_1^{10} + \cdots + S_9 S_5^2 S_1^{32} + S_9 S_5 S_1^{37} + S_5^2 + S_1^{51} + S_1^{10}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9^3 + S_{23} S_9 S_1^{18} + \cdots + S_9 S_5^2 S_1^{31} + S_9 S_5 S_1^{36} + S_9 S_1^{41} + S_9 + S_5^2 S_1^{40} + S_5 S_1^{45}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9 S_5 S_1^{13} + \cdots + S_9 S_1^{41} + S_9 + S_5^3 S_1^{35} + S_5 S_1^{45} + S_1^9) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9^2 S_5 + \cdots + S_5^3 S_1^{31} + S_5^2 S_1^{36} + S_5 S_1^{41} + S_5 + S_1^{46}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9^2 S_1^5 + \cdots + S_9 S_1^{37} + S_5 + S_1^{46} + S_1^5) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9 S_5 S_1^8 + \cdots + S_5^3 S_1^{30} + S_1^{45} + S_1^4) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_5 S_1^9 + \cdots + S_9 S_5 S_1^{23} + S_9 S_1^{28} + S_5^2 S_1^{27} + S_1^{37}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_9 S_1 + S_{23} S_5^2 + S_9^3 S_1^6 + S_9 S_5^2 S_1^{14} + S_9 S_1^{24} + S_1^{33}) + \ldots,$
$\boldsymbol{\sigma_2}(S_{23} S_1^{10} + \cdots + S_5 S_1^{28}) + \ldots,$
$\boldsymbol{\sigma_2}(S_9^2 S_1^2 + S_9 S_1^{11} + S_5^4 + S_5^2 S_1^{10} + S_5 S_1^{15} + S_1^{20}) + \ldots,$
$+350$ polynômes en S_{23}, S_9, S_5, S_1.

FIG. 3.1 – Base d'élimination de $\langle I_N \rangle \cap \mathbb{F}_2[\sigma_4, \ldots, \sigma_1, (S_i)_{i \in Q}]$ pour le code à résidus quadratiques [41,22,9], calculée et formatée par Magali Bardet.

longueur n	nombre d'erreurs	degré de l'extension	nombre d'opérations
71	5	$\mathbb{F}_{2^{35}}$	$2^{11,1}$
73	6	$\mathbb{F}_{2^9}$	$2^{15,3}$
89	8	$\mathbb{F}_{2^{11}}$	$2^{26,2}$
113	7	$\mathbb{F}_{2^{28}}$	2^{20}

FIG. 3.2 – Nombre d'opérations arithmétiques pour décoder des codes à résidus quadratiques binaires.

Codes « faciles » et codes « difficiles » Alors que la méthode (identités de Newton puis élimination des S_i, $i \notin Q$), est la même dans les deux cas, la différence entre les ensembles de définition des codes BCH et des codes à résidus quadratiques suffit à faire changer grandement les temps de calcul de base de Gröbner.

Les codes BCH, qui ont une très grande régularité de leur ensemble de définition, se décodent avec beaucoup moins d'opérations que les codes à résidus quadratiques, dont l'ensemble des zéros est dispersé. Ainsi, le décodage des BCH en longueur 511 ne pose pas de problème, alors que les calculs deviennent difficiles pour les codes à résidus quadratiques en longueur 113.

Jean-Charles Faugère a aussi considéré l'idée de la *trace du précalcul* : il exécute un décodage pour une erreur donnée, et il garde une trace de ce calcul, pour l'algorithme F4 [Fau99]. Cette trace évite de refaire toutes les opérations de mise en place de la matrice principale intervenant dans F4, on réutilise celle obtenue par le précalcul, en changeant ses coefficients. Cette méthode permet de gagner un ordre de grandeur (1000) par rapport au calcul direct de la base de Gröbner.

3.8 Conclusion

Les travaux précédents aux nôtres ont mis l'accent sur le décodage en un coup, en espérant éviter un calcul de base de Gröbner pour chaque mot reçu. Nous avons vu au contraire que le décodage en ligne est plus efficace, grâce à de nouvelles mises en équations, sans les équations de corps. Toutefois, nous ne sommes pas encore en mesure de donner la complexité exacte de ces calculs, et il semble que le coût dépende grandement du code utilisé.

La question que j'avais posée à Magali Bardet pour sa thèse était de savoir si les codes cycliques pouvait se décoder en temps polynomial, à l'opposé des codes linéaires généraux. Une voie pour prouver ce résultat était d'utiliser le décodage en un coup, et d'estimer la taille des formules. Cette question était trop ambitieuse : il semble difficile, de prédire la taille d'une base de Gröbner de manière réaliste. La pratique indique aussi que les formules seront grandes (explosion du décodage en un coup pour les codes à résidus quadratiques), et le théorème de Bruck et Naor, sur la difficulté du décodage des codes linéaires généraux avec précalcul, s'applique peut-être aussi pour la classe des codes cycliques.

Chapitre 4

Décodage des codes d'évaluation avec l'algorithme de Sudan

Une famille de codes présentant des exemples importants pour la théorie et la pratique est la famille des codes obtenus par *évaluation*. Soit X un ensemble. Cette construction générale consiste à évaluer un certain ensemble L de fonctions

$$f : X \to \mathbb{F}_q,$$

sur des points distincts $P_1, \ldots, P_n \in X$. On obtient une *fonction d'évaluation* :

$$\begin{array}{rccc} \mathrm{ev} : & L & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(P_1), \ldots, f(P_n)) \end{array},$$

et le *code d'évaluation* obtenu est $C = \mathrm{ev}(L)$, l'image par ev de L.

Au premier plan des codes d'évaluation, on trouve les *codes de Reed-Solomon*, qui sont parmi les codes les plus utilisés, et qui de plus sont optimaux par rapport à la *borne de Singleton* sur la distance minimale d'un code. Ils sont obtenus en prenant comme espace L une famille de polynômes univariés sur $\mathbb{F}_q$. Les codes de Reed-Solomon se généralisent avec les *codes géométriques*, qui utilisent la théorie des courbes algébriques sur un corps fini.

La deuxième famille est celle de *codes de Reed-Muller*, où les points $P_1, \ldots, P_n$ sont dans l'espace affine $\mathbb{F}_q^m$, et l'espace L des polynômes à évaluer est l'espace des polynômes à m variables sur $\mathbb{F}_q$ de degré inférieur à r. On s'intéresse tout particulièrement au cas $r = 1$: l'espace L est l'ensemble des applications affines sur $\mathbb{F}_q$, et on obtient le code de Reed-Muller d'ordre 1.

J'ai encadré deux thèses sur le décodage de ces deux familles :

- Lancelot Pecquet [Pec01] a travaillé sur les codes de Reed-Solomon et sur les codes géométriques, qui se décodent avec l'algorithme de Guruswami-Sudan [GS99].
- Cédric Tavernier [Tav04] a étudié le décodage des codes de Reed-Muller d'ordre 1, en se basant sur l'algorithme de Goldreich, Rubinfeld et Sudan [GRS95].

Les deux algorithmes sont des algorithmes de *décodage en liste* : on décode plus d'erreurs que la capacité de correction classique, dans les deux cas (Reed-Solomon et Reed-Muller), mais on perd l'unicité de la solution. En revanche, on gagne beaucoup en capacité de correction, comme nous le verrons.

4.1 Trois exemples de codes d'évaluation

Les codes de Reed-Solomon Ces codes se contruisent de la manière suivante. On se donne $x_1, \ldots, x_n \in \mathbb{F}_q$, tous distincts (et donc $n \leq q$). Soit ev la fonction d'évaluation

$$\begin{array}{rccc} \text{ev}: & \mathbb{F}_q[X] & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(x_1), \ldots, f(x_n)) \end{array} .$$

Soit L_k l'espace des polynômes de degré inférieur à k, $k < n$:

$$L_k = \{f(X) \in \mathbb{F}_q[X];\ \deg f(X) < k\}$$

Le *code de Reed-Solomon* de dimension k est :

$$\text{RS}_k = \{\text{ev}\, f(X); \quad f(X) \in L_k\}\,.$$

C'est un code $[n, k, n-k+1]_q$. Bien qu'optimaux en termes de distance minimale, le défaut de ces codes est que leur longueur est bornée par q. En pratique et pour fixer les idées, un des codes de Reed-Solomon le plus utilisé est le code $[n = 255, k = 239, d = 17]_{256}$.

Les codes géométriques Ils servent à remédier au principal problème des codes de Reed-Solomon, qui est d'avoir une longueur bornée par la taille de l'alphabet $\mathbb{F}_q$. La construction historique est due à Goppa [Gop77, Gop81].

Soit $\mathcal{X}$ une courbe lisse projective définie sur $\mathbb{F}_q$, soient $P_1, \ldots, P_n$, n points $\mathbb{F}_q$-rationnels de $\mathcal{X}$, et un espace $L(D)$ associé à un diviseur D sur la courbe. On suppose qu'aucune fonction de $L(D)$ n'a de pôle en un des P_i. Alors on définit la fonction d'évaluation :

$$\begin{array}{rccc} \text{ev}: & L(D) & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(P_1), \ldots, f(P_n)) \end{array} .$$

Pour raccourcir la notation, on note P le diviseur $P_1 + \cdots + P_n$. Le code géométrique $\Gamma(P, D)$ est

$$\Gamma(P, D) = \text{ev}\, L(D).$$

Si g est le genre de la courbe, et $\deg D$ le degré du diviseur D, la dimension k de $\Gamma(P, D)$ est supérieure ou égale à $\deg D - g + 1$ (Théorème de Riemman-Roch), et sa distance minimale supérieure ou égale à $n - \deg D$. On obtient des codes longs en prenant des courbes ayant un grand nombre de points sur $\mathbb{F}_q$.

En pratique, il est difficile de construire des bases des espaces $L(D)$ pour des diviseurs D quelconques sur des courbes quelconques. La classe de codes géométriques la plus en vue pour les applications est celle des *codes hermitiens*. Ils existent quand $q = r^2$, et ils sont définis en utilisant la *courbe hermitienne*, dont l'équation plane affine est :

$$y^r + y = x^{r+1}.$$

Pour cette courbe, on peut expliciter complétement la construction de Goppa, et les *codes hermitiens* sont très bien connus [Sti93]. La courbe hermitienne a r^3 points affines $\mathbb{F}_q$-rationnels, et une place à l'infini P_∞. Son genre est $r(r-1)/2$. En prenant $D = kP_\infty$, on sait décrire explicitement une base de chaque espace $L(kP_\infty)$, pour k variable [Sti93]. Muni de cette base, on sait construire la fonction d'évaluation correspondante. Les codes hermitiens sont si bien connus qu'ils sont exposés de manière élémentaire (sans la théorie des courbes algébriques) dans [JH04].

Pour les comparer aux codes de Reed-Solomon, considérons le cas où l'alphabet $\mathbb{F}_q$ est de taille $16 = 4^2$. Le nombre de points affines est $4^3 = 64$, le genre est

$(4 \cdot 3)/2 = 6$. Pour les valeurs intéressantes de k, en considérant l'espace $L(kP_\infty)$, on obtient une famille de codes du type

$$[n = 64, \geq k - 6, \geq 64 - k + 1]_{16}.$$

Les codes de Reed-Solomon bâtis sur le même alphabet auront des paramètres du type

$$[n = 16, k, 16 - k + 1]_{16},$$

le code hermitien est donc sensiblement plus long, avec une petite perte sur la dimension.

Les codes de Reed-Muller Ils peuvent être vus comme une généralisation multivariées des codes de Reed-Solomon. Au contraire des codes de Reed-Solomon, ce sont des codes très longs pour un petit alphabet, par exemple pour l'alphabet binaire $\mathbb{F}_2$. On se donne n points distincts $P_1, \ldots, P_n \in \mathbb{F}_q^m$, et l'espace de polynômes :

$$L_r = \{f \in \mathbb{F}_q[X_1, \ldots, X_m],\ \deg f \leq r\}.$$

La fonction d'évaluation correspondante est :

$$\begin{array}{rccc} \text{ev}: & L_r & \to & \mathbb{F}_q^n \\ & f & \mapsto & (f(P_1), \ldots, f(P_n)) \end{array}.$$

Dans le cas binaire, en prenant $n = 2^m$, et $\{P_1, \ldots, P_n\}$ tous les éléments de $\mathbb{F}_2^m$, on obtient un code de longueur 2^m, de dimension $\sum_{i=0}^{r} \binom{m}{i}$, et de distance minimale 2^{m-r}. Le cas le plus simple est le code de Reed-Muller à m variables d'ordre 1, obtenu pour $r = 1$. C'est un code $[n = 2^m, k = m + 1, d = 2^{m-1}]_2$. Ces codes ne sont pas très bons en termes de codage (trop petite dimension), mais ils sont très importants en cryptographie, et ont été utilisés historiquement (photographies de Mars par la sonde Mariner 9, en 1971).

4.2 Problème du décodage des codes d'évaluation et approximation par des polynômes

Dans le cas des codes d'évaluation considérés dans ce chapître, il y a un lien très fort entre décodage et approximation. Je présente ce lien pour le cas simple des codes de Reed-Solomon.

Soit RS_k le code de Reed-Solomon de dimension k, construit avec n éléments distincts $x_1, \ldots, x_n \in \mathbb{F}_q$. Soit $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$ le mot reçu. Décoder τ erreurs, c'est trouver le mot de code $c \in \text{RS}_k$, tel que

$$d(c, y) \leq \tau,$$

où d est la distance de Hamming. Soit $f(X)$ le polynôme tel que $c = \text{ev}\, f(X)$. Alors le problème de décoder y devient celui de trouver les polynômes $f(X)$, $\deg f(X) < k$, tels que :

$$|\{i,\ f(x_i) = y_i\}| \geq n - \tau.$$

Si on cherche le polynôme d'interpolation de Lagrange $p(X)$ tel que $p(x_i) = y_i$, $i \in \{1, \ldots, n\}$, alors le degré de $p(X)$ est typiquement $n - 1$. Dans le contexte qui nous intéresse, on cherche un polynôme $f(X)$ de petit degré (borné par k), et on tolère que la courbe $Y - f(X)$ ne passe pas par tous les points (x_i, y_i).

On peut aussi voir le problème comme le problème d'approximation suivant.

> Soit n points $(x_i, y_i) \in \mathbb{F}_q^2$, vus comme une fonction $x_i \mapsto y_i$, trouver les polynômes de bas degré approchant au mieux cette fonction, pour la métrique de Hamming.

Cette problématique se généralise au cas des codes de Reed-Muller. Les polynômes univariés sont remplacés par des polynômes à plusieurs variables de petit degré. Étant donnés n points $P_1, \ldots, P_n \in F_q^m$, et n valeurs $y_1, \ldots, y_n \in \mathbb{F}_q$, on cherche le polynôme à m variables de degré au plus r qui interpole au mieux les points $(P_i, y_i) \in \mathbb{F}_q^{m+1}$, pour la distance de Hamming.

4.3 Le schéma de principe des algorithmes de Sudan et Guruswami

Pour présenter mes contributions, je dois donner le principe des algorithmes de décodage en liste des codes d'évaluation, sur le cas simple des codes de Reed-Solomon. Ces algorithmes forment toute une famille, qui va du cas le plus simple (algorithme de Sudan), au plus élaboré (algorithme de Kötter et Vardy), en passant par l'algorithme de Guruswami-Sudan, qui est une étape intermédiaire essentielle entre ces deux extrêmes.

J'ai besoin de rappeler la définition du (u, v)-degré pondéré d'un polynôme bivarié. C'est le maximum des (u, v)-degrés pondérés de ses monômes, où le (u, v)-degré pondéré du monôme $X^i Y^j$ est $ui + vj$. Je note $\mathrm{wdeg}_{u,v} Q(X, Y)$ le (u, v)-degré pondéré du polynôme $Q(X, Y)$.

L'algorithme de Sudan Pour décoder $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$ dans le code de Reed-Solomon, de dimension k, défini sur le support $x_1, \ldots, x_n$, je rappelle qu'il faut trouver $f(X)$ de degré inférieur à k, tel que $d(\mathrm{ev}\, f(X), y)) \leq \tau$.

L'idée fondatrice de Sudan est de construire un *polynôme d'interpolation*, $Q(X, Y) \in \mathbb{F}_q[X, Y]$, tel que :

1. $Q(X, Y) \neq 0$ (non trivialité) ;
2. $Q(x_i, y_i) = 0$ pour i de 1 à n (interpolation) ;
3. $\mathrm{wdeg}_{1,k-1} Q(X, Y) < n - \tau$ (majoration du degré pondéré).

Alors, on montre que si $f(X)$ est solution du problème de décodage, on a $Q(X, f(X)) = 0$, ou encore $Y - f(X) | Q(X, Y)$.

Ainsi l'algorithme de Susan comporte deux étapes :

Interpolation : trouver $Q(X, Y)$ tels que les conditions 1. 2. et 3. sont satisfaites ;

Recherche de racines : trouver les facteurs $Y - f(X)$ de $Q(X, Y)$.

Ces deux étapes n'ont pas été précisées plus profondément par Sudan, qui s'est contenté de remarquer qu'elles pouvaient se faire en temps polynomial en la longueur n du code.

Pour analyser le nombre d'erreurs que peut corriger cet algorithme, il reste à déterminer à quelles conditions le polynôme d'interpolation $Q(X, Y)$ existe.

Les conditions d'interpolation définissent un système d'équations linéaires sur les coefficients de $Q(X, Y)$. Pour qu'il existe une solution non nulle, il suffit d'avoir plus d'inconnues que d'équations. Le nombre d'équations est le nombre de conditions d'interpolation qui est n, et le nombre de termes N_Q de Q est donné par la condition 3 ci-dessus, de majoration du degré pondéré. Les calculs donnent :

$$\tau \leq n - \sqrt{2(k-1)n}. \tag{4.1}$$

Ce rayon de décodage est supérieur au rayon classique $t = \lfloor \frac{n-k}{2} \rfloor$, quand $k/n \leq 0,172$.

L'algorithme de Guruswami-Sudan L'amélioration de Guruswami-Sudan permet d'obtenir un rayon de décodage toujours supérieur au rayon classique $\lfloor \frac{n-k}{2} \rfloor$, pour tout taux de transmission $R = k/n \in [0, 1]$. On se donne deux paramètres auxiliaires d (un degré) et s (un ordre de multiplicité), et on cherche un polynôme $Q(X, Y)$ d'interpolation, tel que :

1. $Q(X, Y) \neq 0$ (non trivialité) ;
2. $Q(x_i, y_i) = 0$, *avec multiplicité* s, pour i de 1 à n (interpolation) ;
3. $\mathrm{wdeg}_{1,k-1} Q(X, Y) < d$ (majoration du degré pondéré).

Alors, en faisant la même analyse que précédemment, et en optimisant le degré d, on trouve

$$\tau \leq n - \sqrt{(k-1)n\left(1 + \frac{1}{s}\right)}, \tag{4.2}$$

qui tend vers

$$n - \sqrt{(k-1)n}$$

quand s croît. En prenant le rayon relatif $\frac{\tau}{n}$ des sphères de décodage, et le taux de transmission $R \approx \frac{k}{n}$, la formule (4.1) donne

$$\frac{\tau}{n} \leq 1 - \sqrt{2R}, \tag{4.3}$$

alors que la formule (4.2) donne

$$\frac{\tau}{n} \leq 1 - \sqrt{R}. \tag{4.4}$$

La rayon classique de décodage unique est donné par

$$\frac{t}{n} \leq \frac{1-R}{2}. \tag{4.5}$$

Ces trois rayons de décodage sont tracés sur la figure 4.1. On voit que l'algorithme de Guruswami-Sudan est toujours supérieur, en termes de capacité de correction, à l'algorithme classique.

4.4 Généralisations

Ces algorithmes se généralisent aux codes géométriques et aux codes de Reed-Muller. Mais aussi ils se généralisent à d'autres canaux de transmission que le canal q-aire symétrique, seul considéré jusqu'ici. Ce sont des canaux où le dispositif de réception des messages donne une information de qualité sur chaque symbole possible. Le problème de décoder en utilisant cette information de qualité s'appelle le *décodage souple* des codes de Reed-Solomon.

Généralisation à d'autres codes Ce principe de décodage se généralise bien aux codes géométriques. Cette généralisation a d'abord été faite par Shokrollahi et Wasserman [SW99], pour l'algorithme de Sudan, sans multiplicités. Ensuite Guruswami-Sudan ont, dans une seule publication [GS99], introduit la notion de multiplicités, à la fois pour les codes de Reed-Solomon et pour la classe des codes géométriques dits *à un point*. Ce sont des codes où l'espace $L(D)$ des fonctions à évaluer est de la forme $L(kP)$: ce sont les fonctions qui n'ont pas de pôles en dehors du point P, et elles ont un pôle au plus d'ordre k au point P. Les fonctions dans $L(kP)$ sont

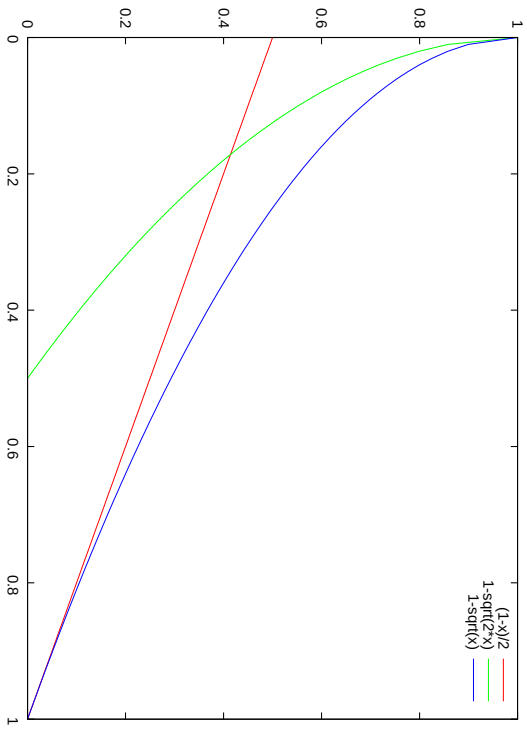

FIG. 4.1 – Comparaison des rayons de décodage de l'algorithme classique, de Sudan et de Guruswami-Sudan.

assez faciles à manier : l'ordre de leur pôle en P joue le rôle du degré des polynômes univariés, et elles ont au plus k zéros sur la courbe.

Sous ma direction, Lancelot Pecquet a dans sa thèse [Pec01] formulé l'algorithme pour les codes géométriques où l'espace $L(D)$ est associé à un diviseur D de forme générale. Ces codes ont quelquefois de meilleurs paramètres que les codes à un point [Mat01].

J'ai aussi étudié la généralisation de cet algorithme au codes de Reed-Muller. La difficulté est de passer des polynômes univariés aux polynômes multivariés. Bien que la généralisation de l'algorithme soit claire, l'optimisation des paramètres auxiliaires s (la multiplicité) et d (le degré) n'est pas immédiate. Ce problème a déjà été traité par Pellikaan et Wu [PW04a, PW04b], avec différentes analyses, notamment avec la théorie des bases de Gröbner. Je donne une nouvelle analyse, reposant sur le lemme de Schwartz-Zippel [Sch80, Zip79], qui borne le nombre de zéros d'un polynôme multivarié sur un corps fini. L'analyse que j'ai faite conduit à un meilleur rayon de décodage [AEKM$^+$06].

Généralisation à d'autres canaux Dans certains schémas de communication, le dispositif récepteur donne pour chaque symbôle une liste des symboles les plus plausibles, en attribuant à chacun une *information de fiabilité*, qui est une probabilité. Un *décodeur souple* est un décodeur qui prend en compte cette information, et retourne le mot de code le plus probable.

Kötter et Vardy [VK00, KV03] ont montré comment transformer cette information de fiabilité en information algébrique, pour décoder les codes de Reed-Solomon, et les codes géométriques. L'algorithme obtenu est de complexité polynomiale en la longueur : c'est le premier décodeur souple des codes de Reed-Solomon de complexité non exponentielle.

Plus précisément, pour chaque position x_i le récepteur propose une liste de symboles $y_{i1}, \ldots, y_{il} \in \mathbb{F}_q$ les plus probables, avec des coefficient de fiabilité $\mu_{i1}, \ldots, \mu_{il}$. Alors on peut associer à ces coefficients de fiabilités des multiplicités $s_{i1}, \ldots, s_{il}$. Cette association doit se faire de manière cohérente par rapport au problème du décodage. Une fois ces multiplicités s_{ij} trouvées, le problème d'interpolation est alors de trouver $Q(X, Y)$ tel que :

1. $Q(X, Y) \neq 0$ (non trivialité) ;
2. $Q(x_i, y_{ij}) = 0$, *avec multiplicité* s_{ij}, pour i de 1 à n, pour j de 1 à l (interpolation) ;
3. $\mathrm{wdeg}_{1,k-1} Q(X, Y) < d$ (majoration du degré pondéré).

Si les multiplicités sont bien choisies, alors parmi les polynômes $f(X)$ tels que $Q(X, f(X)) = 0$, on trouvera les $f(X)$ tels que $\mathrm{ev}\, f(X) = c$, pour les mots de code c les plus vraisemblables.

Lancelot Pecquet a proposé dans sa thèse une méthode différente de celle de Kötter et Vardy pour assigner des multiplicités en fonction des informations de fiabilité. La difficulté est de déterminer une association cohérente de multiplicités aux informations de qualité. En fait, il est même surprenant qu'une information de nature probabiliste (la qualité des symboles reçus) puisse être traduite en information algébrique (multiplicités d'interpolation).

4.5 Implémentation

Ces algorithmes comportent tous les deux étapes suivantes dont la formulation précise peut éventuellement varier suivant les codes et les canaux :

interpolation : on cherche un polynôme en une variable, à coefficients des fonctions ou des polynômes, qui passe par un ensemble de points, avec certaines multiplicités.

recherche de racines : On cherche les racines (qui sont des polynômes ou des fonctions) de ce polynôme.

Avec Lancelot Pecquet nous avons contribué à la recherche de racines, mais seulement dans le cas de l'algorithme de Sudan simple, sans multiplicités [AP98a, AP00b], dans le cas des codes de Reed-Solomon, et dans le cas des codes géométriques. Lancelot Pecquet a ensuite traité le cas avec multiplicités dans sa thèse.

Notre contribution est une adaptation de la méthode de Newton pour approcher les racines d'une équation. Classiquement, pour trouver les solutions $f(X)$ de $Q(X, f(X)) = 0$, on cherche d'abord les racines f_0 de $Q(0, Y)$. Ces racines servent ensuite de point de départ pour construire le développement de $f(X)$, en utilisation des itérations de Newton.

Mais dans le contexte très précis de l'algorithme de Sudan, nous avons montré comment obtenir les points de départ f_0 sans utiliser la factorisation univarié de $Q(X, 0)$. On évite donc la recherche de racines d'un polynôme univarié sur un corps fini. Cela rend la recherche de racines pour Sudan complétement déterministe, et ainsi l'algorithme de Sudan est déterministe dans sa globalité. C'est aussi l'algorithme le plus efficace en pratique, grâce à la rapidité de convergence de la méthode de Newton : il suffit de $\log_2 k$ itérations pour trouver une racine de degré k de $Q(X, Y)$.

En revanche, en présence de multiplicités, la contribution de Lancelot Pecquet, ainsi que toutes celles publiées, nécessite une recherche de racines d'un polynôme univarié sur un corps fini.

En pratique, j'ai implémenté l'algorithme de Sudan en C pour les codes de Reed-Solomon, et en Aldor (anciennement Axiom) pour les codes géométriques les plus simples, les codes hermitiens [Sti93]. L'étape la plus coûteuse en temps est l'étape d'interpolation, alors que le problème et l'algorithme associés sont conceptuellement plus simples que pour l'étape de recherche de racines. La plupart des auteurs considèrent que c'est cette étape qui est la plus coûteuse, et de nombreuses approches ont été proposées. Voir appendice 9.

4.6 Décodage local des codes de Reed-Muller d'ordre 1

Nous allons décrire une problématique très proche du décodage, mais qui se formule en termes « fonctionnels ». Ce problème a été le problème central de la thèse de Cédric Tavernier, sous ma direction [Tav04].

On suppose qu'on a accès à une fonction g « boîte noire »

$$g : \mathbb{F}_2^m \mapsto \mathbb{F}_2.$$

Le modèle de représentation de g est le suivant : on lui soumet $x \in \mathbb{F}_2^m$, et on reçoit en réponse $g(x)$. On cherche la fonction affine $f : \mathbb{F}_2^m \to \mathbb{F}_2$ qui coïncide au mieux avec g, c'est-à -dire telle que la probabilité

$$\Pr_{x \in \mathbb{F}_2^m} (f(x) = g(x))$$

est maximale.

Si on écrit tout le vecteur des valeurs de g, avec la fonction d'évaluation des codes de Reed-Muller :

$$g(x)_{x\in\mathbb{F}_{2^m}} = \operatorname{ev} g,$$

alors c'est exactement le problème de décodage $\operatorname{ev} g$ dans le code de Reed-Muller d'ordre 1. En effet f doit être telle que $d(\operatorname{ev} f, \operatorname{ev} g)$ est minimale, et $\operatorname{ev} f$ est dans le code de Reed-Muller, d'ordre 1.

Mais il faut penser à des applications où le nombre m de variables est grand, par exemple, $m = 64$. En prenant la fonction d'évaluation définie pour les Reed-Muller, on obtient un vecteur de longueur $n = 2^{64}$ qui est impossible à manipuler.

Un *décodeur local* est un algorithme qui fait un certain nombre de requêtes $x \mapsto g(x)$, et qui doit retourner la liste des fonctions f les plus proches. Comme le décodeur local ne dispose pas de la totalité du vecteur des valeurs $\operatorname{ev} g$, il y a une *probabilité d'échec*. Cette probabilité d'échec est la somme de la probabilité de ne pas retourner une des fonctions les plus proches, et de la probabilité de retourner une fonction trop éloignée (probabilité de non détection plus probabilité de fausse alarme).

Le décodeur local doit être de faible complexité, faire un petit nombre de requêtes $x \mapsto g(x)$, et avoir une faible probabilité d'échec.

Cédric Tavernier, se reposant sur les travaux de Goldreich, Rubinfeld et Sudan [GRS95], a construit un algorithme de grande efficacité pour résoudre ce problème. Le nombre de requêtes à l'oracle est $O(u/\varepsilon^2)$, la probabilité d'échec est de l'ordre de 2^{-u}, où u est un paramètre intervenant de manière linéaire dans la complexité qui est

$$O(u\frac{m}{\varepsilon^2}),$$

où ε est tel que

$$\Pr_{x\in\mathbb{F}_2^m}(f(x) = g(x)) = \frac{1}{2} - \varepsilon.$$

La limite du taux d'erreur admissible est $1/2$, et ici ε est un paramètre d'entrée de l'algorithme.

4.7 Application en cryptanalyse

Approximation par des polynômes univariés Thomas Jakobsen [Jak98] a montré comment utiliser l'algorithme de Sudan simple pour trouver des approximations univariées simples de l'algorithme de chiffrement de Knudsen et Nyberg [NK93, NK95].

Le principe est le suivant : on considère l'algorithme de chiffrement comme une boîte noire g, dont les entrées sont les messages clairs (ou une partie des messages clairs), et les sorties les messages chiffrés (ou une partie des messages chiffrés). On considère un certain nombre n de messages clairs comme des éléments $x_1, \ldots, x_n$ dans un corps fini $\mathbb{F}_{2^m}$, et on écrit le vecteur

$$g(x_1), \ldots, g(x_n).$$

On cherche le polynôme de petit degré qui approche au mieux ces valeurs, pour la distance de Hamming. Jakobsen a montré comment utiliser l'algorithme de Sudan pour trouver ces approximations.

De telles approximations sont ensuite utilisées pour faire une *cryptanalyse*, c'est-à -dire pour trouver une faiblesse dans l'algorithme de chiffrement, permettant de retrouver la clé au bout du compte. Je ne décrirai pas en détails la cryptanalyse, et je me contente d'étudier l'étape de recherche d'approximations de bas degré. Cette

méthode, mise au point par Jakobsen, a bien fonctionné pour le système de Knudsen et Nyberg, car celui-ci est spécifié en termes d'opérations simples sur les corps finis, légèrement perturbées pour briser la régularité.

J'ai essayé d'utiliser mon implémentation en C de l'algorithme de Sudan pour cryptanalyser un système de chiffrement à clé secrète, dans le cas d'une expertise d'algorithme de chiffrement développé en interne à Canal+.

Je n'ai pas trouvé de bonnes approximations pour l'algorithme de Canal+. Pour l'expliquer, il faut voir que la méthode de Jakobsen est adaptée pour des systèmes de chiffrement qui s'expriment naturellement en termes de corps finis, et d'opérations polynomiales univariés sur ces corps finis, comme le système de chiffrement de Knudsen et Nyberg.

En revanche, ce n'était pas le cas de l'algorithme de Canal+ que j'ai analysé, qui est décrit en termes d'opérations bit à bit, qui ne trouvent pas d'expression simple en termes de corps fini. C'est le cas en général de la plupart des algorithmes de chiffrement disponibles.

Il faut donc se tourner vers des fonctions qui « collent mieux au problème » : les polynômes multivariés sur $\mathbb{F}_2$, de bas degré. Ces polynômes s'évaluent sur $\mathbb{F}_2$, ce qui correspond mieux aux manipulations bit à bit.

Approximation par des polynômes multivariés Cédric Tavernier a utilisé l'algorithme de sa thèse pour trouver automatiquent des approximations linéaires des fonctions de sortie de l'algorithme de chiffrement DES. De telles approximations sont ensuite utilisées pour mener une cryptanalyse linéaire, comme l'a fait Matsui [Mat94b, Mat94a]. Je ne m'étendrai pas sur la totalité de la cryptanalyse linéaire, mais uniquement sur l'étape préalable de recherche d'approximations.

On considère les entrées de l'algorithme de chiffrement DES (clé et messages clairs) comme vecteurs de m bits, et on considère un bit de sortie du DES. On obtient ainsi une fonction $g : \mathbb{F}_2^m \to \mathbb{F}_2$ qui est représentée par un algorithme : on lui soumet x et on obtient $g(x)$.

Cédric Tavernier a retrouvé automatiquement avec son algorithme l'approximation linéaire du DES trouvée « à la main » par Matsui. Cédric Tavernier a aussi trouvé des meilleures approximations que celle de Matsui, pour des versions réduites du DES.

Il a de plus trouvé de nombreuses approximations, de l'ordre d'une centaine, pour certaines versions réduites. Le fait d'avoir plusieurs approximations plutôt qu'une conduit à une amélioration de la cryptanalyse totale.

4.8 Conclusion

Les codes définis par interpolation permettent de corriger beaucoup d'erreurs quand on utilise le décodage en liste, dont la meilleure réalisation est due à Sudan et Guruswami. Cette percée est aussi fondamentale car elle se généralise bien à d'autres codes, dont la classe importante des codes géométriques. De plus c'est le premier exemple de décodage en liste efficace, dont on ne connait pas de réalisation pour d'autres classes de codes.

Elle se généralise aussi bien à d'autres types de canaux de communications, comme l'ont montré Kötter et Vardy. Ceux-ci ont ainsi réalisé une percée en introduisant le premier algorithme de décodage souple des codes de Reed-Solomon.

Ces algorithmes à haut pouvoir de correction peuvent aussi être utilisés en cryptanalyse. La méthode est d'approcher les fonctions de sorties des algorithmes de chiffrement par des fonctions plus simples : les polynômes univariés ou multivariés de bas degré. C'est grâce à leur fort pouvoir de correction que ces algorithmes trouvent des approximations des fonctions de chiffrement. En effet, les fonctions de chiffrement sont en particulier conçues pour être loin des polynômes (univariés

ou multivariés), et les algorithmes de décodage classiques ne peuvent pas décoder autant d'erreurs.

Le principal problème qui reste à régler est celui de fournir une implémentation efficace des algorithmes de Sudan et Guruswami. Ceux-ci ne sont en effet pas assez rapides pour être utilisés en pratique, bien qu'une implémentation matérielle ait déjà été proposée [AKS04].

Deuxième partie

Cryptographie

Chapitre 5

Utilisation de problèmes de codage en cryptographie

En cryptographie à clé publique, on utilise des problèmes algorithmiques difficiles, pour construire des *primitives cryptographiques*, comme le chiffrement ou le hachage. Les problèmes mathématiques les plus utilisés en standard sont fondés sur des problèmes de la théorie des nombres (logarithme discret pour Diffie-Hellman, factorisation pour RSA [DH76, RSA78]).

Il est souhaitable de bâtir des primitives cryptographiques sur d'autres problématiques que celles-ci, pour différentes raisons. Premièrement, suivant la problématique utilisée, les cryptosystèmes peuvent avoir des caractéristiques différentes et intéressantes (longueur de la clé, taille des messages, rapidité de chiffrement, de déchiffrement). Deuxièmement, on peut s'inquiéter de faire reposer la sécurité des systèmes cryptographiques sur un socle si étroit : tout progrès algorithmique dans ces domaines met en danger le système. Il y a aussi le spectre de l'ordinateur quantique : si celui-ci voit le jour, alors la plupart des primitives cryptographiques s'effondrent.

Un certain nombre de systèmes reposent sur des problèmes difficiles de la théorie des codes, à commencer par le système de McEliece [McE78], qui est presque aussi ancien que RSA. Ce système est fondé sur la difficulté de décoder un code linéaire binaire quelconque. À ce jour, aucun algorithme efficace ne permet de résoudre ce problème, que ce soit en classique ou en quantique. Le cryptosystème de McEliece présente des caractéristiques différentes des autres systèmes, avec des avantages (rapidité, caractère exponentiel des attaques) et des inconvénients (principalement la taille de la clé publique).

J'ai proposé, avec Matthieu Finiasz [AF03], un système de chiffrement qui repose sur la difficulté du décodage des codes de Reed-Solomon, qui présente l'avantage d'avoir des clés bien plus courtes que McEliece. Sa sécurité est hélas mal fondée, et il a été « cassé ».

Ensuite, avec Matthieu Finiasz et Nicolas Sendrier, j'ai proposé une fonction de hachage basée sur le décodage de codes aléatoires [AFS03, AFS05a, AFS05b, AFS05c]. À l'opposé du système précédent, notre proposition a une sécurité bien établie. Curieusement, aucune proposition de fonction de hachage reposant sur les codes n'a précédé la notre, alors que la primitive du hachage est une des plus simples à obtenir.

Dans les deux cas, en plus de propositions théoriques, nous avons proposé des jeux de paramètres concrets. Pour cela, il faut bien analyser les meilleurs algorithmes pour résoudre les problèmes, ce que nous avons fait avec Nicolas Sendrier et Matthieu Finiasz, dans les deux cas.

5.1 Chiffrement fondé sur le décodage des codes de Reed-Solomon

La difficulté du problème Nous avons vu que le problème du décodage des codes linéaires binaires généraux est un problème difficile, tant du point théorique que pratique [BMvT78]. Il est naturel de se poser la question pour la classe beaucoup plus restreinte des codes de Reed-Solomon. Guruswami et Vardy ont exactement montré que ce problème est difficile [GV05]. La formulation du problème pour les codes de Reed-Solomon se précise comme suit.

> Étant donnés $x_1, \ldots, x_n \in \mathbb{F}_q$, $y = (y_1, \ldots, y_n) \in \mathbb{F}_q$, $k, \tau \in \mathbb{N}$, trouver $f \in \mathbb{F}_q[X]$, $\deg f \leq k$, telle que
>
> $$|\{i \in \{1, \ldots, n\} \mid f(x_i) \neq y_i\}| \leq \tau.$$

Guruswami et Sudan ont montré que le problème décisionnel associé est NP-complet. Cela peut paraître surprenant, car les algorithmes de décodage par interpolation que nous avons étudiés permettent de décoder jusqu'à $n - \sqrt{kn}$ erreurs. En fait, le problème devient difficile lorsque l'on essaye de décoder un grand nombre d'erreurs, significativement au delà de ce rayon de décodage de Guruswami et Sudan, voir la figure 5.1.0.0.0.

Avec Matthieu Finiasz, nous avons étudié ce problème [AF03] du point de vue pratique, et, empiriquement, les meilleurs algorithmes sont bien exponentiels en la longueur du code.

Citons aussi les travaux de Kiayias et Yung, qui ont étudié le problème du décodage des codes de Reed-Solomon, aussi en vue de l'utiliser en cryptographie [KY02b, KY02a]. Sous des hypothèses calculatoires faibles, ils montrent, entre autres, qu'il est difficile de distinguer une instance ayant une solution d'une instance aléatoire. Dans notre contexte, cela signifie que la donnée de $c + E$ ne révéle aucune information sur c et E.

Le cryptosystème Augot-Finiasz Avec Matthieu Finiasz [AF03], nous avons proposé un système de chiffrement à clé publique, fondé sur la difficulté du décodage des codes de Reed-Solomon.

Le principe est très similaire au cryptosystème de McEliece, que je rappelle. On considère un code C, pour lequel un algorithme de décodage secret existe. Pour chiffrer un message clair considéré comme un mot $c \in C$, on se contente de lui rajouter une erreur aléatoire e, et on transmet $y = c + e$. À la réception, le destinataire connaît l'algorithme de décodage secret associé au code C, et peut recouvrir c et e à partir de y.

Le système original de McEliece utilise des codes de Goppa classiques, dont la structure est cachée par une permutation secrète des positions. Nous avons proposé d'utiliser des codes basés sur les codes de Reed-Solomon, en cachant leur algorithme de décodage.

On considère C un code qui est la somme directe d'un code de Reed-Solomon et de la droite engendrée par un mot : $C = \mathrm{RS}_k \oplus E$, où E est une erreur de poids élévée, et RS_k le code de Reed-Solomon de dimension k. Il est facile de construire un algorithme de décodage de C, lorsque l'erreur E est connue. On peut alors construire un cryptosystème similaire à celui de McEliece.

Pour spécifier le code C sans révéler E, il suffit de donner $P = c + E$, où $c \in \mathrm{RS}_k$ est pris au hasard. En effet le code RS_k est considéré comme une donnée constante à tous les utilisateurs. Retrouver c et E à partir de P est exactement le problème du décodage des codes de Reed-Solomon, qui est difficile.

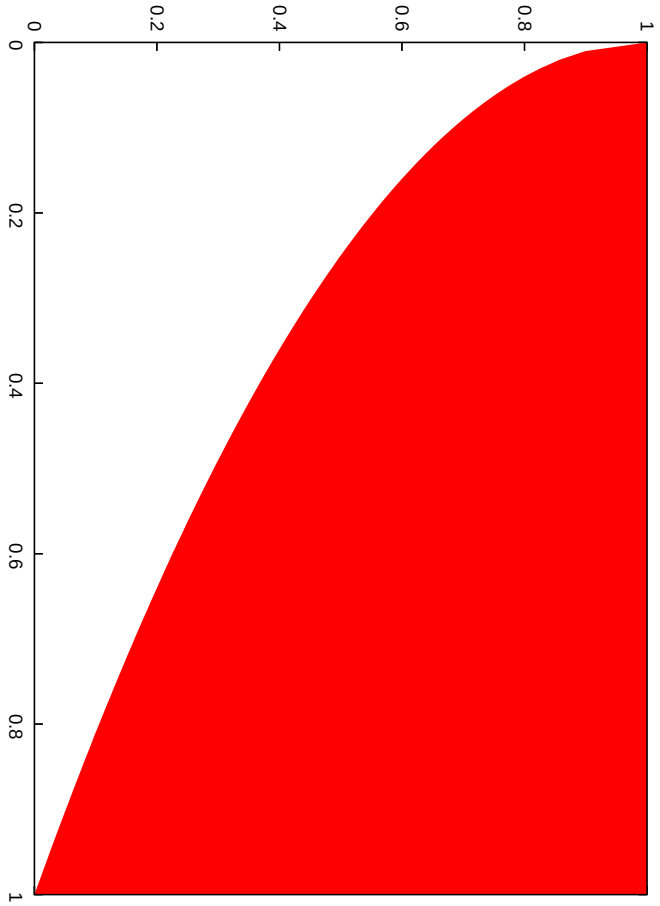

FIG. 5.1 – Tracé de la courbe $\tau = 1 - \sqrt{R}$. Au delà de cette courbe (surface colorée), le problème de décoder τ erreurs dans un code de Reed-Solomon de taux de transmission $R = \frac{k}{n}$ est difficile. En dessous, il devient facile, en utilisant l'algorithme de Guruswami-Sudan.

De plus la donnée de P est assez courte : il suffit de donner *un mot* de longueur n, à comparer au cryptosystème de McEliece basé sur les codes de Goppa, où il faut donner *une matrice* $k \times n$, qui est la matrice génératrice du code masqué.

En pratique, nous avons proposé un système de chiffrement à clé publique, dont la clé publique est de 3072 bits, à comparer aux $\approx$ 500000 bits obtenus pour le cryptosystème classique de McEliece.

Attaques et réparations Notre cryptosystème a subi une attaque sur le message : il existe un algorithme, qui est capable de retrouver le message clair à partir du message chiffré. Cette attaque est due à Coron [Cor03a, Cor04], et ne permet pas de retrouver c et E à partir de $c+E$. Mais elle montre que le code $C = \mathrm{RS}_k \oplus E$ est décodable en temps polynomial, même sans connaître E.

Avec Pierre Loidreau [AFL03], nous avons proposé une réparation, assez technique, qui utilise des propriétés de l'opérateur trace : $\mathrm{Tr} : \mathbb{F}_{2^m} \to \mathbb{F}_2$. Cette technique permettait de faire échouer l'algorithme de Coron. Mais Coron a pu adapter son algorithme à ce nouveau contexte, pour cryptanalyser le nouveau système de chiffrement [Cor03b]. Kiayias et Yung ont aussi essayé de réparer notre système, sans succés [KY04].

En conclusion, notre système ne posséde pas de *réduction de sécurité*. C'est-à-dire que nous n'avons pas pu montrer qu'il est équivalent de retrouver le message clair à partir du message chiffré et de résoudre le problème du décodage du code de Reed-Solomon associé. Et d'ailleurs cette équivalence est fausse, comme l'algorithme de Coron le montre.

Le problème de décodage des codes de Reed-Solomon, dans sa version difficile (quand le taux d'erreur est élevé), est cependant une primitive intéressante pour bâtir des cryptosystèmes. Mais ceux-ci semblent difficiles à obtenir : Kiayias et Yung ont proposé quelques maigres primitives fondées sur ce problème [KY01b, KY01a, KY02b, KY02a, KY05]. Entre autres ils ont proposé un mécanisme de chiffrement à clé secrète (certes peu rapide), et ils ont le mérite d'avoir bâti une réduction de sécurité, garantissant la sécurité de leur système.

Comme mentionné plus haut, un des sous-produits de notre proposition de cryptosystème est l'étude des meilleurs algorithmes pour décoder les codes de Reed-Solomon quand le taux d'erreur est élevé. C'est important car cela permettra éventuellement d'échelonner d'autres cryptosystèmes fondés sur le décodage des codes de Reed-Solomon. Par exemple, Kiayias et Yung, bien que faisant des systèmes avec une réduction de sécurité théorique, ne font pas cette étude algorithmique, et ne proposent donc pas des paramètres concrets.

5.2 Fonction de hachage fondée sur le décodage par syndrome

Notion de fonction de hachage Une *fonction de hachage cryptographique* est une fonction $h : \{0,1\}^* \to \{0,1\}^r$, prenant en entrée un mot binaire de longueur quelconque, retournant un mot de taille fixe, de r bits, et telle que

1. étant donné y, il est difficile de trouver x tel que $h(x) = y$ (*pre-image resistance*) ;
2. étant donné y et x tel que $h(x) = y$, il est difficile de trouver x' tel que $h(x') = y$ (*second pre-image resistance*) ;
3. il est difficile de trouver un couple (x, x') tel que $h(x) = h(x')$ (*collision resistance*).

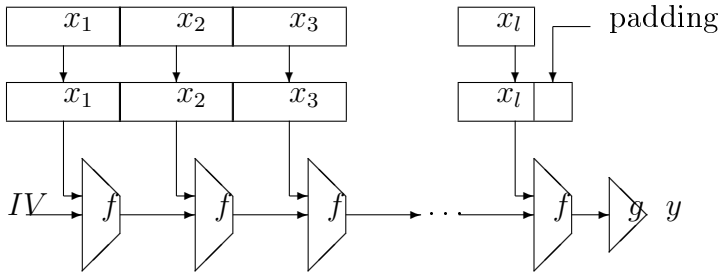

FIG. 5.2 – Schéma itératif de Merkle et Damgård, où f est la fonction de compression.

Une telle primitive, qui semble très simple, est pourtant fort utile en cryptographie, et est constamment et fréquemment utilisée (certificats numériques, intégrité des documents numériques etc).

Nous nous contenterons de définir la notion de difficulté à l'aide de la notion heuristique de *facteur de travail* (WF, work factor), qui est le nombre d'opérations élémentaires nécessaires à atteindre l'un des buts ci-dessus. On estime actuellement qu'un facteur de travail de 2^{80} garantit une grande sécurité.

Les fonctions de hachage les plus populaires sont celles basées sur `MD4`, qui est un schéma général d'algorithme : `MD4`, `MD5`, `SHA` et ses variantes, `RIPEMD` etc. Il s'agit d'algorithmes dédiés, très proches des algorithmes de la cryptographie à clé secrète. Ces fonctions ont été récemment attaquées [WLF+05, WY05, BCJ+05] après l'annonce [WFLY04]. Il y a depuis une forte demande de nouvelles fonctions de hachage cryptographiques, et une nouvelle proposition serait très pertinente, pour palier aux faiblesses de sécurité des fonctions connues.

Le principe de Markle et Damgård Nous allons proposer une construction de fonction de hachage, suivant le schéma classique de Merkle et Damgård [Mer90, Dr90], qui repose sur la notion de *fonction de compression*. Une fonction de compression est une fonction $f : \mathbb{F}_2^l \to \mathbb{F}_2^r$, avec $l > r$

On découpe alors le message $x \in \{0,1\}^*$ en blocs de r bits $x_1 \dots x_n$. On itère ensuite l'opération

$$z_{i+1} \leftarrow f(z_i || x_i),$$

où z_i est l'état courant, initialisé à un vecteur constant z_1, et où x_n, le dernier bloc, a été au préalable augmenté (paddé).

Enfin, une dernière transformation g est éventuellement appliquée au dernier état z_n obtenu, pour obtenir le haché final $g(z_n)$. Cette construction est figurée sur le schéma 5.2.

Il est prouvé [Mer90, Dr90] que la sécurité de la fonction globale h n'est pas moindre que celle de la fonction itérée f. Le concepteur de la fonction de hachage h peut alors se concentrer sur les fonctions auxiliaires f et g.

C'est ce que nous ferons, mais nous avons d'abord besoin de donner la notion de décodage par syndrome.

Décodage par syndrome Un code linéaire binaire C de longueur n étant un $\mathbb{F}_2$ sous espace vectoriel de $\mathbb{F}_2^n$, il peut être spécifié par sa *matrice de parité* H, qui est telle que

$$c \in C \iff Hc^t = 0.$$

Dans le contexte du décodage, soit $y \in \mathbb{F}_2^n$ le mot reçu, et c mot de code à distance w de y. On écrit $c = y - e$, où e est l'erreur, et on a

$$Hc^t = 0 \implies Hy^t = He^t.$$

On connait donc la quantité $S = He^t$ qu'on appelle le *syndrome* de e. Pour décoder, il suffit de trouver e de poids inférieur à w tel que $He^t = S$. C'est le problème du décodage par syndrome, qui est prouvé et réputé difficle [BMvT78, BN90], même avec précalcul.

De même, lorsque $S = 0$, trouver $e \neq 0$, de poids inférieur à w, tel que $He^t = 0$ est aussi un problème difficile : c'est la recherche de mots de poids faible dans un code [Var97].

Notre fonction de compression Pour construire notre fonction de compression, nous proposons premièrement de choisir strictement aléatoirement une matrice de parité H $r \times n$ binaire, et nous considérons le code C associé.

Ensuite, notons $W_{n,w}$ l'ensemble des mots binaires de longueur n et de poids w. Nous proposons de prendre comme fonction de compression la fonction

$$\begin{array}{rccc} \tilde{f}: & W_{n,w} & \to & \mathbb{F}_2^r \\ & e & \mapsto & He^t \end{array}.$$

qui est à la fois difficile à inverser (décodage par syndrome), et pour laquelle il est difficile de trouver $e_1, e_2 \in W_{n,w}$ tels que

$$\tilde{f}e_1 = \tilde{f}e_2,$$

ce qui correspond à la recherche de mots de faible poids : si $\tilde{f}e_1 = \tilde{f}e_2$, où e_1 et e_2 sont de poids inférieur à w, alors $\tilde{f}(e_1 - e_2) = 0$, c'est-à-dire que le mot $e_1 - e_2$ est dans le code, et a un poinds inférieur à $2w$.

Cette fonction de compression ne se prête pas facilement à une réalisation pratique : il faut en effet trouver un moyen de coder les éléments de $W_{n,w}$. Premièrement, le cardinal de $W_{n,w}$ est $\binom{n}{w}$, qui n'est pas une puissance de 2. Ensuite, les algorithmes réalisant un codage des entiers de $\{0, \ldots, \binom{n}{w} - 1\}$ vers $W_{n,w}$ requièrent une arithmétique de grands entiers (manipulation de coefficients binomiaux) et sont coûteux [Cov73].

Nous proposons de considérer l'ensemble $R_{n,w}$ des *mots réguliers* de longueur n et de poids w. Cet ensemble est défini quand n est un multiple de w : on découpe l'intervalle $\{0, \ldots, n-1\}$ en w intervalles de longueur $\frac{n}{w}$. Un mot régulier de longueur n et de poids w est un mot ayant exactement un 1 dans chacun de ces intervalles, voir Figure 5.3. Le cardinal de $R_{n,w}$ est alors $(\frac{n}{w})^w$.

Pour construire notre fonction de hachage, nous avons le choix de n et de w, que nous prenons tel que $\frac{w}{n}$ soit une puissance de 2, par exemple 2^{l_0}. Alors le cardinal de $R_{n,w}$ est 2^{wl_0}, soit 2^l avec $l = wl_0$.

Il est alors facile de construire un codage $\phi : \{0,1\}^l \to R_{n,w}$, de plus très efficace.

Nous proposons donc comme fonction de compression la fonction f suivante :

$$\begin{array}{rccc} f: & \{0,1\}^l & \to & \mathbb{F}_2^r \\ & x & \mapsto & H(\phi(x))^t \end{array}.$$

Rappelons que notre fonction est une fonction de *compression* : on doit choisir les paramètres tels que $l > r$.

Sécurité théorique On peut se demander si le problème d'inverser, c'est-à-dire de trouver un mot régulier admettant un syndrome donné, n'est pas devenu facile, en comparaison du problème standard de décodage par syndrome. La même question

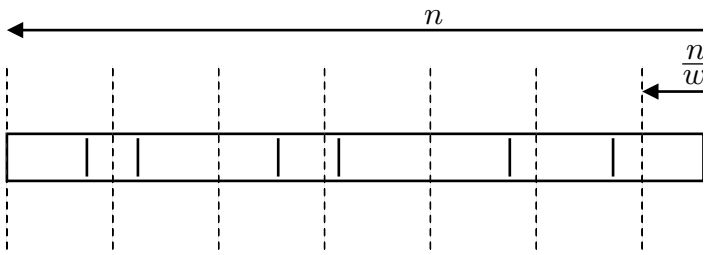

FIG. 5.3 – Un mot de poids régulier : chaque trait vertical plein indique un « 1 », les autres positions étant nulles.

se pose pour le problème des collisions. Matthieu Finiasz a montré dans sa thèse que les problèmes décisionnels associés à ces deux problèmes sont NP-complets [Fin04].

Nous pensons de plus que le problème est difficile pour des instances aléatoires, et non pas seulement dans le cas le pire, comme dans le cas du décodage classique. C'est pourquoi nous préconisons de choisir la matrice H aléatoirement, et de la garder en table, plutôt que d'utiliser la matrice de parité d'un code particulier, qui pourrait avoir trop de structure.

À l'opposé des fonctions de hachage usuelles, notre construction présente une réduction de sécurité : inverser la fonction de hachage revient exactement à décoder. Il y a peu de propositions de fonctions à sécurité : la notre et celle de Lenstra et al. qui repose sur la théorie des nombres [CLS06].

Sécurité en pratique En pratique, nous avons étudié les algorithmes de décodage basés sur la méthode de la recherche d'ensembles d'informations dont il existe beaucoup de variantes [Bar98]. Ces méthodes sont génériques, mais elles sont faciles à adapter aux cas des mots réguliers. La complexité de ces algorithmes est exponentielle en r, la taille du haché. Il est alors possible de concevoir un jeu de paramètres tel que le coût de cet algorithme soit de l'ordre de 2^{80}, comme je l'ai fait avec Matthieu Finiasz [AFS03].

Notre fonction de compression prête cependant le flanc à une autre attaque, qui est celle du *paradoxe des anniversaires généralisé* de Wagner [Wag02]. La complexité de cette méthode est aussi exponentielle, mais avec un exposant plus petit.

Cette remarque sur notre première proposition de paramètres a été faite par Coron et Joux [CJ04], ce qui permet de casser notre premier jeu de paramètres. Toutefois la complexité reste exponentielle, et le schéma de principe reste bon. Il suffit simplement de réajuster les paramètres, comme nous l'avons fait dans [AFS05a, AFS05b, AFS05c].

Efficacité Notre fonction de compression nécessite très peu de calculs : il suffit de faire un produit matrice-vecteur, à coefficients dans le corps $\mathbb{F}_2$. La seule opération arithmétique mise en œuvre est le `xor` binaire, très peu coûteux. Le tableau 5.1 présente des jeux de paramètres pour lesquels la sécurité est établie. On voit qu'il y a compromis entre la taille de la matrice H et le nombre de `xor` à effectuer par bit du message.

En pratique, nous avons constaté que le facteur limitant la rapidité de la fonction de hachage est la latence des accès mémoire : la matrice H étant assez grosse, elle ne contient pas dans la mémoire cache des processeurs, et les accès à la mémoire deviennent prépondérants devant les opérations dans le processeur.

5.3 Conclusion

Nous avons proposé deux nouvelles constructions de primitives cryptographiques fondées sur les codes. La première construction, un algorithme de chiffrement, présente l'avantage de présenter des clés plus courtes, mais ne posséde pas de réduction de sécurité : elle a été cassée, sans aucune réparation en vue. On voit cependant que le problème du décodage des Reed-Solomon est potentiellement intéressant pour la cryptographie (Kiayias et Yun [KY01b, KY01a, KY02b, KY02a, KY05]), mais la réalisation de primitives cryptographiques efficaces et pertinentes est un objectif fuyant.

La deuxième construction, plus basique, d'une fonction de hachage, présente une réduction de sécurité. Mais il a été assez difficile de concevoir un jeu de paramètres garantissant une bonne sécurité.

Cette fonction de hachage complète la famille des cryptosystèmes fondés sur les codes : le chiffrement était couvert par McEliece [McE78], la signature par Courtois,

$\log_2\left(\frac{n}{w}\right)$	w	n	N_{XOR}	matrix size
16	41	2686976	64.0	~ 1 Gbit
15	44	1441792	67.7	550 Mbits
14	47	770048	72.9	293 Mbits
13	51	417792	77.6	159 Mbits
12	55	225280	84.6	86 Mbits
11	60	122880	92.3	47 Mbits
10	67	68608	99.3	26 Mbits
9	75	38400	109.1	15 Mbits
8	85	21760	121.4	8.3 Mbits
7	98	12544	137.1	4.8 Mbits
6	116	7424	156.8	2.8 Mbits
5	142	4544	183.2	1.7 Mbits
4	185	2960	217.6	1.1 Mbits

TAB. 5.1 – Choix de paramètres pour $r = 400$ et $a = 4$. N_{XOR} est le nombre de `xor` bit à bit à effectuer, par bit de message (tableau compilé par M. Finiasz).

Finiasz et Sendrier [CFS01], et le hachage par notre proposition. Cependant elle souffre, comme les autres systèmes fondés sur les codes, du défaut de devoir stocker une matrice de grande taille.

Chapitre 6

Cryptographie fondée sur le protocole de Diffie-Hellman

L'archétype même du protocole d'échange de clé est le protocole de Diffie-Hellman [DH76]. Étant donné un générateur g d'un groupe cyclique, deux utilisateurs Alice et Bob peuvent convenir d'une clé secrète comme suit :

1. Alice envoie sur un canal public g^{x_a} à Bob ;
2. Bob envoie sur un canal public g^{x_b} à Alice ;
3. Alice peut calculer $(g^{x_b})^{x_a}$, Bob peut calculer $(g^{x_a})^{x_b}$;
4. ces deux données sont égales, elles peuvent servir de clé partagée dans une application ultérieure.

Dans ce chapitre, je décris deux travaux que j'ai effectués en vue d'applications directes, et ces deux travaux reposent tous les deux sur le protocole de Diffie-Hellman. Le premier travail étudiait certains problèmes relatifs aux droits d'auteurs, le deuxième traitait de problèmes de distibution de clé dans des réseaux sans fil.

Protection des droits d'auteurs Avec Caroline Fontaine, dans le cadre du projet européen Aquarelle (1996-2000), j'ai utilisé le protocole de Diffie-Hellman pour mettre en œuvre une autorité de certification dans un schéma de protection de droits d'auteurs, dans le projet européen Aquarelle pour la diffusion du patrimoine culturel européen.

La technique utilisée pour protéger les droits d'auteurs était celle du *marquage d'image* : on incruste une information invisible dans l'image, de sorte à identifier le propriétaire de l'image. Notre contribution a été de définir comment gérer les clés dans un système complet de marquage d'images, en faisant abstraction de l'algorithme de marquage lui-même [Aug96b, AF97, AF98, ADF98, ABD+99].

Réseaux sans fil Dans un autre contexte d'application, les réseaux Ad Hoc, j'ai encadré la thèse de Raghav Bhaskar, qui a travaillé sur la généralisation du protocole de Diffie-Hellman à un nombre quelconque de participants. Il n'y a pas de généralisation immédiate, ni canonique, du protocole de Diffie-Hellman, et de nombreuses propositions ont été avancées, dont notamment le protocole CLIQUE [AST00, STW00, STW96], les protocoles à base d'arbre [KPT00] et le protocole BM [BD95]. Nous avons contribué au domaine en proposant un nouveau protocole, qui présente la caractéristique intéressante de résister au pertes de message. Nous avons à la fois une proposition cryptographique avec une preuve de sécurité [ABIS05, ABIS07], mais aussi avec nos collègues de HiperCom, nous spécifions complètement ce protocole dans les réseaux avec perte de message [BMA+06].

6.1 Protection des droits d'auteurs

6.1.1 Le projet européen Aquarelle

Dans le cadre du projet européen Aquarelle (1996-2000), j'ai eu à m'occuper, avec Caroline Fontaine, des problèmes liés à la protection d'images diffusées sur Internet. Ce projet européen, très important en nombre de participants, visait à unifier et à intégrer les ressources patrimoniales européennes, des grands musées nationaux ou d'instituts privés. Le but de cette unification était de présenter un accès cohérent au patrimoine culturel européen, en utilisant le Web, alors naissant.

Les partenaires culturels s'étaient inquiétés du problème de la protection des droits d'auteurs relatifs aux images diffusées par le système. La solution envisagée était d'utiliser la technique du *marquage d'images* (watermarking). Cela consiste à rajouter une sorte de filigrane, invisible à l'œil nu, mais qui peut être détectée suivant un procédé donné. Ce filigrane contient de l'information au sujet de l'ayant-droit sur l'image.

6.1.2 Problèmes de clés relatifs au marquage d'images

Très vite, il est apparu que la notion de marquage n'était pas claire. Principalement le problème se pose de savoir comment l'opération de marquage est paramétrée par des clés. En effet, si on a une opération de marquage fixe, sans clé, notée f :

$$f(I, M) \mapsto I^*$$

où M est un message à camoufler, I l'image, et I^* l'image marquée, alors l'attaquant, qui connaît f,[1] peut déterminer comment le marquage a été fait, et enlever la marque.

Il faut donc supposer que l'opération de marquage est paramétrée par une clé K :

$$f_K(I, M) \mapsto I^*,$$

de sorte que l'attaquant, pour retirer la marque, doit essayer toutes les clés K.

Le problème se situe au niveau de la preuve de l'existence de la marque : pour cela il faut détenir la clé K. Se pose alors la problématique suivante :

- soit le propriétaire de l'image, disposant de la clé K, effectue la vérification lui même, mais il ne peut pas convaincre autrui ;
- soit il donne la clé K pour permettre à autrui de vérifier l'existence de la marque. Dans ce cas, il compromet la sécurité de la marque, puisqu'il a révélé la clé K de marquage.

Nous avions décidé de faire effectuer la vérification de la marque par un *tiers de confiance*. C'est quelqu'un qui est considéré comme fiable à la fois par les ayant-droits et par les personnes désirant avoir une preuve de l'existence de la marque.

6.1.3 Séparation du marquage et de la vérification

Il reste alors le problème de savoir qui effectue l'opération de marquage. Si le marquage est à la charge du tiers de confiance, alors se pose le problème de lui transférer les images non marquées de manière sécurisée. De plus, si le tiers de confiance gère de nombreux ayants-droit, il peut succomber sous une charge de calcul trop lourde (les algorithmes de marquage dont nous disposions en 1996 prenaient de l'ordre d'une minute pour marquer une seule image).

[1] Je rappelle les principe de Kerckhoff en cryptographie, qui stipule que le secret doit résider dans les clés et non pas dans les méthodes ou les algorithmes, qui finissent en général par être éventés.

Nous avons décidé de répartir le travail comme suit : à charge de l'ayant droit de marquer les images, à charge du tiers de confiance de vérifier le marquage. Ainsi le coût du marquage n'était pas reporté sur les tiers de confiance.

Les clés de marquage doivent alors être partagées entre l'ayant-droit et le tiers de confiance. Nous avons proposé d'employer le protocole de Diffie-Hellman pour établir la clé entre l'ayant-droit et le tiers de confiance [Aug96b, AF97, AF98, ADF98, ABD+99].

Dans le cadre du projet Aquarelle, nous avons réalisé un prototype (DHWM : Diffie-Hellman Watermarking) de l'échange de clés entre les deux parties, suivies de la phase de marquage, et de la phase de vérification. Pour la partie marquage et vérification du marquage proprement dite, nous avons travaillé sur deux algorithmes proposés par l'Université Catholique de Louvain (Benoit Macq, Jean-François Delaigle). Enfin, pour la partie réseau, nous avons utilisé le protocole `HTTP` (version 1.0 à l'époque). La principale difficulté était lors de l'étape de vérification de la marque : le chargement d'une image par le serveur implémentant le tiers de confiance nécessite un processus de `upload` qui était très mal documenté.

Ce prototype a fait l'objet d'une démonstration devant des représentants de la Comission Européenne.

Ces travaux sont anciens, et je n'ai pas poursuivi dans cette direction par la suite. Par souci de complétude, je signale que les domaines de la cryptographie et du marquage ont évolué pour se rencontrer, pour définir la *sécurité du marquage* [Fur05, CDF06], et pour aussi introduire le « marquage à clé publique » [UUC+05]. Donc la séparation que nous avions faite entre cryptographie et marquage n'est plus d'actualité.

6.2 Réseaux sans fil

6.2.1 La variante de Hughes du protocole de Diffie-Hellman

Les protocoles de mise en accord de clé semblent intéressants pour instiller de la sécurité dans des petits réseaux sans fil. En effet sur de tels canaux de communication, il est facile d'écouter les messages transmis, et d'injecter des messages frauduleux. Une solution intéressante est d'utiliser la cryptographie symétrique, pour chiffrer les messages afin d'obtenir de la confidentialité, et aussi pour authentifier les messages. Les méthodes pour atteindre ces objectifs nécessitent l'existence d'une clé partagée par tous les membres du réseaux. Il reste alors le problème de mettre tous les membres du réseau d'accord sur une clé secrète.

Le protocole de Diffie-Hellman permet à deux utilisateurs de se mettre d'accord sur un clé partagée, en échangeant seulement des données publiques. Il n'est pas évident de généraliser ce protocole à m utilisateurs. Avec Raghav Bhaskar, nous avons proposé un protocole d'échange de clé basé sur la variante dite de Hughes du protocole de Diffie-Hellman [Hug94]. Cette variante, présentée informellement à CRYPTO'94, est peu connue. Elle est présentée dans le livre de Schneier, deuxième édition [Sch95]. Je la rappelle ici :

1. Alice envoie sur un canal public g^{x_a} à Bob ;
2. Bob envoie sur un canal public $g^{x_a x_b}$ à Alice ;
3. Alice peut calculer $(g^{x_a x_b})^{x_a^{-1}} = g^{x_b}$;
4. Alice et Bob partagent donc g^{x_b}.

On voit que le secret g^{x_b} est fixé par Bob, et ne dépend pas de la contribution d'Alice g^{x_a}. C'est ce que nous utiliserons.

6.2.2 Pour un nombre quelconque d'utilisateurs

Pour généraliser à $m+1$ utilisateurs $M_0, \ldots, M_m$, on va d'abord supposer qu'ils sont d'accord sur un chef M_0. Le protocole se déroule comme suit :

1. chaque membre M_i, $i \in \{1, \ldots, m\}$, envoie au chef M_0 g^{x_i}, qu'on appelera *sa contribution* ;
2. le chef M_0 répond $g^{x_i x_0}$ à chaque membre M_i, $\in \{1, \ldots, m\}$;
3. chaque membre M_i peut calculer $(g^{x_i x_0})^{x_i^{-1}} = g^{x_0}$: tous les membres partagent donc g^{x_0}.

Pour rendre la clé réellement dépendante des contributions de chaque membre, on décide que la clé partagée sera

$$g^{x_0} \prod_{i \in \{1,\ldots,m\}} g^{x_i x_0}.$$

6.2.3 Preuve de sécurité

Une preuve de sécurité, d'un tel protocole, est une réduction d'un problème décisionnel réputé difficile à celui d'une attaque sur la clé partagée. Ainsi on déduira qu'un attaquant efficace ne peut pas exister, car on pourrait l'utiliser pour résoudre le problème difficile.

Très précisément, il faut définir le modèle de sécurité qui définit le cadre dans lequel la réduction va se faire [BCP02, KY03] :

1. modélisation précise des participants, d'exécutions concurrentes du protocole ;
2. modélisation de la notion de session, et des changements dynamiques de la composition du groupe ;
3. modélisation de l'attaquant : les moyens mis à sa disposition, et ses objectifs.

Raghav Bhaskar, dans sa thèse, a prouvé la sécurité de ce protocole contre un *adversaire passif* (qui se contente d'écouter les communications). Pour un *adversaire actif*, il suffit de faire une transformation générique due à Katz et Yung, qui consiste à authentifier tous les messages échangés avec des signatures électroniques [KY03]. Toutefois, pour des raisons d'efficacité, Raghav Bhaskar a construit une version directement authentifiée du protocole, qui nécessite moins d'échanges que la transformation générique de Katz et Yung. Les deux versions du protocole admettent une preuve de sécurité dans le cas de l'adversaire actif [ABIS07].

6.2.4 Résistance aux pertes de messages

Nous comparons notre protocole aux protocoles existants dans les tables 6.1 et 6.2. La table 6.1 rassemble les protocoles ayant un nombre non constant d'étapes, alors que la table 6.2 présente les protocoles ayant un nombre constant d'étapes. Notre protocole se compare favorablement aux protocoles existants.

Mais surtout, notre protocole résiste aux pertes de messages comme suit :

- si un membre M_i échoue à faire parvenir sa contribution à M_0, celui-ci ne le voit pas, mais le protocole fonctionne correctement pour le reste du groupe ;
- si un membre ne reçoit pas la réponse du chef, il échoue à calculer la clé, mais le reste du groupe calcule correctement la clé.

Ces propriétés rendent notre protocole bien adapté aux réseaux sans fil, où les pertes de message sont fréquentes.

De plus notre protocole présente l'intêrêt qu'il suffit de se mettre d'accord sur un chef, les autres membres n'ayant pas besoin de structurer. L'inconvénient est que le chef a une charge de calcul plus lourde que les autres participants : le protocole est déséquilibré, et nous l'appelons AGDH (Asymetric Group Diffie-Hellman).

	Expo par membre	Messages	Broadcasts	Étapes
ITW [ITW82]	m	$m(m-1)$	0	$m-1$
GDH.1 [STW96]	$i+1$	$2(m-1)$	0	$2(m-1)$
GDH.2 [BCP02]	$i+1$	$m-1$	1	m
GDH.3 [STW96]	3	$2m-3$	2	$m+1$
Perrig [Per99]	$\log_2 m+1$	m	$m-2$	$\log_2 m$
Dutta [DB05]	$\log_3 m$	m	m	$\log_3 m$

TAB. 6.1 – Récapitulatif des protocoles à grand nombre d'étapess pour m membres.

	Expo par membre	Messages	Étapes	Structure
Octopus [BW98]	4	$3m-4$	4	Hypercube
BDB [BD95, KY03]	3	$2m$	2	Cercle
Catalano [BC04]	$m+1$	$m(m-1)$	2	Aucune
KLL [KLL04]	3	$2m$	2	Cercle
NKYW [NLKW04]	$2^{\ddagger}$	m	2	chef
STR [KPT04]	$(m-i)^{*}$	m	2	arbre
Le notre (AGDH)	2^{**}	m	2	chef

† : m exponentiations pour la station de base.
‡ : $m+1$ exponentiations et $(m-1)$ inverse pour le nœud parent.
$*$: jusqu'à $2m$ exponentiations pour le nœud « sponsor ».
$**$: m exponentiations pour le chef.

TAB. 6.2 – Présentation des protocoles ayant un nombre constant d'étapes, pour m membres.

6.2.5 De la cryptographie aux réseaux

Ces bonnes propriétés du protocole ont intéressé nos collégues de l'INRIA travaillant dans le domaine des réseaux sans fil. Avec Raghav Bhaksar, Cédric Adjih, et Paul Mühlethaler du projet HiperCom, j'ai travaillé à la spécification du protocole pour les réseaux sans fil [BMA+06]. La principale difficulté est de s'assurer que tous les membres ont bien la même vision du groupe, et qu'ils sont d'accord sur le chef.

Pour cela, le chef du groupe émet périodiquement des messages `IGROUP`, dans lequel il inclut sa réponse cryptographique telle que précédemment, et la composition du groupe, telle qu'il la conçoit. De son coté, chaque membre M_i envoie un message `IREPLY`, où il met sa contribution cryptographique. On maintient, au niveau de tous les participants, les deux propriétés suivantes :

- chaque membre sait l'existence d'un chef et la composition du groupe à la réception des messages `IGROUP` ;
- le chef connait les membres qui sont présents dans le groupe à la réception des messages `IREPLY` qu'il reçoit.

Il y a un problème de démarrage du processus. Nous le réglons avec un mécanisme d'auto-élection du chef : quand un membre ne reçoit pas de messages `IGROUP`, il décide de s'auto-élire chef. Cela se produit en cas de défection du chef, ou en cas de scission du réseau. Ce mécanisme d'auto-élection peut présenter des conflits, quand plusieurs membres s'élisent chef simultanément. Une règle simple permet de résoudre ces conflits : est chef celui qui a l'identifiant le plus faible (adresse MAC, adresse IP, etc).

Les travaux sont en cours pour spécifier complètement ce protocole : la description simple en quelques lignes, vue ci-dessus, donne lieu à plusieurs pages de pseudocode pour décrire tous les états de tous les membres, avec leurs transitions.

Sécurité dans le monde réel Une question difficile est de savoir si le protocole, ainsi adapté et spécifié pour les réseaux avec perte de messages, est toujours sûr au sens formel (c'est-à-dire qu'il admet une preuve de sécurité). En effet, dans notre adaptation, il y a beaucoup de répétitions des messages `IGROUP` du chef, pour confirmer sa vision du groupe, et aussi des messages `IREPLY` des membres pour confirmer leur présence. De plus la notion de session devient assez diffuse quand les notions temporelles paraîssent.

En résumé, les notions très formelles des modèles cryptographiques standards deviennent floues dans notre adaptation aux réseaux sans fil. La problématique de recherche qui se pose est de prouver la sécurité du protocole adapté. Cela passe d'abord par définir un modèle de sécurité, qui doit nécessairement dévier de ceux proposés en standard dans la littérature [BR94, BCPQ01, KY03].

6.3 Conclusion

Le protocole de Diffie-Hellman, considéré comme fondateur en 1976 de la cryptographie à clé publique, se révèle être très intéressant pour les applications. En ce qui me concerne, je suis passé d'une utilisation *heuristique* de ce protocole (marquage d'image), à une utilisation plus rigoureuse (réseaux sans fils). Dans le premier cas, je n'avais pas établi de preuve de sécurité du protocole mis en place pour le marquage d'images, alors que dans le deuxième cas, avec Raghav Bhaskar, nous avons fourni un preuve de sécurité du protocole cryptographique.

Pour ce dernier cas, nous avons tenté d'adapter le protocole au cas des réseaux avec perte de message. Cela est assez difficile de bien spécifier le protocole réel, et du point de vue théorique, le modèle de sécurité cryptographique ne reflète plus l'application réelle. Développer un modèle de sécurité est la première étape avant de fournir la preuve de sécurité. Mais cela est difficile, et notamment, c'est la notion même de *session* qui pose problème, comme souligné dans [BM06].

Troisième partie

Conclusion et perspectives

Chapitre 7

Conclusion et perspectives de recherches

En guise de conclusion, je vais présenter des axes de recherche, dans la continuité du codage algébrique. Le codage algébrique est le mieux incarné par les codes de Reed-Solomon, et les codes géométriques, qui sont de très bons codes, massivement utilisés en pratique. Alors que ces codes étaient jusque maintenant décodés sur la base de la « décision dure » les travaux de Kötter et Vardy [VK00, KV03] ont ouvert la porte au décodage « à décision souple » des codes de Reed-Solomon, sur la base de l'algorithme de Guruswami-Sudan et de l'introduction de multiplicités variables dans l'interpolation bivariée. Toutefois de nombreux problèmes se posent pour traduire cette approche dans les applications, le principal étant celui de l'implémentation. Les deux étapes des algorithmes à base d'interpolation (interpolation bivariée et recherche de racines) sont encore trop coûteuses pour atteindre les débits de transmission exigés actuellement, sur du matériel raisonnable.

Mes propositions de travaux futurs sont : le décodage en liste des codes de Reed-Solomon et des codes géométriques ; la construction des codes géométriques et l'étude des algorithmes classiques (non basés sur l'interpolation) pour les décoder ; enfin l'étude des aspects négatifs du problème du décodage pour construire des cryptosystèmes fondés sur les codes d'évaluation.

7.1 Décodage des codes de Reed-Solomon avec l'algorithme de Guruswami-Sudan

Je propose d'abord de développer des méthodes pour l'implantation efficace de l'algorithme de Guruswami-Sudan. En effet les auteurs se sont contentés de décrire leurs algorithmes comme étant de complexité polynomiale, en utilisant comme boîtes noires deux briques provenant du calcul formel : la résolution exacte de systèmes d'équations linéaires et la factorisation de polynômes à plusieurs variables sur les corps finis. Si des algorithmes ont été proposés pour résoudre ces problèmes, il n'est pas clair quel est le plus susceptible d'être appliqué en pratique. Il faut donc les implémenter pour mieux saisir leur efficacité. J'ai déjà une implémentation en langage de bas niveau, C, dans le cas où les multiplicités sont égales à 1, mais le cas des multiplicités supérieures semble plus difficile à implanter de manière efficace. Je compte développer les algorithmes utilisant des bases de Gröbner qui interviennent en tant qu'objets dans l'étape d'algèbre linéaire et il semble que des techniques de changement de bases [FGLM93] peuvent être employées pour obtenir le polynôme minimal d'interpolation [LO06] ; quant à l'étape de factorisation, je compte la rendre

"factorisation-free", ce qui a déjà été possible dans le cas de l'algorithme de Sudan simple, avec multiplicité égale à un.

7.2 Généralisation en codes géométriques

L'axe majeur de mes futurs travaux sera la construction des codes géométriques et le développement de leurs algorithmes de décodage. Comme déjà dit, les codes géométriques permettent de remédier à un défaut majeur des codes de Reed-Solomon : la taille de l'alphabet d'un code de Reed-Solomon doit être supérieure à la longueur du code, ce qui empêche d'utiliser des codes de Reed-Solomon très longs. Les codes géométriques présentent l'avantage, à alphabet fixé, de permettre de construire des codes aussi longs que l'on désire. Réciproquement, à longueur donnée, on peut se permettre d'avoir un alphabet plus petit, donc plus maniable. L'utilisation de cette dernière propriété a été étudiée dans pour accélérer le décodage dans [BS04], et aussi dans le contexte du codage pour les disques durs dans [MM05].

Il faut étudier l'algorithme de Guruswami-Sudan pour les codes géométriques. En effet, la généralisation des codes de Reed-Solomon aux codes géométriques étant naturelle, la généralisation aux codes géométriques se fait correctement [GS99].

Comme pour les codes de Reed-Solomon, il faut faire baisser la complexité des algorithmes. Le travail d'implantation reste à faire, et de grandes connaissances des courbes algébriques sont nécessaires, pour manier efficacement les outils à mettre en œuvre. Toutefois, toute amélioration obtenue dans le cas des Reed-Solomon s'étendra peu ou prou aux cas des courbes géométriques, à condition de généraliser avec les bons objets. Donc les bonnes idées obtenues pour le cas des Reed-Solomon devraient se prolonger naturellement au cas des codes géométriques.

Une nouvelle question se pose, qui n'existe pas dans le cas des codes de Reed-Solomon (car ceux-ci sont uniquement déterminés, alors que la classe des codes géométriques est très variée) : peut-on construire (et décoder) des codes géométriques qui sont bons pour le décodage en liste ? En effet, Guruswami à montré [Gur04] que des codes bons pour le critère classique de la distance minimale peuvent prétendre à être bons pour le décodage en liste, mais que l'on peut obtenir des résultats encore meilleurs en essayant de chercher directement des codes bons pour le décodage en liste.

Enfin, ces algorithmes de décodage, bien que très prometteurs en termes de capacité de correction, ne sont pas aussi bien finalisés que les algorithmes à base de calcul de syndrome, comme survolés dans [HP95]. Ces derniers algorithmes sont bien compris, et des implantations matérielles ont déjà été soit proposées, soit réalisées à titre de prototype [AKS04]. Je pense qu'on peut encore améliorer ces algorithmes, en utilisant l'arithmétique rapide des polynômes. La transformée de Fourier discrète est en effet très rapide en caractéristique 2, même en très petite longueur : par exemple on a 1014 multiplications pour la longueur 511 [Fed06].

7.3 Cryptographie basée sur les codes de Reed-Solomon

Comme déjà mentionné, la difficulté du décodage a été établie pour la classe restreinte des codes de Reed-Solomon [GV05], qui sont, avec les codes géométriques, le point central de mon programme de recherche. C'est une aubaine, car cela permet d'avoir des instances difficiles d'un problème en cherchant dans une classe plus restreinte. Cela signifie, par exemple, que les instances difficiles sont plus courtes à spécifier, ce qui pourrait se traduire, en cryptographie, par des clés publiques plus

courtes. Je cherche donc à construire des cryptosystèmes basés sur le problème du décodage des codes de Reed-Solomon.

Le système de chiffrement que j'ai proposé a été cassé, car il était mal fondé : il existait une attaque autre que l'attaque frontale du problème du décodage sous-jacent. Je propose de construire des systèmes cryptographiques *bien fondés* sur la difficulté du décodage des codes de Reed-Solomon. Pour cela je commencerai avec les travaux de Kiayias et Yung [KY05], qui sont capables d'utiliser la primitive du décodage des codes de Reed-Solomon dans diverses constructions cryptographiques. Ils ont déjà fait le travail de prouver les bonnes propriétés cryptographiques du problème du décodage des codes de Reed-Solomon, et sont capables de bâtir des primitives cryptographiques avec les bonnes réductions de sécurité. Toutefois les primitives obtenues sont quelque peu ésotériques, et je me concentrerai sur la construction de primitives plus classiques.

Il semble intéressant de se poser ici aussi la question de la généralisation aux codes géométriques. La question théorique qui se pose ici, est d'établir la difficulté du décodage pour des codes géométriques à alphabet fixé. Un résultat partiel est dû à Cheng [Che05], qui ne s'applique qu'aux codes géométriques définis sur les courbes elliptiques, dont l'alphabet croît exponentiellement en la longueur du code. Or les codes géométriques sont intéressants en ce qu'on peut faire croître la longueur en conservant une taille fixe pour l'alphabet. En conséquence, si des cryptosystèmes peuvent être construits sur le problème du décodage des codes géométriques, ils devraient avoir des clés plus courtes que ceux construits avec les codes de Reed-Solomon, car utiliseraient un alphabet plus petit.

Quatrième partie

Présentations détaillées

Chapitre 8

Décodage des codes cycliques avec les bases de Gröbner

Ce chapitre est consacré à mes travaux sur le décodage des codes cycliques généraux, jusqu'à et au delà de la capacité de correction $t = \lfloor \frac{d-1}{2} \rfloor$, en utilisant la théorie de l'élimination et les bases de Gröbner. Le problème est, à partir des syndromes de l'erreur, de retrouver l'erreur, ou plus précisément le polynôme localisateur de l'erreur. Nous verrons qu'il y a deux approches :

- décodage avec syndromes spécialisés ;
- décodage avec syndromes symboliques ;

et nous dirons laquelle de ces méthodes est préférable en pratique.

Ces travaux sont une prolongation des travaux de ma thèse. En effet, alors que dans ma thèse, je me contentais de déterminer, avec des systèmes d'équations algébriques, les mots de poids minimal des codes cycliques, je me suis rendu compte ensuite que les mêmes méthodes peuvent être utilisées en décodage. Les techniques employées pour faire les calculs reposant fortement sur la théorie des bases de Gröbner, cela a conduit assez naturellement à une codirection de thèse avec Jean-Charles Faugère. L'étudiante était Magali Bardet, et nous avons obtenu de nouveaux résultats, notamment grâce à la mise en équations donnée par les identités de Newton.

Dans ce chapitre, je présente les propriétés mathématiques de l'idéal engendré par les équations de Newton, en caractéristique finie.

8.1 Le problème du décodage des codes cycliques

Soit $\mathbb{F}_q$ le corps à q éléments, et n un entier premier à q. Dans le cas de certains énoncés généraux, nous noterons $\mathbb{F}$ le corps de base, non nécessairement fini. Les énoncés les plus significatifs en terme de codage seront obtenus sur $\mathbb{F}_2$ que nous noterons bien $\mathbb{F}_2$. Un code cyclique C de longueur n sur $\mathbb{F}_q$ est un idéal de $\mathbb{F}_q[X]/(X^n - 1)$. L'anneau $\mathbb{F}_q[X]/(X^n - 1)$ étant principal, tout code cyclique C admet un polynôme générateur $g(X)$, qui divise $X^n - 1$. En se donnant une racine primitive n-ième de l'unité α sur $\mathbb{F}_q$, avec $\alpha \in \mathbb{F}_{q^m}$, on définit l'ensemble de définition Q du code C :

$$Q = \{i \in \{0, \ldots, n-1\}; g(\alpha^i) = 0\}. \tag{8.1}$$

On a donc

$$c \in C \iff c(\alpha^i) = 0, \quad \forall i \in Q. \tag{8.2}$$

Soit y le mot reçu, que l'on écrit $y = c + e$ où c étant le mot de code émis et où e est l'erreur. Alors en calculant $y(\alpha^i)$, pour $i \in Q$, on obtient

$$y(\alpha^i) = c(\alpha^i) + e(\alpha^i) = e(\alpha^i); \quad i \in Q. \tag{8.3}$$

Ainsi, le décodeur connaît les $e(\alpha^i)$ pour $i \in Q$, que l'on appelle les syndromes de l'erreur. On les note S_i^*, $i \in Q$. On note bien que les S_i^* sont dans l'extension $\mathbb{F}_{q^m}$ de $\mathbb{F}_q$, celle où réside la racine n-ième de l'unité. De plus, comme $e(X)$ est à coefficients dans $\mathbb{F}_q$, on a $e(\alpha^i)^q = e(\alpha^{iq \bmod n})$, soit

$$S_i^{*q} = S_{iq \bmod n}^*, \quad i \in Q. \tag{8.4}$$

Nous dirons qu'une famille de S_i^*, $i \in \{0, \ldots, n-1\}$, vérifiant les équations (8.4) est *admissible* pour le problème du décodage. Nous sommes en mesure de formuler le problème du décodage d'un code cyclique.

Problème : Décodage par syndrome d'un code cyclique

Instance :
- q une puissance d'un nombre premier, une représentation du corps fini $\mathbb{F}_q$;
- n premier à q, une représentation de $\mathbb{F}_{q^m}$, le corps de décomposition de $X^n - 1$;
- $Q = \{i_1, \ldots, i_l\}$, définissant un code cyclique,
- $t \in \mathbb{N}$,
- des syndromes admissibles $S_{i_1}^*, \ldots, S_{i_l}^*$.

Question : Existe-t-il $e(X) \in \mathbb{F}_q[X]$, $\deg e(X) < n$, et $w(e) \leq t$ tels que

$$e(\alpha^i) = S_i^*, i \in \{i_1, \ldots, i_l\}.$$

C'est le strict pendant du problème du décodage par syndrome, adapté au cas particulier des codes cycliques, et que nous donnons ci-dessous :

Problème : Décodage par Syndrome (d'un code quelconque).

Instance : Une matrice $r \times n$ binaire H, un mot S de F_2^r et un entier $w > 0$.

Question : Existe-t-il un mot x dans F_2^n de poids $\leq w$ tel que $Hx^T = S$?

Le problème Décodage par Syndrome est NP-complet [BMvT78]. Le cas des codes cycliques n'est pas connu. Guruswami et Sudan on récemment démontré que le décodage des codes de Reed-Solomon est aussi NP-complet [GV05]. Or les codes de Reed-Solomon peuvent être vus comme des codes cycliques pour lesquels $n|q-1$, c'est-à-dire des codes définis sur $\mathbb{F}_q$ de longueur n telle que le corps de décomposition de $X^n - 1$ est $\mathbb{F}_q$. Cela pourrait indiquer que le décodage de certains codes cycliques est un problème difficile. Mais les codes de Reed-Solomon intervenant dans la preuve sont tels que la taille de l'alphabet est très grande par rapport à la longueur. Cela ne dit donc rien d'intéressant relativement au problème du décodage des codes cycliques binaires, par exemple.

8.2 Les équations de Newton

Il y a à distinguer le cas binaire $q = 2$, et le cas dit q-aire. La différence avec le cas q-aire, en ce domaine est fondamentale : les syndromes, qui sont dans le cas q-aire certains coefficients de la transformée de Fourier, deviennent les fonctions puissances des localisateurs de l'erreur, que nous définissons comme suit. Dans le cas binaire, si par exemple $e = (e_0, \ldots, e_{n-1})$ est l'erreur, alors les localisateurs de e sont les $Z_i^* = \alpha^{j_i}$, où $j_1, \ldots, j_w$ sont les indices des positions non nulles de e, et

où w est le poids de l'erreur. Par exemple, si $e = (0, 1, 0, 0, 1)$, le poids de e est 2, et les localisateurs sont $Z_1^* = \alpha^1$ et $Z_2^* = \alpha^4$. Nous avons alors

$$S_j^* = \sum_{i=1}^{w} e_i \cdot (\alpha^j)^i = \sum_{i|e_i \neq 0} (\alpha^i)^j = \sum_{i=1}^{w} (Z_i^*)^j, \quad j \in \{1, \ldots, n\}. \tag{8.5}$$

Dans le cas q-aire, nous notons Y_i^*, $i \in \{1, \ldots, w\}$ les valeurs de e sur ses positions non nulles. Par exemple, soit le mot $e = (0, \beta, 0, 0, \beta + 1)$, défini sur $\mathbb{F}_4$, avec $\beta^2 = \beta + 1$, nous lui associons ses localisateurs comme précédemment, mais nous lui associons aussi $Y_1^* = \beta$, et $Y_2^* = \beta + 1$.

Nous notons $\mathcal{F}$ la transformée de Fourier :

$$\begin{array}{rccc} \mathcal{F}: & \overline{\mathbb{F}_q}^n & \to & \overline{\mathbb{F}_q}^n \\ & e & \mapsto & S^* = (S_0^*, \ldots, S_{n-1}^*) \end{array} \tag{8.6}$$

où nous notons encore $S_j^* = e(\alpha^j)$. On a alors

$$S_j^* = \sum_{i=1}^{w} e_i \cdot (\alpha^j)^i = \sum_{i|e_i \neq 0} e_i (\alpha^i)^j = \sum_{i=1}^{w} Y_i^* (Z_i^*)^j, \quad j \in \{1, \ldots, n\}.$$

Définissons les fonctions symétriques des localisateurs de e :

$$\sigma_i^* = \sum_{1 \leq j_1 < \ldots, j_i \leq w} X_{j_1}^* \cdots X_{j_w}^*, \quad i \in \{1, \ldots, w\}, \tag{8.7}$$

et le polynôme localisateur

$$\sigma^*(Z) = 1 + \sum_{i=1}^{w} \sigma_i^* Z^i = \prod_{i=1}^{w} (1 - Z_i^* Z). \tag{8.8}$$

Les algorithmes de décodage des codes cycliques se focalisent sur le problème de trouver le polynôme localisateur de l'erreur, en fonction des syndromes de l'erreur. Il reste alors le problème de trouver les racines du polynôme localisateur, mais on considère que c'est un problème facile à résoudre en pratique, par la méthode dite de Chien [Chi64], qui est une recherche exhaustive efficace. Donc nous nous concentrerons sur le problème de trouver les σ_i. Le lien entre les syndromes (qui sont les fonctions puissances ou les coefficients de la transformée de Fourier) est donné par les relations de Newton. Dans la notation qui suit, nous enlevons l'astérisque $*$ pour indiquer que l'on considère les systèmes polynomiaux d'indéterminées σ_i et S_i.

Proposition 1. *Soit $e \in \overline{\mathbb{F}}_q[X]/(X^n - 1)$ un mot de poids w, alors sa transformée de Fourier $\underline{S}_n^*$ et les fonctions symétriques élémentaires associées $\underline{\sigma}_w^*$ vérifient la* partie circulaire *des identités de Newton $I_{C,n,w}$*

$$I_{C,n,w} = \left\{ S_{w+i \bmod n} + \sum_{j=1}^{w} \sigma_j S_{w+i-j \bmod n}, \quad i \in \{1, \ldots, n\} \right\}; \tag{8.9}$$

Si, de plus, e est à coefficients 0 ou 1 (c'est le cas lorsque $e \in \mathbb{F}_2[X]/(X^n - 1)$), ses fonctions puissances et symétriques élémentaires vérifient la partie triangulaire *des identités de Newton :*

$$I_{T,w} = \{ S_i + \sum_{j=1}^{i-1} S_{i-j} \sigma_j + i\sigma_i \quad i \in \{1, \ldots, w\} \}. \tag{8.10}$$

Nous numérotons les équations de Newton précisément comme suit :

$$eq_i : \begin{cases} S_i + \sum_{j=1}^{i-1} \sigma_j S_{i-j} + i\sigma_i, & i \leq w, \\ S_i + \sum_{j=1}^{w} \sigma_j S_{i-j}, & i \geq w. \end{cases} \tag{8.11}$$

On peut aussi écrire la partie circulaire des identités de Newton sous la forme polynomiale suivante :

$$S(Z)\sigma_v(Z) = 0 \bmod Z^n - 1, \text{ avec } S(Z) = \sum_{i=1}^{n} S_i Z^{n-i} \text{ et } \sigma_v(Z) = 1 + \sum_{i=1}^{v} \sigma_i Z^i. \tag{8.12}$$

Notons une troisième écriture de la partie circulante des équations de Newton, sous forme matricielle. Soit la matrice C_S :

$$C_S = \begin{pmatrix} S_0 & S_1 & \dots & S_{n-2} & S_{n-1} \\ S_1 & S_2 & \dots & S_{n-1} & S_0 \\ \vdots & & & & \vdots \\ S_{n-1} & S_0 & \dots & S_{n-3} & S_{n-2} \end{pmatrix}. \tag{8.13}$$

Alors $I_{C,n,w}$ s'écrit :

$$C_S \begin{bmatrix} \sigma_w \\ \sigma_{w-1} \\ \vdots \\ \sigma_1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = 0. \tag{8.14}$$

Enfin, dans le cas où le mot e est à coefficients dans $\{0, 1\}$, en utilisant la partie triangulaire des équations de Newton, puis la partie circulante, nous pouvons récursivement exprimer chaque S_i en fonction des σ_i. Par exemple, pour $w = 2$, les équations de Newton sont de la forme :

$$\begin{aligned} S_1 + \sigma_1 &= 0 \\ S_2 + \sigma_1 S_1 + 2\sigma_2 &= 0 \\ S_3 + \sigma_1 S_2 + \sigma_2 S_1 &= 0 \\ S_4 + \sigma_1 S_3 + \sigma_2 S_2 &= 0 \\ &\vdots \end{aligned}$$

On obtient :

$$\begin{aligned} S_1 &= \sigma_1 \\ S_2 &= \sigma_1^2 + 2\sigma_2 \\ S_3 &= \sigma_1^3 + 3\sigma_2\sigma_1 \\ S_4 &= \sigma_1^4 + 4\sigma_2\sigma_1^2 + 2\sigma_2^2 \\ &\vdots \end{aligned}$$

De manière générale, il existe un polynôme $W_i = W_{w,i}(\sigma_1, \ldots, \sigma_w)$ tel que

$$S_i = W_{w,i}(\sigma_1, \ldots, \sigma_w). \tag{8.15}$$

On a même une formule pour les polynômes $W_{w,i}$:

$$W_{w,i}(\sigma_1, \ldots, \sigma_w) = \sum (-1)^{i_2+i_4+i_6+\cdots} \frac{(i_1 + i_2 + \cdots + i_w - 1)!}{i_1! \cdots i_w!} \, i \, \sigma_1^{i_1} \cdots \sigma_w^{i_w}, \tag{8.16}$$

où la somme est prise sur les w-uplets tel que $i_1 + 2i_2 + \cdots + wi_w = i$. Ces formules sont appelées les *formules de Waring* [LN96].

Pour le lecteur non habitué aux corps finis, nous remarquons que la partie triangulaire des équations de Newton ne permet pas d'exprimer facilement les σ_i en fonction des S_i, car des termes $i\sigma_i$ peuvent s'annuler lorsque la caractéristique p divise i. Ce problème ne se pose pas en caractéristique 0, où on peut toujours exprimer facilement les fonctions symétriques élémentaires en terme des fonctions puissances.

Les identités de Newton font intervenir les σ_i, que nous cherchons, les syndromes S_i, $i \in Q$, que nous connaissons, mais aussi les S_i, $i \notin Q$, que nous ne connaissons pas dans le contexte du décodage. On peut aussi voir le système des équations comme un système linéaire en les σ_i, à coefficients les syndromes connus, mais aussi les syndromes inconnus. Nous allons étudier les propriétés de l'idéal des équations de Newton, quand on élimine les syndromes inconnus, dans le but de trouver les σ_i en fonction des syndromes S_i, $i \in Q$.

J'ai présenté, page 21, un exemple introductif, dû à [RYT90], qui illustre bien cette démarche.

8.3 Algorithmes algébriques de décodage des codes cycliques

Il est remarquable que, à la vue seule de l'ensemble de définition Q d'un code cyclique C, on soit souvent capable de borner inférieurement la distance minimale de C. On dispose ainsi de la célèbre borne BCH, mais aussi des bornes de Hartman-Tzeng [HT72], de la borne de Roos [Roo83]. Plus remarquable encore est le fait qu'à chacune de ces bornes soit associé un algorithme de décodage jusqu'à $t = \lfloor \frac{d'-1}{2} \rfloor$, où d' est la borne inférieure que l'on a obtenue sur la distance minimale du code considéré. Dans le cas des codes BCH, on utilise alors l'algorithme de Berlekamp-Massey ou l'algorithme d'Euclide, et on a des généralisations correspondant aux bornes de Hartman-Tzeng et de Roos [FT91]. Ces algorithmes ont la propriété d'être *génériques*, en un certain sens : bien que ne s'appliquant pas à tout code cyclique, ils sont capables de décoder tous les codes dont l'ensemble de définition permet d'établir la borne BCH, la borne de Hartman-Tzeng ou la borne de Roos.

Toutefois, lorsque l'ensemble de définition du code cyclique C est quelconque, on n'a pas d'algorithme pour décoder C, même si l'on est capable de borner inférieurement la distance minimale avec des algorithmes génériques [MS88]. Ainsi, il n'existe pas d'algorithme algébrique général pour décoder tout code cyclique. Un des cas les plus importants est celui des codes à résidus quadratiques qui sont des codes spécifiques pour lesquels on n'a pas d'algorithme de décodage général bien que l'on soit capable de borner a priori la distance minimale, avec la fameuse *square-root bound* [PHB98]. Tout au plus dispose-t-on d'algorithmes de décodage pour chaque cas particulier : nous citerons les références [RYT90, RYTH90, RTCY92, CRT94, LWCL95, HRTC01, CTR+03, TCCL05] qui montrent bien qu'un nouvel algorithme de décodage est à concevoir pour chaque longueur de code à résidus quadratiques considéré, à savoir les longueurs 31, 23, 41, 73, 47, 71, 79, 97, et enfin 103 et 113.

Dans le cas des codes à résidus quadratiques non binaires, les algorithmes sont encore dédiés [Hum92, HH93, HH95, HH98]. D'autres codes, dit de Zettelberg, sont encore considérés dans [DN93, DN92], avec des algorithmes encore Ad Hoc.

Quand la longueur des codes à résidus quadratiques croît, la taille des formules croît, et souvent les auteurs échouent à obtenir des relations de degré 1. Par exemple, pour le code à résidus quadratiques de longueur 41, de capacité de correction 4, les auteurs [RTCY92] obtiennent une relation de degré 4 pour σ_2, en fonction des syndromes connus. Il faut alors factoriser cette relation, à chaque décodage, pour trouver 4 valeurs possibles pour σ_2, que l'on teste ensuite.

Dans certains cas, la réalisation matérielle (hardware) est envisagée [DN94].

Nous faisons notre étude en deux étapes : nous étudions la variété associée à l'idéal engendré par les équations de Newton, et ensuite nous étudions l'idéal défini par les équations. Nous pensons que cet idéal est radical, mais nous n'avons pu le prouver. Nous introduisons un idéal qui le contient, et qui a la propriété d'être radical. Enfin nous montrons que ce nouvel idéal contient bien des formules de degré 1 qui peuvent être utilisées pour le décodage.

8.4 Les travaux sur le décodage avec les bases de Gröbner

L'idée de décoder les codes cycliques avec les bases de Gröbner a été introduite par Chen, Reed, Helleseth et Truong [CRHT94d, CRHT94c]. Ils n'utilisent pas les relations de Newton, mais directement l'expression des syndromes en termes des localisateurs donnée par

$$S_j^* = \sum_{i=1}^{w} e_i \cdot (\alpha^j)^i = \sum_{i|e_i \neq 0} (\alpha^i)^j = \sum_{i=1}^{w} (Z_i)^j, \quad j \in Q; \tag{8.17}$$

où les S_j^*, $j \in Q$ sont les syndromes (connus) du mot reçu, et les Z_i, $i \in \{1, \ldots, v\}$ sont les localisateurs (inconnus) que l'on cherche à déterminer. Ils considèrent un décodage avec syndromes spécialisés : c'est-à-dire que pour chaque mot reçu on calcule une base de Gröbner. Ils ont montré que, pour l'ordre lexicographique, une base de Gröbner de l'idéal engendré par ces relations contient le polynôme localisateur de l'erreur. Ce système a ensuite été étudié par Loustaunau et von York [LY97], avec des syndromes non spécialisés, pour obtenir des polynômes tels que, lorsque les coefficients sont spécialisés, on obtienne le polynôme localisateur.

Ces systèmes ont été considérés pour le problème de déterminer la distance minimale d'un code cyclique [MS03, Sal02, Sal03]. Caboara et Mora [CM02] affinent l'étude de la base de Gröbner du système (8.17), avec des syndromes symboliques, pour améliorer l'analyse de Loustaunau et von York.

À notre connaissance, nos travaux sont les seuls étudiant le système des équations de Newton.

8.5 Élimination et spécialisation

Pour tenter d'avoir une notation claire, nous mettrons systématiquement une astérisque lorsque nous parlerons d'une valeur donnée $S_i^* \in \mathbb{F}_{q^m}$, par exemple un syndrome donné d'une erreur donnée; par opposition au symbole S_i qui servira d'indéterminée dans des systèmes d'équations. De même, nous différencierons $\sigma_i^* \in \mathbb{F}_{q^m}$ de l'indéterminée σ_i, et $Z_i^* \in \mathbb{F}_{q^m}$ de l'indéterminée Z_i. Nous dirons aussi que S_i^* (resp. σ_i^*, Z_i^*) est une spécialisation de S_i (resp. σ_i, Z_i). Nous notons de plus $\underline{S}_n$ (respectivement $\underline{\sigma}_v$) le vecteur $(S_0, \ldots, S_{n-1})$ (respectivement $(\sigma_1, \ldots, \sigma_v)$.)

Enfin, pour des énoncés généraux, nous utiliserons les indéterminées $x_1, \ldots, x_n$.

Théorème 1. *Soit un idéal $I \in \mathbb{F}[x_1, \ldots, x_n]$, et $I_k = I \cap \mathbb{F}[x_{k+1}, \ldots, x_n]$ le k-ième idéal d'élimination. Soit G une base de Gröbner de I pour un ordre éliminant $x_1, \ldots, x_k$, alors*

$$G \cap \mathbb{F}[x_1, \ldots, x_n]$$

est une base de Gröbner de I_k.

Théorème 2. *Soit un idéal $I \in \mathbb{F}[x_1, \ldots, x_n]$, et V la variété associée, i.e. l'ensemble des zéros communs des polynômes de I. Soit $V_k = \mathrm{Pr}_k(V)$, la projection de V sur les $n-k$ dernières coordonnées, et $I_k = I \cap \mathbb{F}[x_{k+1}, \ldots, x_n]$ le k-ième idéal d'élimination. Alors*

$$V(I_k) = \overline{V_k}, \tag{8.18}$$

où $\overline{V_k}$ est la fermeture de V_k.

Corollaire 1. *Soit un idéal $I \in \mathbb{F}[x_1, \ldots, x_n]$, I_k le k-ième idéal d'élimination, V_k la projection de $V(I)$ sur les $n-k$ dernières coordonnées. Si $V(I_k)$ ou V_k est finie, alors*

$$V(I_k) = V_k.$$

Autrement dit, toute solution partielle $(x^*_{k+1}, \ldots, x^*_n) \in V(I \cap \mathbb{F}[x_{k-1}, \ldots, x_n])$ s'étend en au moins une solution $(x_1, \ldots, x_n) \in V(I)$.

Soit $f \in \mathbb{F}[y, \underline{x}]$, où $\underline{x} = (x_1, \ldots, x_n)$. Nous notons $LT(f)$ le terme de tête d'un polynôme f, $LT(I)$ l'ensemble des termes de tête d'un idéal I, et $LT_y(f)$ le terme de tête de f vu comme un polynôme de $\mathbb{F}_q[y][\underline{x}]$.

Théorème 3 (Théorème de spécialisation, cas zéro-dimensionnel [Gia89]). *Soit $I \subset \mathbb{F}[y, \underline{x}]$ un idéal de dimension zéro, et φ_{x^*} une spécialisation $\underline{x} \mapsto \underline{x}^*$ de toutes les variables sauf une :*

$$\varphi_{x^*} : \mathbb{F}[y, \underline{x}] \to \mathbb{F}[y]. \tag{8.19}$$

Alors il existe un polynôme $g \in I$, tel que $\varphi_{x^*}(g)$ engendre $\varphi_{x^*}(I)$, et $\deg_y g = \deg_y \varphi_{x^*}(g)$.

Proposition 2 ([FGT01]). *Soit $\mathbb{F}[y, \underline{x}] = \mathbb{F}[y, x_1, \ldots, x_n]$, et soit G une base de Gröbner d'un idéal $I \subset \mathbb{F}[y, \underline{x}]$, et soit φ_{x^*} une spécialisation $\underline{x} \mapsto \underline{x}^*$ de toutes les variables sauf y. Pour $g \in I$, notons $\delta(g) = \deg_y g - \deg_y \varphi_{x^*}(g)$, la chute de degré de g, et $\delta(I) = \min_{g \in I} \delta(g)$.*

Soit $g_0 \in G$ de degré minimal en y tel que $\varphi_{x^*}(g_0) \neq 0$. Alors $\varphi_{x^*}(g_0)$ engendre $\varphi_{x^*}(I)$, et $\delta(I) = \delta(g_0)$. De plus, s'il existe $h \in I$ tel que $\varphi_{x^*}(LT_y(h)) \neq 0$, alors $\deg_y g_0 = \deg_y \varphi_{x^*}(g_0)$, et $LT(\varphi_{x^*}(I)) = \varphi_{x^*}(LT_x(I))$.

8.6 La variété associée à l'idéal des équations de Newton

La transformée de Fourier vérifie la propriété suivante, relative au poids, suivant ce au'énonce le théorème suivant, souvent référencé comme le théorème de Blahut. Nous le donnons dans le contexte de la clôture algébrique de $\mathbb{F}_q$.

Théorème 4. *Soit $S^*(Z) = \sum_{i=1}^{n} S^*_i Z^{n-i}$ la transformée de Fourier de $c \in \overline{\mathbb{F}}_q^n$. Alors le poids de c est égal au rang de la matrice circulante suivante C_{S^*} :*

$$C_{S^*} = \begin{pmatrix} S^*_0 & S^*_1 & \ldots & S^*_{n-2} & S^*_{n-1} \\ S^*_1 & S^*_2 & \ldots & S^*_{n-1} & S^*_0 \\ \vdots & & & & \vdots \\ S^*_{n-1} & S^*_0 & \ldots & S^*_{n-3} & S^*_{n-2} \end{pmatrix}. \tag{8.20}$$

Démonstration. Ce théorème est bien connu, dans le cas où $c \in \mathbb{F}_q$. Je redonne une preuve fonctionnant dans le cas de la clôture algébrique. Écrivons $c = (c_0, \ldots, c_{n-1})$. On a

$$\begin{pmatrix} S_i^* \\ S_{i+1}^* \\ \vdots \\ S_{i+n-1}^* \end{pmatrix} = F \begin{pmatrix} c_0 \\ \alpha^i c_1 \\ \vdots \\ \alpha^{(n-1)i} c_{n-1} \end{pmatrix},$$

avec

$$F = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ 1 & \alpha^1 & \ldots & \alpha^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{n-1} & \ldots & \alpha^{(n-1)(n-1)} \end{pmatrix}.$$

Alors :

$$\begin{aligned} C_{S^*} &= F \begin{pmatrix} c_0 & c_0 & \ldots & c_0 \\ c_1 & \alpha c_1 & \ldots & \alpha^{n-1} c_1 \\ \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & \alpha^{n-1} c_{n-1} & \ldots & \alpha^{(n-1)(n-1)} c_{n-1} \end{pmatrix} \\ &= F \begin{pmatrix} c_0 & 0 & & \ldots \\ 0 & c_1 & 0 & \ldots \\ \vdots & & \ddots & \\ 0 & \ldots & 0 & c_{n-1} \end{pmatrix} F. \end{aligned}$$

Maintenant F est une matrice de Vandermonde inversible, et le rang de la matrice intérieure est égal au poids de c. □

Le théorème suivant est une sorte de réciproque des relations de Newton, et fait le lien entre le poids des solutions du système et le poids v pour lequel est écrit le système. De plus, il décrit la très forte structure des solutions du système des équations de Newton.

Théorème 5. *Soit $\underline{S}^* \in \overline{\mathbb{F}}_q^n$ et e la transformée de Fourier inverse de $\underline{S}^*$.*

1. La partie circulante du système spécialisé des équations de Newton $I_{C,n,v}(\underline{S} \mapsto \underline{S}^*)$ a une solution

$$\underline{\rho}^* = (\rho_1^*, \ldots, \rho_v^*) \in \overline{\mathbb{F}}_2^v,$$

si et seulement si le poids w de e est inférieur ou égal à v.

2. Soit $\underline{\rho}^*$ une solution comme précédemment, si on note

$$\sigma^*(Z) = \prod_{j=1}^{w} (1 - Z_j^* Z) = 1 + \sum_{i=1}^{w} \sigma_i^* Z^i,$$

le polynôme localisateur de e, les Z_j^* étant les localisateurs de e, et si on construit

$$\rho^*(Z) = 1 + \sum_{i=1}^{v} \rho_i^* Z^i,$$

à partir de $\underline{\rho}^*$, alors $\rho^*(Z)$ est multiple de $\sigma^*(Z)$.

3. Dans le cas où q est une puissance de 2, si $\underline{S}^*$ et $\underline{\rho}^*$ sont de plus solutions de la partie triangulaire $I_{T,v}(\underline{S} \mapsto \underline{S}^*)$, alors e est un mot à coefficients 0 ou 1, et il existe $G(Z) \in \overline{\mathbb{F}}_2[Z]$, et un entier $l \geq 0$, tels que $\rho^*(Z)$ est de la forme

$$\rho^*(Z) = \sigma^*(Z) G(Z)^2 Z^l. \tag{8.21}$$

Démonstration. 1. Prouvons que d'abord s'il existe une solution $\underline{\rho}^*$ à $I_{C,n,v}(\underline{S} \mapsto \underline{S}^*)$, alors le poids w de e vérifie $w \leq v$. Soit C_{S^*} la matrice construite à partir de la transformée de Fourier de e comme dans (8.20). À partir de l'expression matricielle de la partie circulante des équations de Newton, on vérifie que la solution $(\rho_1^*, \ldots, \rho_v^*)$ conduit à une expression de la $(v+1)$-ième colonne de la matrice en termes des v premières colonnes. De même, la $(v+2)$-ième colonne s'exprime linéairement en termes des v colonnes précédentes. La même propriété est vérifiée pour la $(v+i)$-ième colonne de la matrice, pour $i \in \{1, \ldots, n-v\}$. Ainsi les v premières colonnes engendrent l'espace de toutes les colonnes. La rang de la matrice est donc inférieur ou égal à v, et le poids de e est donc inférieur à v d'après le théorème 4.

Réciproquement, si le poids de e est w, avec $w \leq v$, alors les fonctions symétriques élémentaires $\sigma_1^*, \ldots, \sigma_w^*$ des localisateurs de e sont solutions de la partie circulante des équations de Newton pour le poids w, et on vérifie, à partir de (8.14) que

$$(\sigma_1^*, \ldots, \sigma_w^*, \sigma_{w+1}^* = 0 \ldots, \sigma_v^* = 0)$$

est solution de $I_{C,n,v}(\underline{S} \mapsto \underline{S}^*)$.

2. Soit $F \subset \overline{\mathbb{F}}_q^v$ l'ensemble des solutions $\underline{\rho}_v^*$ de $I_{C,n,v}(\underline{S} \mapsto \underline{S}^*)$. Alors comme le rang de C_{S^*} est w, la forme matricielle des équations de Newton (8.14) permet de dire que F est un espace affine de dimension $v - w$. Soit F' l'espace

$$F' = \left\{ \underline{\rho}^* = (\rho_1^*, \ldots, \rho_v^*) \in \overline{\mathbb{F}}_q^v \mid \sigma^*(Z) \text{ divise } \rho^*(Z) = 1 + \sum_{i=1}^{v} \rho_i^* Z^i \right\},$$

alors F' est aussi un espace affine et $\dim F' = v - w$. Soit $\underline{\rho}^* \in F'$ et $\rho^*(Z)$ le polynôme construit à partir de $\underline{\rho}^*$, comme $\sigma^*(Z) | \rho^*(Z)$ et $S^*(Z)\sigma^*(Z) = 0 \bmod Z^n - 1$, on a

$$S^*(Z)\rho^*(Z) = 0 \bmod Z^n - 1,$$

c'est-à-dire $(\rho_1^*, \ldots, \rho_v^*) \in F$. Donc $F' \subset F$. Comme F' et F ont même dimension, ils sont égaux.

3. Soit $\sigma^*(Z)$ le polynôme localisateur de e, et $\rho^*(Z)$ comme dans l'énoncé du théorème, alors le point *2.* du théorème donne $\sigma^*(Z) | \rho^*(Z)$. Soit donc $Z_1^*, \ldots, Z_w^*$ les localisateurs de e, et $Z_{w+1}^* \ldots, Z_v^*$ les racines de $\rho^*(Z)$ qui ne sont pas racines de $\sigma^*(Z)$. Comme $\underline{S}^*$ et $\underline{\rho}^*$ sont racines des équations de Newton triangulaires, ils vérifient les formules de Waring

$$S_i^* = W_{v,i}(\rho_1^*, \ldots, \rho_v^*), \quad i \in \{1, \ldots, n\} \tag{8.22}$$

et comme les ρ_i^* sont les fonctions symétriques élémentaires de $Z_1^*, \ldots, Z_v^*$, cela donne :

$$S_i^* = \sum_{j=0}^{v} {Z_j^*}^i, \quad i \in \{1, \ldots, n\}. \tag{8.23}$$

D'autre part, $\underline{S}^*$ est la transformée de Fourier de e. Soit Y_j^*, $j \in \{1, \ldots, w\}$ le coefficient de e en la position Z_j^*, avec $Y_j^* \neq 0$. Le calcul de la transformée de Fourier donne

$$S_i^* = \sum_{j=1}^{w} Y_j^* {Z_j^*}^i, i \in \{1, \ldots, n\}. \tag{8.24}$$

En égalisant les termes de droite de (8.23) et (8.24), on obtient la relation matricielle

$$\begin{bmatrix} Z_1^* & \dots & Z_w^* & Z_{w+1} & \dots & Z_v^* \\ {Z_1^*}^2 & \dots & {Z_w^*}^2 & {Z_{w+1}}^2 & \dots & {Z_v^*}^2 \\ \vdots & & & & & \vdots \\ {Z_1^*}^n & \dots & {Z_w^*}^n & {Z_{w+1}^*}^n & \dots & {Z_v^*}^n \end{bmatrix} \begin{bmatrix} Y_1^* + 1 \\ \vdots \\ Y_w^* + 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = 0. \tag{8.25}$$

En considérant la matrice constituée des v premières lignes de la matrice de droite, nous avons une matrice de Vandermonde, de déterminant

$$\left(\prod_{i=1}^{v} Z_i^*\right) \left(\prod_{1 \le j_1 < j_2 \le v} (Z_{j_2}^* - Z_{j_1}^*)\right),$$

qui doit être nul. Cela peut se produire de trois manières, compte tenu du fait que les premiers Z_j^*, $j \in \{1, \dots, w\}$, sont distincts et non nuls :

1. soit l'un des Z_j^* est nul, pour $w+1 \le j \le v$;
2. soit $Z_{j_1}^* = Z_{j_2}^*$, pour $w+1 \le j_1, j_2 \le v$;
3. soit $Z_{j_1}^* = Z_{j_2}^*$, avec $1 \le j_1 \le w$ et $w+1 \le j_2 \le v$.

Dans le cas 1, on peut réécrire (8.25), en enlevant la j-ième colonne de la matrice. On constate que $Z_j^* = 0$ donne un facteur Z de $\rho^*(Z)$.

Dans le cas 2, l'effet de la caractéristique 2 est d'annuler les termes $Z_{j_1}^*$ et $Z_{j_2}^*$ dans l'équation (8.25), que l'on peut réécrire en enlevant la j_1-ième et la j_2-ième colonne. On constate que $Z_{j_1}^* = Z_{j_2}^*$ contribue à l'apparition d'un facteur carré dans $\rho^*(Z)$.

Dans le cas 3, les colonnes j_1 et j_2 vont s'additionner, et on obtient une relation

$$\begin{bmatrix} Z_1^* & \dots & Z_w^* & Z_{w+1} & \dots & Z_v^* \\ {Z_1^*}^2 & \dots & {Z_w^*}^2 & {Z_{w+1}}^2 & \dots & {Z_v^*}^2 \\ \vdots & & & & & \vdots \\ {Z_1^*}^n & \dots & {Z_w^*}^n & {Z_{w+1}^*}^n & \dots & {Z_v^*}^n \end{bmatrix} \begin{bmatrix} Y_1^* + 1 \\ \vdots \\ Y_{j_1}^* \\ \vdots \\ Y_w^* + 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = 0. \tag{8.26}$$

où la colonne j_2 a disparu de la matrice de gauche, et où le vecteur de droite a un terme en 1 de moins.

Dans les trois cas 1, 2, et 3, on a diminué la taille de la matrice de gauche, en éliminant des colonnes faisant intervenir des Z_j^*, avec $w+1 \le j \le v$. En répétant le processus, on obtient une relation :

$$\begin{bmatrix} Z_1^* & \dots & Z_w^* \\ {Z_1^*}^2 & \dots & {Z_w^*}^2 \\ \vdots & \vdots & \\ {Z_1^*}^n & \dots & {Z_w^*}^n \end{bmatrix} \begin{bmatrix} Y_1^* + \epsilon_1 \\ \vdots \\ Y_w^* + \epsilon_w \end{bmatrix} = 0, \tag{8.27}$$

qui ne contient plus les Z_j^*, $w+1 \le j \le v$, et où $\epsilon_i = 0$ ou 1 (effet de la caractéristique 2). Le déterminant de la sous-matrice issue de w premières lignes est égal à

$$\left(\prod_{i=1}^{w} Z_i^*\right) \left(\prod_{1 \le j_1 < j_2 \le w} (Z_{j_2}^* - Z_{j_1}^*)\right),$$

qui est non nul car les Z_j^*, $j \in \{1, \ldots, w\}$ sont distincts et non nuls. On a donc

$$\begin{bmatrix} Y_1^* + \epsilon_1 \\ \vdots \\ Y_w^* + \epsilon_w \end{bmatrix} = 0,$$

ce qui entraîne $Y_i^* = \epsilon_i$, $i \in \{1, \ldots, w\}$, ce qui n'est possible que si $\epsilon_i = 1$ et $Y_i = 1$, $i \in \{1, \ldots, w\}$, car les Y_i sont non nuls : e est donc bien un mot binaire. Dans le processus qui a conduit à la forme (8.27), avec $\epsilon_i = 1$, $i \in \{1, \ldots, w\}$, on note qu'à chaque étape on a eu

- soit l'existence d'un indice j, $j \in \{(w+1), \ldots, v\}$, tel que $Z_j^* = 0$ qui contribue au facteur Z^l annoncé dans l'équation 8.21 de l'énoncé du théorème ;
- soit deux indices $j_1, j_2 \in \{(w+1), \ldots, v\}$ tel que $Z_{j_1}^* = Z_{j_2}^*$, ce qui contribue au facteur $G(Z)^2$ de l'équation 8.21 de l'énoncé du théorème ;
- soit $j_1 \in \{1, \ldots, w\}$ et $j_2 \in \{(w+1), \ldots, v\}$ tels que $Z_{j_1}^* = Z_{j_2}^*$, ce qui change la valeur de ϵ_{j_1} en $\epsilon_{j_1} + 1$. Pour j_1 donné, comme ϵ_{j_1} finit par être égal à 1 au bout du processus, le cas où il existe j_2 tel que $Z_{j_2}^* = Z_{j_1}^*$ n'a pu se produire qu'un nombre pair de fois, ce qui contribue au facteur $G(Z)^2$ de l'équation 8.21 de l'énoncé du théorème.

□

Corollaire 2. *Soit $I = I_{C,n,v} \cap \mathbb{F}_q[\underline{S}]$ l'idéal d'élimination des fonctions symétriques. Alors $V(I)$ est l'ensemble des transformées de Fourier des mots de poids inférieur ou égal à v. On a de plus* $\dim V(I) = v$.

Soit $J = (I_{C,n,v} + I_{t,v}) \cap \mathbb{F}_q[\underline{S}]$. Alors $V(J)$ est l'ensemble des fonctions symétriques élémentaires des mots à coefficients 0 *ou* 1 *de poids au plus v. De plus* $\dim V(J) = 0$.

Démonstration. Dans le premier cas, on a $V(I) = \overline{\Pr(V)}$ où $\Pr(V)$ est la projection, sur l'espace des transformées de Fourier, de la variété $V = V(I_{C,n,v})$. Or $Pr(V)$ est l'ensemble des transformées de Fourier des mots de poids inférieur ou égal à v. Nous montrons que c'est un fermé. En effet, d'après le théorème 4, le mot e est de poids inférieur ou égal à v si et seulement si la matrice

$$C_{S^*} = \begin{pmatrix} S_0^* & S_1^* & \ldots & S_{n-2}^* & S_{n-1}^* \\ S_1^* & S_2^* & \ldots & S_{n-1}^* & S_0^* \\ \vdots & & & & \vdots \\ S_{n-1}^* & S_0^* & \ldots & S_{n-3}^* & S_{n-2}^* \end{pmatrix}. \tag{8.28}$$

est de rang inférieur à v. Cela se produit quand tous les mineurs d'ordre $v+1$ sont nuls, ce qui constitue bien un ensemble d'équations algébriques sur les S_i. Donc $V(I) = \overline{\Pr(V)} = \Pr(V)$.

La variété $V = V(I_{C,n,v} \cap \mathbb{F}_2[\underline{S}])$ est l'ensemble des transformée de Fourier des mots de $\overline{\mathbb{F}}_q^n$ de poids inférieur à w. Soit $W = \mathcal{F}^{-1}(V)$, l'image inverse de V par la transformée de Fourier. Alors W est une variété de même dimension que V, car V est algébrique et $\mathcal{F}$ est linéaire inversible . Soit $S = \{i_1, \ldots, i_w\} \subset \{1, \ldots, n\}$, de cardinal w, avec $w \leq v$. Alors l'ensemble des mots $e \in \overline{\mathbb{F}}_q^n$ de support S est l'ensemble

$$W_S = \left\{ e \in \overline{\mathbb{F}}_q^n ;\ e_j = 0,\ j \notin S \right\}, \tag{8.29}$$

qui est une variété de dimension w (c'est un sous-espace vectoriel de dimension w). Nous avons enfin

$$W = \bigcup_{S, |S| \leq v} W_S, \tag{8.30}$$

qui est une variété de dimension v, en tant qu'union de variétés de dimension inférieure ou égale à v, dont au moins une est de dimension v.

Dans le deuxième cas, l'ensemble des mots binaires de poids inférieur à v est fini. Donc les fonctions puissance associées sont en nombre fini. Le corollaire 1 s'applique directement. □

8.7 Propriétés algébriques de l'idéal des équations de Newton

Dans cette partie, nous allons étudier la radicalité de l'idéal des équation de Newton. Dans un premier temps, nous considérons l'idéal des équations de Newton (triangulaires et circulantes) auquel on adjoint les équations de corps. Nous noterons donc $I^0_{N,n,v}$ le système :

$$I^0_{N,n,v} = \left\{ \begin{array}{ll} S_i + \sum_{j=1}^{i-1} \sigma_j S_{i-j} + i\sigma_i, & i \in \{1,\ldots,v\} \\ S_{v+i} + \sum_{j=1}^{v} \sigma_j S_{v+i-j}, & i \in \{1,\ldots,n\} \\ S_{n+i} + S_i, & i \in \{1,\ldots,v\} \\ \sigma_j^{q^m} - \sigma_j, & j \in \{1,\ldots,v\} \\ S_j^{q^m} - S_j, & j \in \{1,\ldots,n\} \end{array} \right\} \subset \mathbb{F}_2[\underline{S}, \underline{\sigma}_v]. \tag{8.31}$$

où $\mathbb{F}_{q^m}$ est le corps de décomposition de $X^n - 1$ sur $\mathbb{F}_q$.

Nous utiliserons la proposition suivante.

Proposition 3. *Soit I un idéal de $\mathbb{F}[x_1,\ldots,x_n]$, de dimension zéro. Si pour chaque chaque indéterminée x_i, $i \in \{1,\ldots,n\}$, I contient un polynôme de $\mathbb{F}[x_i]$ sans facteurs multiples, alors I est radical.*

Démonstration. C'est une conséquence directe de la proposition 2.7 du chapitre 2 de [CLO05]. □

Corollaire 3. *L'idéal $I^0_{N,n,v}$ est radical et de dimension 0.*

Une de nos contributions est de montrer que l'on manipuler des idéaux sans introduire les équations de corps, tout en gardant les bonnes propriétés de radicalité. En effet les équations de corps sont du type $S_i^{q^m} - S_i$, où F_{q^m} est le corps de décomposition de $X^n - 1$ sur $\mathbb{F}_q$, et q^m peut être élevé, même pour des codes de longueur modeste. Par exemple, pour le code à résidus quadratiques de longueur 41, le cors de décomposition de $X^{41} - 1$ est $\mathbb{F}_{2^{20}}$. L'idéal $I^0_{N,n,v}$ contient les équations $S_i^{2^{20}} - S_i$. Nous notons $I_{N,n,v}$ l'idéal sans les équations de corps :

$$I_{N,n,v} = \left\{ \begin{array}{ll} S_i + \sum_{j=1}^{i-1} \sigma_j S_{i-j} + i\sigma_i, & i \in \{1,\ldots,v\} \\ S_{v+i} + \sum_{j=1}^{v} \sigma_j S_{v+i-j}, & i \in \{1,\ldots,n\} \\ S_{n+i} + S_i, & i \in \{1,\ldots,v\} \end{array} \right\} \subset \mathbb{F}_2[\underline{S}, \underline{\sigma}_v]. \tag{8.32}$$

Nous allons étudier les idéaux auxiliaires suivant :

Définition 1. *Soit $A = \mathbb{F}_q[Z_1,\ldots,Z_v,\sigma_1,\ldots,\sigma_v,A_1,\ldots,A_n,\ldots A_{n+v}]$, nous définissons les idéaux suivants de A : l'idéal des fonctions symétriques :*

$$I_{\sigma,v} = \left\langle \sigma_i - \sum_{1 \le j_1 < \cdots < j_i \le v} Z_{j_1} \ldots Z_{j_i}; i \in \{1,\ldots,v\} \right\rangle; \tag{8.33}$$

l'idéal des fonctions puissances :

$$I_{S,n,v} = \left\langle \begin{array}{ll} S_i - \sum_{j=1}^{v} Z_j^i, & i \in \{1,\ldots,n+v\}; \\ S_{i+n} - S_i, & i \in \{1,\ldots,v\} \end{array} \right\rangle. \tag{8.34}$$

Il faut bien noter la présence des relations $S_{i+n} - S_i$ qui reflètent que les Z_i^*, $i \in \{1, \ldots, v\}$, sont des racines n-ièmes de l'unité.

Nous aurons besoin du théorème suivant.

Théorème 6 (Machi-Valibouze [Val95]). *Soit* $f(Z) = Z^v + \sum_{i=1}^{v} \sigma_i Z^{v-i} \in \mathbb{F}[\sigma_v][Z]$. *Soient les polynômes* f_i, $i \in \{1, \ldots, v\}$, *construits de proche en proche comme suit* (modules de Cauchy) *:*

$$f_1(Z_1) = f(Z_1), \tag{8.35}$$

$$f_{i+1}(Z_1, \ldots, Z_{i+1}) = \frac{f_i(Z_1, \ldots, Z_{i-1}, Z_i) - f_i(Z_1, \ldots, Z_{i-1}, Z_{i+1})}{Z_i - Z_{i+1}}. \tag{8.36}$$

Alors pour $i \in \{1, \ldots, v\}$, $f_i \in \mathbb{F}[\underline{\sigma}_v][Z_1, \ldots, Z_i]$. *De plus, en considérant l'ordre lexicographique* $Z_v > Z_{v-1} > \cdots > Z_1 > \sigma_v > \cdots > \sigma_1$, f_i *a un terme de tête en* Z_i *uniquement, et* $G_{\sigma,v} = \{f_1, \ldots, f_v\}$ *est une base de Gröbner de* $I_{\sigma,v}$.

Proposition 4. $(I_{S,n,v} + I_{\sigma,v}) \cap \mathbb{F}_q[\underline{S}_n, \underline{\sigma}_v] = I_{N,n,v}$.

Démonstration. Nous pouvons déduire les relations de Newton (triangulaires et circulantes) à partir de la définition des fonctions puissances et des fonctions symétriques élémentaires. Nous avons donc

$$I_N \subset (I_S + I_\sigma) \cap \mathbb{F}_q[\sigma_1, \ldots, \sigma_v, S_1, \ldots, S_n, \ldots, S_{n+v}]. \tag{8.37}$$

Pour prouver l'inclusion inverse, nous procédons en deux temps : nous prouvons d'abord que $I_S + I_\sigma = I_N + I_\sigma$, et ensuite que

$$(I_N + I_\sigma) \cap \mathbb{F}[\sigma_1, \ldots, \sigma_v, S_1, \ldots, S_{n+v}] = I_N. \tag{8.38}$$

Les formules de Waring sont déduites des identités de Newton :

$$S_i - W_{v,i}(\sigma_1, \ldots, \sigma_v) = 0 \bmod I_N, \quad i \in \{1, \ldots, n+v\}. \tag{8.39}$$

Notons s_i, $i \in \{1, \ldots, v\}$, les polynômes $s_i = \sum_{1 \leq j_1 < \cdots < j_i \leq v} Z_{j_1} \ldots Z_{j_i}$, alors $\sigma_i - s_i \in I_\sigma$, et$S_i - W_{v,i}(s_1, \ldots, s_v) \in I_N + I_\sigma$.

Enfin, notons p_i, $i \in \{1, \ldots, n+v\}$, le polynôme $\sum_{j=1}^{v} Z_j^i$, alors, d'après les formules de Waring : $p_i = W_{v,i}(s_1, \ldots, s_v)$, ce qui entraîne

$$S_i - p_i \in I_N + I_\sigma, \quad i \in \{1, \ldots, n+v\},$$

i.e. $I_S \subset I_N + I\sigma$. Donc $I_S + I_\sigma \subset I_N + I_\sigma$ et l'égalité $I_S + I_\sigma = I_N + I_\sigma$ s'ensuit. Nous prouvons maintenant (8.38). Soit G_N une base de Gröbner de I_N, pour un ordre quelconque, et soit G_σ la base de Gröbner de I_σ décrite dans le théorème 6. Alors

$$G_N \cup G_\sigma$$

est une base de Gröbner de $I_N + I_\sigma$. En effet, d'après le théorème 6, G_σ ne contient aucun polynôme dont le terme de tête contient une des indéterminées σ_j. Donc les termes de tête des polynômes de G_N et G_σ sont étrangers, $G_N \cup G_\sigma$ est bien une base de Gröbner. Le théorème d'élimination des bases de Gröbner entraîne que

$$(G_N \cup G_\sigma) \cap \mathbb{F}_2[\sigma_1, \ldots, \sigma_n, A_1, \ldots, A_{n+v}] \tag{8.40}$$

est une base de Gröbner de $(I_N + I_\sigma) \cap \mathbb{F}_2[\sigma_1, \ldots, \sigma_v, S_1, \ldots, S_{n+v}]$. Comme

$$G_\sigma \cap \mathbb{F}_2[\sigma_1, \ldots, \sigma_v] = \{0\},$$

G_N est une base de Gröbner de $(I_N + I_\sigma) \cap \mathbb{F}_2[\sigma_1, \ldots, \sigma_v, S_1, \ldots, S_{n+v}]$.

□

En pratique on a toujours observé que l'idéal $I_{N,n,v}$ est radical, et c'est celui-là que nous avons utilisé pour les expérimentations. Nous allons construire un idéal plus gros que $I_{N,n,v}$, mais qui présente plus de propriétés algébriques, et pour lequel nous aurons beaucoup de propriétés intéressantes. Nous avons besoin de la notion d'idéal saturé, pour éliminer de $I_{N,n,v}$ les composantes correspondant au mots de poids strictement inférieurs à v.

Définition 2. *Soit $I \subset \mathbb{F}[x_1, \ldots, x_n]$ un idéal, et soit $f \in \mathbb{F}[x_1, \ldots, x_n]$ fixé. La saturé de I par rapport à f est l'idéal*

$$I : f^\infty = \{g \in \mathbb{F}[x_1, \ldots, x_n] : f^m g \in I \text{ pour un } m > 0\} \tag{8.41}$$

Proposition 5. *Soit $I = \langle f_1, \ldots, f_s \rangle \subset \mathbb{F}[x_1, \ldots, x_n]$ un idéal et soit un polynôme $f \in \mathbb{F}[x_1, \ldots, x_n]$ fixé. Soit y une nouvelle indéterminée et soit*

$$\tilde{I} = \langle f_1, \ldots, f_s, 1 - fy \rangle \subset \mathbb{F}[x_1, \ldots, x_n, y],$$

alors $I : f^\infty = \tilde{I} \cap \mathbb{F}[x_1, \ldots, x_n]$.

Lemme 1. *Soit $I \in \mathbb{F}[x_1, \ldots, x_n]$ un idéal et $f \in \mathbb{F}[x_1, \ldots, x_n]$ fixé, tels que* $\dim(I : f^\infty) = 0$. *Alors, pour tout $x \in V(I : f^\infty)$, $f(x) \neq 0$. En particulier $(I : f^\infty) + \langle f \rangle = \langle 1 \rangle$.*

Démonstration. Soit $\tilde{I} = I + \langle 1 - fy \rangle$, tel que $I : f^\infty = \tilde{I} \cap \mathbb{F}[x_1, \ldots, x_n]$. Alors

$$\overline{\Pr(V(\tilde{I}))} = V(I : f^\infty), \tag{8.42}$$

où Pr est la projection $(y, x_1, \ldots, x_n) \mapsto (x_1, \ldots, x_n)$. Comme $V(I : f^\infty)$ est fini, on a que $\Pr(V(\tilde{I}))$ est fini aussi. Donc $\Pr(V(\tilde{I}))$ est donc égal à sa fermeture. On a donc

$$V(I : f^\infty) = \Pr(V(\tilde{I})). \tag{8.43}$$

Comme tout élément $(y, x_1, \ldots, x_n)$ de $\tilde{I}$ vérifie $f(x_1, \ldots, x_n) \neq 0$, cette propriété est conservée par projection. □

Proposition 6. *Soit $n, v > 0$, et soit $\underline{Z}$ les indéterminées $Z_1, \ldots, Z_v$. Considérons le polynôme*

$$\Delta(Z_1, \ldots, Z_v) = Z_1 \cdots Z_v \prod_{1 \leq i < j \leq v} (Z_i - Z_j)\,. \tag{8.44}$$

Alors l'idéal saturé $I_{S,n,v} : \Delta^\infty$ contient les polynômes

$$Z_i^n - 1, \quad i \in \{1, \ldots, v\},$$

et les polynômes

$$S_i^{q^m} - S_i, \quad i \in \{1, \ldots, n\}$$

où $\mathbb{F}_{q^m}$ est le corps de décomposition de $X^n - 1$ sur $\mathbb{F}_q$. En particulier, c'est un idéal radical de dimension 0.

Démonstration. D'après les relations $S_{i+n} - S_i$, $i \in \{1, \ldots, v\}$, nous avons

$$\begin{aligned} 0 &= S_{n+i} - S_i \bmod I_{S,n,v} && (8.45)\\ &= \sum_{j=1}^{v} Z_j^{n+i} - \sum_{j=1}^{v} Z_j^i \bmod I_{S,n,v} && (8.46)\\ &= \sum_{j=1}^{v} Z_j^i \left(Z_j^n - 1\right) \bmod I_{S,n,v}. && (8.47) \end{aligned}$$

Les équations (8.47) peuvent s'écrire matriciellement comme suit :

$$M \begin{pmatrix} Z_1^n - 1 \\ \vdots \\ Z_v^n - 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \bmod I_{S,n,v}, \tag{8.48}$$

où M la matrice carrée $v \times v$ $(M_{i,j}) = \left(Z_j^i\right)$. Le déterminant de cette matrice est Δ. En considérant l'idéal suivant :

$$\tilde{I}_{S,n,v} = I_{S,n,v} + \langle 1 - y\Delta \rangle,$$

on a que Δ est inversible modulo $\tilde{I}_{S,n,v}$, ce qui entraîne que la matrice M est inversible modulo $\tilde{I}_{S,n,v}$. Nous avons donc

$$\begin{pmatrix} Z_1^n - 1 \\ \vdots \\ Z_w^n - 1 \end{pmatrix} = 0 \bmod \tilde{I}_{S,n,v}. \tag{8.49}$$

Donc, pour $i \in \{1, \ldots, v\}$, $Z_i^n - 1 \in \tilde{I}_{S,n,v} \cap \mathbb{F}[\underline{S}_n, \underline{Z}_v] = I_{S,n,v} : \Delta^\infty$. Si $\mathbb{F}_{q^m}$ est le corps de décomposition de $X^n - 1$, alors $X^n - 1 | X^{q^m} - X$: cela entraîne $Z_i^{q^m} - Z_i \in I_{S,n,v} : \Delta^\infty$, pour $i \in \{1, \ldots, v\}$, qui entraîne à son tour $S_j^{q^m} - S_j \in I_{S,n,v} : \Delta^\infty$, pour $j \in \{1, \ldots, n+v\}$, par linéarité de $x \mapsto x^{q^m}$ sur $\mathbb{F}_q$. □

Corollaire 4. *On a* $\dim I_{S,n,v} : \Delta^\infty = 0$. *Pour tout* $(\underline{Z}^*, \underline{S}^*) \in V(I_{S,n,v} : \Delta^\infty)$, $\Delta(\underline{Z}^*) \neq 0$. *En particulier* $(I_{S,n,v} : \Delta^\infty) + \langle \Delta \rangle = \langle 1 \rangle$.

Ce corollaire découle directement du lemme suivant.

Lemme 2. *Soit* $I \in \mathbb{F}[x_1, \ldots, x_n]$ *un idéal et* $f \in \mathbb{F}[x_1, \ldots, x_n]$ *fixé, tels que* $\dim(I : f^\infty) = 0$. *Alors, pour tout* $x \in V(I : f^\infty)$, $f(x) \neq 0$. *En particulier* $(I : f^\infty) + \langle f \rangle = \langle 1 \rangle$.

Démonstration. Soit $\tilde{I} = I + \langle 1 - fy \rangle$, tel que $I : f^\infty = \tilde{I} \cap \mathbb{F}[x_1, \ldots, x_n]$. Alors

$$\overline{\Pr(V(\tilde{I}))} = V(I : f^\infty), \tag{8.50}$$

où Pr est la projection $(y, x_1, \ldots, x_n) \mapsto (x_1, \ldots, x_n)$. Comme $V(I : f^\infty)$ est fini, on a que $\Pr(V(\tilde{I}))$ est fini aussi. Donc $\Pr(V(\tilde{I}))$ est donc égal à sa fermeture. On a donc

$$V(I : f^\infty) = \Pr(V(\tilde{I})). \tag{8.51}$$

Comme tout élément $(y, x_1, \ldots, x_n)$ de $\tilde{I}$ vérifie $f(x_1, \ldots, x_n) \neq 0$, cette propriété est conservée par projection. □

Corollaire 5. *L'idéal*

$$\langle I_{\sigma,v} + (I_{S,n,v} : \Delta^\infty) \rangle \cap \mathbb{F}_q[\underline{S}, \underline{\sigma}_v] \tag{8.52}$$

contient les équations de corps

$$S_i^{q^m} - S_i, \ i \in \{1, \ldots, n+v\}, \quad \textit{et} \ \ \sigma_i^{q^m} - \sigma_i, \ i \in \{1, \ldots, v\}. \tag{8.53}$$

C'est donc un idéal radical de dimension zéro.

Nous noterons $I^\infty_{N,n,v}$ l'idéal $\langle I_{\sigma,v} + (I_{S,n,v} : \Delta^\infty) \rangle \cap \mathbb{F}_q[\underline{S}, \underline{\sigma}_v]$. On a $I_{N,n,v} \subset I^\infty_{N,n,v}$.

Proposition 7. *Dans le cas de la caractéristique 2, soit $T_{n,v}$ l'ensemble des $(\underline{S}^*, \underline{\sigma}_v^*)$ où $\underline{S}^*$ est la transformée de Fourier d'un mot binaire e de poids exactement v, et $\underline{\sigma}_v^*$ est l'ensemble des fonctions symétriques élémentaires de e. Alors*

$$V(I_N^\infty) = T_v \text{ et } I_N^\infty = I(T_v). \tag{8.54}$$

Démonstration. Nous avons $V(I_N^\infty) \subset V(I_N)$, donc $V(I_N^\infty)$ ne contient que des mots de poids inférieur à v, par le théorème 5. Or tout mot de poids exactement v conduit à une solution $(\underline{\sigma}_v^*, \underline{S}_n^*)$ de I_N^∞. Donc $T_{n,v} \subset V(I_{N,n,v})$. Soit maintenant $(\underline{S}^*, \underline{\rho}_v^*) \in V(I_{N,n,v}^\infty)$, alors $\underline{S}_n^*$ est la transformée de Fourier d'un mot e de poids strictement inférieur à v. Soit $\sigma^*(Z)$ le polynôme localisateur de e. Alors le théorème 5 nous dit que $\rho^*(Z) = 1 + \sum_{i=1}^{v} \rho_i^* Z^i$ est de la forme

$$\rho^*(Z) = \sigma^*(Z) G(Z)^2 Z^l. \tag{8.55}$$

avec : ou bien $l \geq 1$ ou bien $\deg G(Z) \geq 1$, car $\deg \sigma^*(Z) \leq v - 1$. Alors, par le corollaire 1, $(\underline{S}_n^*, \underline{\rho}^*)$ s'étend en

$$(\underline{Z}_v^*, \underline{\rho}_v^*, \underline{S}^*) \in V((I_S + I_\sigma) : \Delta^\infty), \tag{8.56}$$

où $\underline{Z}^* = (Z_1^*, \ldots, Z_v^*)$ est tel que, par définition de l'idéal $I_{\sigma,v}$:

$$\rho^*(Z) = \prod_{i=1}^{v} (1 - Z_i^* Z). \tag{8.57}$$

La forme de $\rho^*(Z)$ donnée par (8.55) entraîne que soit l'un des Z_j^* est nul, soit deux d'entre eux sont égaux. Dans les deux cas, $\Delta(Z_1^*, \ldots, Z_v^*) = 0$, ce qui est contradiction avec le corollaire 4. Donc $V(I_N^\infty) = T_v$, et on en déduit $I_N^\infty = I(T_v)$, car I_N^∞ est radical. □

La proposition précédente permet de ne pas considérer les solutions de poids strictement inférieur à v dans la variété associée à I_N. Ces solutions sont appelées *spurious solutions* dans [Sal02], qui ne traite toutefois que le cas du calcul de la distance minimale.

Corollaire 6. *Soit $\underline{S}^*$ la transformée de Fourier d'un mot e. Alors, si e est de poids exactement v,*

$$I_N^\infty(\underline{S} \mapsto \underline{S}^*) = \langle \sigma_v - \sigma_v^*, \ldots, \sigma_1 - \sigma_1^* \rangle, \tag{8.58}$$

où $\sigma_1^*, \ldots \sigma_v^*$ sont les fonctions symétriques élémentaires des localisateurs de e. Si e est de poids différent de v, alors

$$I_N^\infty(\underline{S} \mapsto \underline{S}^*) = \langle 1 \rangle. \tag{8.59}$$

Démonstration. Soit e un mot de poids exactement v de transformée de Fourier $\underline{S}_n^*$. Le théorème 5 entraîne que les solutions $\rho_1^*, \ldots, \rho_v^*$ de $I_N(\underline{S} \mapsto \underline{S}^*)$ sont telles que le polynôme $\rho^*(Z) = 1 + \sum \rho_i^* Z^i$ est multiple du polynôme localisateur $\sigma^*(Z) = 1 + \sum_{i=1}^{w} \sigma_i^* Z^i$, où w est le poids de e. Comme e est de poids exactement v, $\rho^*(Z) = \sigma^*(Z)$, et donc

$$V(I_N(\underline{S} \mapsto \underline{S}^*)) = \{(\sigma_v^*, \ldots, \sigma_1^*)\}. \tag{8.60}$$

D'autre part on a aussi $V(I_N^\infty(\underline{S} \mapsto \underline{S}^*)) \ni (\sigma_v^*, \ldots, \sigma_1^*)$. Donc

$$V(I_N^\infty(\underline{S} \mapsto \underline{S}^*)) = V(I_N(\underline{S} \mapsto \underline{S}^*)) = \{(\sigma_v^*, \ldots, \sigma_1^*)\}. \tag{8.61}$$

Comme les deux idéaux à gauche et à droite de (8.61) sont radicaux, on a bien

$$I_N^\infty(\underline{S} \mapsto \underline{S}^*) = \langle \sigma_v - \sigma_v^*, \ldots, \sigma_1 - \sigma_1^* \rangle. \tag{8.62}$$

Si e est de poids différent de v, nous montrons que $V(I_N^\infty(\underline{S} \mapsto \underline{S}^*) = \emptyset$, le résultat en découlant par le Nullstellenstaz faible. Supposons d'abord que le poids de e est supérieur strictement à v. Alors s'il existe $\underline{\rho}^*$ tel que $\underline{\rho}^* \in V(I_N^\infty(\underline{S} \mapsto \underline{S}^*))$, alors les poids de e est inférieur ou égal à v, par le théorème 5, ce qui est une contradiction.

Supposons pour finir que le poids de e est strictement inférieur à v. Soit $\underline{\rho}^* \in V(I_N^\infty(\underline{S} \mapsto \underline{S}^*))$, alors le théorème 5 entraîne que

$$\rho^*(Z) = \sigma^*(Z)G(Z)^2 Z^l, \tag{8.63}$$

où $\rho^*(Z) = 1 + \sum \rho_i^* Z^i$, et où $\sigma^*(Z)$ est le polynôme localisateur de e. Le corollaire 1 nous permet d'étendre $(\underline{\rho}^*, \underline{S}^*)$ en $(\underline{Z}^*, \underline{\rho}^*, \underline{S}^*)$, tel que

$$(\underline{Z}^*, \underline{\rho}^*, \underline{S}^*) \in V(I_\sigma + (I_S : \Delta^\infty)). \tag{8.64}$$

Mais le forme de $\rho^*(Z)$ entraîne que soit l'un des Z_i^* est nul, soit deux d'entre eux sont égaux. Dans les deux cas, $\Delta(Z^*) = 0$, ce qui est impossible d'après le corollaire 4. □

Théorème 7. *Pour chaque $j \in \{1, \ldots, v\}$, pour chaque $\underline{S}^*$, qui est la transformée de Fourier d'un mot de poids exactement v, I_N^∞ contient un polynôme de la forme $q_j(\underline{S})\sigma_j + r_j(\underline{S})$, tel que $q_j(\underline{S}^*) \neq 0$.*

Démonstration. Soit $j \in \{1, \ldots, v\}$, et considérons l'idéal d'élimination

$$I_j = I_N^\infty \cap \mathbb{F}_2[\sigma_j, \underline{S}]. \tag{8.65}$$

Soit $\underline{S}^*$ la transformée de Fourier d'un mot de poids exactement v. L'idéal I_j est un idéal de dimension zéro. Le théorème 3, de spécialisation dans le cas zéro dimensionnel, s'applique. Il existe donc $g_j \in I_j$ tel que

$$g_j(\underline{S} \mapsto \underline{S}^*) \tag{8.66}$$

engendre $I_j(\underline{S} \mapsto \underline{S}^*)$, et tel que $\deg_{\sigma_j} g_j(\underline{S} \mapsto \underline{S}^*) = \deg_{\sigma_j} g_j$. Nous savons, par le théorème 6 que

$$I_N^\infty(\underline{S} \mapsto \underline{S}^*) = \langle \sigma_v - \sigma_v^*, \ldots, \sigma_1 - \sigma_1^* \rangle . \tag{8.67}$$

Donc

$$I_j(\underline{S} \mapsto \underline{S}^*) = \langle \sigma_j - \sigma_j^* \rangle . \tag{8.68}$$

On en déduit que g_j est de degré 1 en σ_j. Donc $g_j = q_j\underline{S})\sigma_j + r_j(\underline{S})$, avec de plus $q_j(\underline{S}^*) \neq 0$. □

Théorème 8. *Pour chaque $i \in \{1, \ldots, v\}$, l'idéal $I_{N,v}^\infty$ contient un polynôme $\sigma_i + r_i(\underline{S})$.*

Démonstration. Soit $i \in \{1, \ldots, v\}$. Pour chaque transformée de Fourier $\underline{S}^*$ d'un mot de poids v, il existe dans $I_{N,v}^\infty$ un polynôme $g = q(\underline{S})\sigma_i + r(\underline{S})$, tel que $q(\underline{S}) \neq 0$. Soit $g_1, \ldots, g_l$ tous ces polynômes, de la forme $q_1(\underline{S})\sigma_i + r_1(\underline{S}), \ldots, q_l(\underline{S})\sigma_i + r_l(\underline{S})$. Soit $I = I_{N,v}^\infty \cap \mathbb{F}_q[\underline{S}]$. Alors $V(I)$ est l'ensemble des transformées de Fourier des mots de poids exactement v, et pour chaque $\underline{S}^* \in V(I)$, il existe un $q_j(\underline{S})$ tel que $q_j(\underline{S}^*) \neq 0$. On a donc

$$V(\langle q_1, \ldots, q_l \rangle + I) = \emptyset, \tag{8.69}$$

donc, d'après le Nullstellenstaz faible

$$\langle q_1, \ldots, q_l \rangle + I = \mathbb{F}_q[\underline{S}]. \tag{8.70}$$

Il existe donc $t_1, \ldots, t_l \in \mathbb{F}_q[\underline{S}]$, et $t \in I$, tels que

$$t_1 q_1 + \ldots t_l q_l + t = 1. \tag{8.71}$$

En faisant la combinaison linéaire $t_1 g_1 + \dots t_l g_l$, on a

$$\begin{aligned} t_1 g_1 + \dots t_l g_l &= (t_1 q_1 + \dots t_l q_l)\sigma_i + (t_1 r_1 + \dots t_l r_l) \\ &= (1-t)\sigma_i + (t_1 r_1 + \dots t_l r_l) \\ &= \sigma_i + r(\underline{S}) \bmod I \end{aligned} \tag{8.72}$$

Comme $g_1, \dots, g_l \in I^{\infty}_{N,n}$, et $I \subset I^{\infty}_{N,v}$, on a $\sigma_i + r(\underline{S}) \in I^{\infty}_{N,v}$. □

Ainsi, nous avons des formules donnant, pour les mots de poids v donné, les fonctions symétriques élémentaires en fonctions de tous les coefficients de la transformée. Mais dans le contexte du décodage, on est intéressé par des formules pour les σ_i en fonction seulement des syndromes S_Q connus.

8.8 Utilisation des équations de Newton en décodage

Après avoir étudié les propriétés du système des équations de Newton en considérant le vecteur entier $\underline{S}^*$ de la transformée de Fourier, nous étudions le système des équations de Newton lorsque l'on connait qu'une partie de la transformée de Fourier. Dans le contexte du décodage, le code C est donné son ensemble de définition Q, de complémentaire N, et un mot c est dans le code C si et seulement si

$$c(\alpha^i) = 0, \quad i \in Q. \tag{8.73}$$

Le motif d'erreur e est donc connu par les $e(\alpha^i)$, $i \in Q$. Le problème naturel qui se pose alors est le suivant : peut-on éliminer, dans les équations de Newton, les syndromes inconnus S_j, $j \in N$, et obtenir des formules de degré un en les σ_i, où les σ_i sont les fonctions symétriques élémentaires des localisateurs. Il y a deux manières de traiter ce problème :

1. soit le décodage formel : on élimine par un précalcul les syndromes inconnus dans le système I_N (ou I^{∞}_N), pour obtenir des formules de de la forme $q_j(S_Q)\sigma_j + r_j(S_Q)$. Au moment du décodage, on substitue les syndromes S^*_Q pour obtenir les valeurs des σ_i.
2. soit le décodage en ligne : au moment du décodage, on effectue la substitutions $\underline{S} \mapsto \underline{S}^*$ dans le système I_N (ou I^{∞}_N), et on élimine les syndromes inconnus S_N, pour obtenir les valeurs des σ_i, par des relations de degré 1.

Dans les deux cas, il faut prouver qu'on obtient des formules de degré 1 en les coefficients σ_i du polynôme localisateur.

Nous étudions d'abord le résultat pour le point 2, où l'on substitue d'abord, pour éliminer ensuite. Ensuite nous étudions le point 1. Des difficultés techniques se présentent : nous aimerions pouvoir utiliser l'idéal des relations de Newton $I_{N,n,v}$ plutôt que $I^{\infty}_{N,n,v}$, car il peut être coûteux de calculer une base de Gröbner de ce dernier. Mais nous avons seulement prouvé la radicalité de $I^{\infty}_{N,n,v}$, bien que nous conjecturons que $I_{N,n,v}$ soit aussi radical, donc nous établirons le résultat pour le décodage formel en utilisant $I^{\infty}_{N,n,v}$, alors que pour le décodage en ligne nous pouvons utiliser $I_{N,n,v}$.

8.9 Décodage en ligne

Théorème 9. *Soit S^*_Q l'ensemble des syndromes d'une erreur e de poids w, telle que e est le seul mot de poids inférieur à v admettant S^*_Q comme syndromes (c'est le cas si $v \leq t$). Soit I l'idéal*

$$I_{N,v}(S_Q \mapsto S^*_Q) + \langle \sigma_{w+1}, \dots, \sigma_v \rangle, \tag{8.74}$$

alors,

$$I = \left\langle \sigma_1 - \sigma_1^*, \ldots, \sigma_w - \sigma_w^*, \sigma_{w+1}, \ldots, \sigma_v, S_j - S_j^*, j \in N \right\rangle \tag{8.75}$$

Démonstration. La preuve est en deux étapes : on montre d'abord l'égalité des variétés associées ; puis que l'idéal du membre de gauche de 8.75 est radical.

Soit donc $(\underline{\rho}^*, \underline{S}_N^*)$ solution de $I(S_Q \mapsto S_Q^*, \sigma_{w+1}, \ldots, \sigma_v)$. Alors par le théorème 5, le poids de la transformée de Fourier inverse e' de (S_N^*, S_Q^*) est inférieur à v. Étant donnée l'hypothèse que e est le seul mot de poids inférieur à v admettant S_Q^* comme syndromes, on a $e = e'$, et $S_N^* = S_N(e)$. D'autre part, le théorème 5, entraîne que $\rho^*(Z)$ est multiple du polynôme localisateur $\sigma^*(Z)$ de e. Comme les termes $\rho_{w+1}^*, \ldots, \rho_v^*$ sont nuls, on a $\rho^*(Z) = \sigma^*(Z)$. Les variétés associées sont bien égales.

Pour la radicalité, on construit l'idéal

$$J = I_{S,w}(\underline{S}_Q \mapsto \underline{S}_Q^*) + I_{\sigma,w} + \langle \sigma_{w+1}, \ldots, \sigma_v \rangle \in \mathbb{F}_1[\underline{Z}_w, \underline{\sigma}_v \underline{S}_N], \tag{8.76}$$

qui est tel que

$$J \cap \mathbb{F}_2[\underline{\sigma}_v, \underline{S}] = I_{N,v}(\underline{S}_Q \mapsto \underline{S}_Q^*) + \langle \sigma_{w+1}, \ldots, \sigma_v \rangle . \tag{8.77}$$

Alors J est de dimension zéro, et les seules racines $\underline{Z}_w^* = (Z_1^*, \ldots, Z_w^*)$ telles que $(\underline{Z}_w^*, \underline{\sigma}_w^*, \sigma_{w+1}^* = 0, \ldots, \sigma_v^* = 0 S_N^*) \in V(J)$ sont, à permutation prés, les racines du polynôme localisateur $\sigma^*(Z)$. Elles vérifient donc $\Delta_w(Z_w) \neq 0$. En utilisant les équations $S_{i+n} - S_i = 0$, on construit le système

$$M \begin{pmatrix} Z_1^n - 1 \\ \vdots \\ Z_w^n - 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \bmod J, \tag{8.78}$$

où M est la matrice carrée $v \times v$ $(M_{i,j}) = \left(Z_j^i\right)$, de déterminant $\Delta_w(\underline{Z}_w)$. Alors $\Delta_w(Z_w)$ est inversible modulo J, et donc $Z_i^n - 1 \in J$, $i \in \{1, \ldots, w\}$. L'idéal J contient donc les équations de corps, il est radical. □

Théorème 10. *Soit S_Q^* les syndromes d'une erreur e de poids w, telle que e est le seul mot de poids inférieur à v admettant S_Q^* comme syndromes (c'est le cas si $v \leq t$). Alors, si $w = v$*

$$I_N^\infty(S_Q \mapsto S_Q^*) = \left\langle \sigma_1 - \sigma_1^*, \ldots, \sigma_v - \sigma_v^*, S_j - S_j^*, j \in N \right\rangle \tag{8.79}$$

et si $w \neq v$ alors

$$I_N^\infty(S_Q \mapsto S_Q^*) = \langle 1 \rangle . \tag{8.80}$$

Démonstration. Supposons que $w = v$, alors, soit $(\underline{\sigma}_v^*, \underline{S}_n^*) \in V(I_N^\infty(S_Q \mapsto S_Q^*))$. Alors le poids de la transformée inverse e' de (S_Q^*, S_N^*) est égal à v (proposition 7). Comme e est l'unique mot de poids v admettant S_Q^* comme syndromes, on a $e = e'$. Cela prouve l'égalité des variétés. L'égalité des idéaux s'ensuit car $I_{N,v}^\infty$ est radical. □

Dans le décodage en ligne (calcul avec les syndromes spécialisés), on peut utiliser l'idéal $I_{N,v}(S_Q \mapsto S_Q^*)$, en annulant progressivement $\sigma_1, \ldots, \sigma_v$, puis $\sigma_2, \ldots, \sigma_v$, jusqu'à obtenir une base de Gröbner différente de 1, auquel cas on a dans la base de Gröbner les polynômes $\sigma_j - \sigma_j^*$, ce qui donne les coefficients du polynôme localisateur.

Une autre solution est d'utiliser l'idéal $I_N^\infty(S_Q \mapsto S_Q^*)$, en augmentant le poids jusqu'à l'obtention d'une base de Gröbner différente de $\langle 1 \rangle$. Alors on aura bien obtenu, dans la base de Gröbner, les polynômes $\sigma_i - \sigma_i^*$, donnant les coefficients du polynôme localisateur.

8.10 Décodage formel

Nous nous intéressons à l'idéal $I_N \cap \mathbb{F}_2[\underline{\sigma}_v, S_Q]$, c'est-à-dire à l'idéal obtenu en éliminant les syndromes inconnus. L'objectif est de trouver des formules de degré un pour les σ_i. Nous avons le théorème suivant, similaire au théorème 7, mais avec la propriété que seuls les syndromes S_Q apparaissent dans les formules pour les σ_i.

Proposition 8. *Soit C un code cyclique et v inférieur à la capacité de correction t de C. Alors, pour tout $j \in \{1, \ldots, v\}$, et pour chaque S_Q^*, qui est la transformée de Fourier d'un mot de poids exactement v, il existe $p_j = q_j(\underline{S})\sigma_j + r_j(\underline{S}_Q) \in I_{N,v}^{\infty} \cap \mathbb{F}_2[\underline{\sigma}_v, S_Q]$, tel que $q_j(S_Q^*) \neq 0$,*

Démonstration. On a d'abord que

$$V(I_{N,v}^{\infty} \cap \mathbb{F}_2[S_Q]) \tag{8.81}$$

est l'ensemble des syndromes des mots de poids exactement v (théorème de fermeture dans le cas d'un idéal de dimension zéro). Notons $T_{Q,v}$ cet ensemble. Alors $I(T_{Q,v}) = I_{N,v}^{\infty} \cap \mathbb{F}_2[S_Q])$, car l'idéal $I_{N,v}^{\infty} \cap \mathbb{F}_2[S_Q]$ est radical. Soit l'idéal d'élimination

$$I_j^{\infty} = I_{N,v}^{\infty} \cap \mathbb{F}_2[\sigma_j, S_Q], \tag{8.82}$$

soit G_j une base de Gröbner réduite pour un ordre éliminant σ_j, et finalement g_j de degré minimal en σ_j que l'on écrit

$$g_j = p_j(S_q)\sigma_j^l + \ldots \tag{8.83}$$

Supposons $l > 1$. Soit S_Q^* un syndrome de poids exactement v, alors, d'après le théorème 10

$$I_j^{\infty}(S_Q \mapsto S_Q^*)) = \langle \sigma_j - \sigma_j^* \rangle . \tag{8.84}$$

Le théorème 2 entraîne donc $p_j(S_Q^*) = 0$ (chute de degré). Cela est vrai pour tout syndrome $S \in T_{Q,v}$ de mot de poids v, on a donc $p_j \in I(T_{Q,v}) = I_j^{\infty} \cap \mathbb{F}_2[S_Q]$, ce qui est une contradiction car G_j est une base de Gröbner réduite. Donc

$$g_j = q_j(\underline{S})\sigma_j + r_j(\underline{S}). \tag{8.85}$$

On a de plus, $q_j(S_Q^*) \neq 0$, car $g_j(S_Q \mapsto S_Q^*)$ est une base de Gröbner de $I_j(S_Q \mapsto S_Q^*)$, laquelle est égale à $\sigma_j - \sigma_j^*$, où σ_j^* est le j-ième coefficient du polynôme localisateur de e. □

De même que précédemment, nous avons en réalité le résultat plus fort suivant :

Corollaire 7. *Si v est plus petit que la capacité de correction du code C, d'ensemble de définition Q, alors, pour chaque $i \in \{1, \ldots, v\}$, l'idéal $I_{N,v}^{\infty} \cap \mathbb{F}_q[\underline{\sigma}_v, S_Q]$ contient un polynôme $\sigma_i + r_i(S_Q)$.*

Démonstration. Soit $i \in \{1, \ldots, v\}$, pour chaque transformée de Fourier $\underline{S}^*$ d'un mot de poids v, il existe $g = q(S_q)\sigma_i + r(S_Q) \in I_{N,v}^{\infty} \cap \mathbb{F}_2[\underline{\sigma}_v, S_Q]$ tel que $q(S_Q) \neq 0$. Soit, pour tous les mots de poids v, $g_1, \ldots, g_l$, tous ces polynômes, de la forme $q_1(S_Q)\sigma_i + r_1(S_Q), \ldots q_l(S_Q)\sigma_i + r(S_Q)$. Soit $I = I_{N,v}^{\infty} \cap \mathbb{F}_1[S_Q]$. Alors $V(I)$ est l'ensemble des syndromes de mots de poids exactement v, et pour chaque $S_Q^* \in V(I)$, il existe un $q_j(S_Q)$ tel que $q_j(S_Q^*) \neq 0$. On a donc

$$V(\langle q_1, \ldots, q_l \rangle + I) = \emptyset, \tag{8.86}$$

et donc

$$\langle q_1, \ldots, q_l \rangle + I = \mathbb{F}_q[S_Q] \tag{8.87}$$

Il existe donc $t_1, \ldots, t_l \in \mathbb{F}_q[S_Q]$, et $t \in I$, tels que

$$t_1 q_1 + \cdots + t_l q_l + t = 1. \tag{8.88}$$

En faisant la combinaison linéaire $t_1 g_1 + \cdots + t_l g_l$, on a

$$\begin{aligned} t_1 g_1 + \ldots t_l g_l &= (t_1 q_1 + \ldots t_l q_l)\sigma_i + (t_1 r_1 + \ldots t_l r_l) \\ &= (1-t)\sigma_i + (t_1 r_1 + \ldots t_l r_l) \\ &= \sigma_i + r(S_Q) \bmod I \end{aligned} \tag{8.89}$$

Comme $g_1, \ldots, g_l \in I_{n,v}^{\infty} \cap \mathbb{F}_q[\underline{\sigma_v}, S_Q]$, et que $I \subset \mathbb{F}_q[\underline{\sigma_v}, S_Q]$, on a $\sigma_i + r(S_Q) \in \mathbb{F}_q[\underline{\sigma_v}, S_Q]$. □

8.11 En pratique

Magali Bardet a montré dans sa thèse que le décodage formel, c'est-à-dire l'obtention de formules dans lesquelles on substitue les syndromes, devient vite impraticable. La figure 3.1 donne un exemple, en petite longueur i.e. 41, où on voit déjà que les formules deviennent vite trop importantes pour être utilisées.

Magali Bardet a proposé de faire un calcul de base de Gröbner pour chaque mot à décoder, en spécialisant les syndromes (décodage en ligne). Ces calculs peuvent être grandement accélérés, en utilisant la technique de la *trace du pré-calcul*. Cette technique consiste à tirer, dans une étape de précalcul, un syndrome S_Q^* au hasard, et à calculer la base de Gröbner de $I_{N,v}(S_Q \mapsto S_Q^*)$. Pendant cette étape, on garde en mémoire tous les calculs fructueux : en effet un algorithme de base de Gröbner effectue essentiellement des réductions de polynômes par rapport à d'autres polynômes, et la plupart de ces réductions sont nulles, donc ne contribuent pas au résultat final (algorithmes de Buchberger, algorithme F_4). Une fois qu'on a gardé toutes les opérations qui ne donnent pas de réduction à zéro, on réapplique l'algorithme de base de Gröbner, en n'effectuant que les calculs mémorisés.

Cette méthode est conjecturelle : elle repose que sur l'hypothèse que tous les syndromes (au moins ceux de même poids) vont se comporter génériquement de la même manière, par rapport aux étapes du calcul de la base de Gröbner. En pratique, cela a bien été vérifié pour tous les décodages effectués.

Le nombre d'opérations arithmétique sur le corps de décomposition de $X^n - 1$, où n est la longueur du code, est assez faible, voir la figure 3.2, page 25.

Chapitre 9

Décodage par interpolation

Ce chapitre est consacré à mes travaux sur les algorithmes de Sudan, Shokrollahi-Wasserman et Guruswami-Sudan, pour le décodage des codes d'évaluation, tels que les codes de Reed-Solomon et les codes géométriques. Ces algorithmes reposent sur le notion de *décodage en liste* : on augmente le rayon de décodage, mais la conséquence est que l'on perd l'unicité du mot de code le plus proche, obtenue lorsqu'on décode jusqu'à $\lfloor \frac{d-1}{2} \rfloor$ erreurs. Ces algorithmes reposent tous sur le même schéma général suivant : soit (x_i, y_i), $i \in \{1, \ldots, n\}$, la donnée du problème, et on cherche $f(X)$ telle que $f(x_i) = y_i$ pour le grand nombre d'indices possible. On construit un polynôme $Q(X, Y)$, tel que $Q(x_i, y_i) = 0$, avec des conditions sur le degré pondéré de $Q(X, Y)$ telles que si $f(X)$ est solution du problème, alors $Q(X, f(X)) = 0$. Il ne reste plus qu'à factoriser $Q(X, Y)$ pour récupérer les facteurs $Y - f(X)$. Les problèmes liés au calcul du polynôme $Q(X, Y)$, ainsi qu'à sa factorisation ont été mis de cotés par Sudan et Guruswami, et des nombreuses propositions ont été faites pour résoudre ces problèmes efficacement. Nous avons contribué, avec Lancelot Pecquet, à ce domaine, en proposant une méthode rapide de recherche de racines $f(X)$ d'un polynôme $Q(X, Y)$, dans le cas simple de l'algorithme de Sudan.

La deuxième contribution (non publiée) est l'étude de généralisations multivariées de l'algorithme de Guruswami-Sudan, lorsqu'on traite des codes de Reed-Muller et des codes produits de codes Reed-Solomon. Pellikaan et Wu ont traité le problème [PW04b] du décodage des codes de Reed et Muller, avec une analyse reposant sur la théorie des bases de Gröbner. Nous proposons une analyse plus simple, reposant sur un lemme de Schwarz-Zippel généralisé, qui donne un meilleur rayon de décodage.

9.1 Introduction aux algorithmes de décodage par interpolation

9.1.1 Le problème

On considère le code de Reed-Solomon, défini comme code d'évaluation, comme suit : soit $X = \{x_1, \ldots, x_n\}$, un ensemble de n points distincts dans le corps fini $\mathbb{F}_q$, soit

$$L_k = \{f \in \mathbb{F}_q[X]; \quad \deg f < k\}. \tag{9.1}$$

Le code de Reed-Solomon C_k, de paramètres $[n, k, n-k+1]$ est l'ensemble

$$C_k = \{(f(x_1), \ldots, f(x_n)), \quad f \in L_k\}. \tag{9.2}$$

Le problème du décodage en liste des codes de Reed-Solomon est le suivant :

Problème DÉCODAGE EN LISTE DES CODES DE REED-SOLOMON

Instance – q une puissance d'un nombre premier
– $n, k \in \mathbb{N}$, $1 \leq k < n$, t, $1 \leq t < n$
– $X = \{x_1, \ldots, x_n\} \subset \mathbb{F}_q$, les x_i étant distincts : c'est le support du code de Reed-Solomon C_k.
– $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$.

Réponse La liste de tous les mots de code à distance au plus t de y.

Définition 3. *Soit $f \in \mathbb{F}_q[X]$, $x_1, \ldots, x_n$ n éléments distincts de $\mathbb{F}_q$, et $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$. La* distance de Hamming *de f à y, notée $d(f, y)$ est*

$$|\{i \in \{1, \ldots, n\}; f(x_i) \neq y_i\}| \tag{9.3}$$

Avec cette notation, on peut écrire le problème

Réponse La liste de tous les polynômes $f \in \mathbb{F}_q[X]$, de degré inférieur à k tels que $d(f, y) \leq t$.

Notons qu'une manière équivalente de définir la réponse est la suivante :

Réponse la liste de tous les polynômes de degré inférieur à k tels que $f(x_i) = y_i$ pour au moins $\mu = n - t$ valeurs de l'indice i.

Ainsi on peut voir ce problème comme un problème d'approximation pour la distance de Hamming. L'interpolation de Lagrange nous permet d'interpoler sur toutes les valeurs, mais nous perdons la contrainte sur le degré : si on veut des solutions avec un degré inférieur à k, il faut admettre de n'interpoler que sur un nombre moindre de points. On peut parler d'*interpolation partielle*.

9.1.2 La difficulté du problème

Goldreich, Rubinfeld et Sudan ont montré [GRS00] que le problème, où les x_i ne sont pas distincts, apparemment proche est NP-dur. Précisément, le problème est le suivant.

Problème RECONSTRUCTION DE POLYNÔMES.

Instance – q une puissance d'un nombre premier
– $k, t, \mu \in \mathbb{N}$,
– n points $(x_i, y_i) \in \mathbb{F}_q^2$, $i \in \{1, \ldots, n\}$. Il n'est pas requis que les x_i soient distincts.

Question Existe-t'il un polynôme de degré au plus k tel que $f(x_i) = y_i$ pour au moins μ valeurs de i.

En réalité il est explicitement utilisé dans la preuve [GRS00] que les x_i ne sont pas distincts : une instance quelconque du problème du problème SUBSET SUM [GJ79] est réduite à une instance de RECONSTRUCTION DE POLYNÔMES, où il y a des couples (x_i, y_i), (x_j, y_j), avec des égalités $x_i = x_j$. Mais en ce qui concerne le vrai problème de décodage, c'est-à-dire quand les x_i sont distincts, Guruswami et Vardy [GV05] ont montré que le problème du décodage des codes de Reed-Solomon est aussi NP-dur. Hélas les codes pour lesquels ils montrent que le problème du décodage est difficile sont des codes sur un alphabet $\mathbb{F}_q$ avec q exponentiel en la longueur, ce qui limite nettement la portée du résultat en pratique.

9.1.3 L'algorithme de Berlekamp-Welch

Toutefois dans le cas où $n - \mu = t < \frac{n-k}{2}$, le problème est résoluble en temps polynomial, grâce notamment à l'algorithme de Berlekamp-Welch [WB86]. Nous le rappelons brièvement, car il est le premier algorithme de décodage par interpolation, et permet de bien introduire l'algorithme de Guruswami-Sudan.

Algorithme de Berlekamp-Welch

Constantes q une puissance d'un nombre premier ;

Données $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$, $k \in \mathbb{N}$, le support du code de Reed-Solomon ; $y = (y_1, \ldots, y_n)$ le mot reçu.

Résultat L'unique polynôme f de degré inférieur à k tel que $d(f, y) \leq t = \frac{n-k}{2}$.

Étape d'interpolation trouver $U(X)$ et $V(X)$ tel que

$$U(x_i)y_i + V(x_i) = 0, \quad i \in \{1, \ldots, n\}, \tag{9.4}$$

et tel que

$$\deg U(X) \leq t \text{ et } \deg V(X) < k + t. \tag{9.5}$$

Étape de recherche de racines retourner $f = \frac{V(X)}{U(X)}$.

On montre que si $Q(X, Y) = U(X)Y + V(X)$ vérifie (9.4) et (9.5), alors la fraction $-\frac{V(X)}{U(X)}$ ne dépend pas de Q. En particulier on retrouvera bien la solution f au problème de décodage, car elle permet de construire les polynômes

$$U(X) = E(X) \text{ et } V(X) = f(X)E(X) \tag{9.6}$$

où $E(X) = \prod(X - x_{i_j})$ est le polynôme localisateur des erreurs, c'est-à-dire tel que $E(x_i) = 0$ quand $f(x_i) \neq y_i$.

9.1.4 L'algorithme de Guruswami-Sudan

La percée fondamentale dans le domaine du décodage des codes de Reed-Solomon est due à Sudan [Sud97], pour être ensuite améliorée par Guruswami et Sudan [GS99]. Nous rappelons brièvement l'algorithme de Guruswami-Sudan, celui de Sudan en étant un cas particulier. L'idée est d'améliorer l'algorithme de Berlekamp-Welch, en introduisant un polynôme $Q(X, Y)$, non plus de degré 1, mais de degré supérieur à 1. Dans cet algorithmes intervient la notion de multiplicité, que définissons comme suit.

Définition 4. *Soit $f = f(X_1, \ldots, X_m) = f_0 + f_1 + \cdots + f_d$, où f_i est homogène de degré i. La* multiplicité *de f en $(0, \ldots, 0)$ est le plus petit indice i tel que $f_i \neq 0$. Soit $p = (p_1, \ldots, p_m) \in \mathbb{F}_q^m$, la multiplicité de f en p, notée* $\mathrm{mult}(f; p)$*, est la multiplicité en 0 de $f(X_1 + p_1, \ldots, X_m + p_m)$.*

Nous avons aussi besoin de la notion de degré pondéré.

Définition 5. *Le degré pondéré par $u_1, \ldots, u_m$ du monôme $aX_1^{i_1} \cdots X_m^{i_m}$ est $u_1 i_1 + \cdots u_m i_m$. Soit $f = f(X_1, \ldots, X_m) \in \mathbb{F}_q[X_1, \ldots, X_m]$, le degré pondéré de f, noté $wdeg_{u_1,\ldots,u_m} f$ est le maximum des degrés pondérés des monômes de f.*

Nous pouvons donner le principe de l'algorithme de Guruswami-Sudan :

Algorithme de Guruswami-Sudan

Constantes q une puissance d'un nombre premier ; $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$ le support du code de Reed-Solomon , $k, \mu \in \mathbb{N}$;

Données $y = (y_1, \ldots, y_n)$ le mot reçu.

Résultat tous les polynômes f de degré inférieur à k tel que $f(x_i) = y_i$ pour au moins μ valeurs de i.

Paramètres auxiliaires d et s à déterminer dans l'analyse de l'algorithme.

Étape d'interpolation trouver un polynôme $Q(X, Y) \in \mathbb{F}_q[X, Y]$ tel que

1. $Q(X, Y) \neq 0$,
2. $\mathrm{wdeg}_{1,k-1} Q(X, Y) \leq d$,

3. $\text{mult}(Q;(x_i,y_i)) = s$, $i \in \{1,\dots,n\}$.

Factorisation Constituer la liste L des $f = f(X)$ tels que $Q(X,f) = 0$.

vérification retourner les $f \in L$ tels que $\deg f < k$, et $d(f,y) < t$.

Pour analyser l'algorithme, on procède en deux étape : premièrement, on constate que $Q_f = Q(X,f)$ est de degré au plus d (grâce à la condition de degré pondéré) ; ensuite si $f(x_i) = y_i$ pour μ valeurs de i, alors le polynôme Q_f a au moins $s\mu$ zéros, comptés avec multiplicité. Donc si

$$s\mu > d, \tag{9.7}$$

$Q(X,f)$ est identiquement nul. Deuxièmement, il faut être assuré qu'un polynôme $Q(X,Y)$ existe quelque soit le mot y à décoder. On remarque que résoudre l'étape d'interpolation revient à résoudre un système d'équations linéaires, dont les inconnues sont les coefficients de $Q(X,Y)$. On écrit donc la condition

$$N_Q > N_{\text{eq}} \tag{9.8}$$

où N_Q est le nombre de termes de Q, et N_{eq} le nombre d'équations linéaires imposées par les conditions $\text{mult}(Q;(x_i,y_i)) = s$, $i \in \{1,\dots,n\}$. Cela donne

$$\frac{d(d+2)}{2(k-1)} > n\binom{s+1}{2}. \tag{9.9}$$

En éliminant d dans (9.9) et dans (9.7), on obtient

Théorème 11. *L'algorithme de Guruswami-Sudan, avec le choix s pour la multiplicité, décode jusqu'à t erreurs, avec*

$$t = n - \mu \leq n\left(1 - \sqrt{\frac{k-1}{n}(1+\frac{1}{s})}\right). \tag{9.10}$$

L'algorithme de Sudan d'origine, revient à faire $s = 1$ (ne pas considérer de multiplicités). Dans ce cas, la capacité de correction est $t = n(1 - \sqrt{2\frac{k-1}{n}})$. À l'opposé, quand s croît, on obtient une capacité de correction de $n(1-\sqrt{R})$, où $R = \frac{k-1}{n}$ est sensiblement le taux de transmission du code. On exprime aussi ce taux de correction sous la forme $n - \sqrt{n(n-d)}$, où d est la distance minimale du code.

Du point de vue de la complexité de cet algorithme, les auteurs citent les papiers principaux sur la factorisation multivariée [Len85, Kal85, Gri86], en affirmant que celle-ci peut se faire en temps polynomial déterministe en q et le degré du polynôme, ou en temps polynomial probabiliste en $\log q$ et le degré du polynôme. Nous verrons dans la suite quelques travaux, en codage, pour traiter efficacement le problème de l'interpolation, et aussi le problème de la factorisation.

9.2 Algorithmes de Sudan et Guruswami-Sudan : interpolation efficace

Rappelons le problème à résoudre pour effectuer la première étape, où on cherche le polynôme $Q(X,Y)$.

Données $\{x_1,\dots,x_m\} \subset \mathbb{F}_q$, $y = (y_1,\dots,y_n) \in \mathbb{F}_q^n$, d un degré, et s un ordre de multiplicité.

Question Trouver un polynôme $Q(X,Y) \in \mathbb{F}_q[X,Y]$ tel que

1. $Q(X,Y) \neq 0$,
2. $\mathrm{wdeg}_{1,k-1} Q(X,Y) \leq d$,
3. $\mathrm{mult}(Q;(x_i,y_i)) = s$, $i \in \{1,\ldots,n\}$.

Le problème est un système à $n\binom{s+1}{2} = O(ns^2)$ équations (pour chaque point le nombre d'équations correspond au nombre de monômes de degré inférieur ou égal à $s-1$). L'approche utilisant l'algorithme de Gauss donne donc $O(n^3 s^6)$.

Nous citerons les principaux travaux dans ce domaine. Il y a deux approches : la première est l'approche par matrices structurées [OS99, OS03a], qui a donné lieu à un brevet [OS03b]. Cette approche consiste à dire que le système d'équations à résoudre est défini par une matrice fortement structurée, que l'on peut stocker de manière compacte. Les auteurs parlent de « structure de déplacement », et obtiennent une complexité de $O(ln^2 s^4)$, où l est la taille de la liste des solutions. Notons que cette approche s'applique aussi aux codes géométriques hermitiens, avec une complexité de $O(lq^7) = O(ln^{7/3})$.

Une autre approche est de construire, par un processus itératif, une famille de polynômes $Q_i(X,Y)$, où $i = 1,\ldots,l$ et l une borne sur la taille de la liste, qui se calcule explicitement. Ces polynômes vérifient les conditions d'interpolation jusqu'à un certain point (x_i,y_i), et on les met à jour pour passer au point suivant $x_{i+1},y_{i+1})$. C'est un algorithme très similaire à l'algorithme de Berlekamp-Massey [Ber68, Mas69]. Cette approche a été suggérée par [NH00]. On peut faire remonter ces techniques à l'algorithme « FIA »(Fundamental Iterative Algorithm) de Feng et Tzeng [FT91], qui est un algorithme qui donne la plus courte combinaison linéaire des premières colonnes d'une matrice, algorithme étudié dans la thèse de Kötter [K̈96], dans le contexte du décodage des codes géométriques.

On constate aussi qu'on peut construire une base de Gröbner de l'idéal des polynômes qui satisfont les contraintes d'interpolation [MTV04], et retenir dans cette base un polynôme $Q(X,Y)$ minimal. O'Keeffe et Fitzpatrick [OF02] étudient le problème de l'interpolation en utilisant les bases de Gröbner, dans un contexte plus large, avec application au décodage et (oralement) ils prétendent avoir une complexité de $O(n^2 s^4)$. Lee et O'Sullivan [LO06], et Aleknovich [Ale02, Ale05] utilisent l'approche par bases de Gröbner, en pouvant écrire directement une base Gröbner de l'idéal pour un ordre donné, que l'on transforme en une base de Gröbner pour l'ordre désiré. Enfin Sakata a proposé une version de son algorithme BMS (Berlekamp-Massey-Sakata) [Sak90], pour trouver un polynôme d'interpolation [SNF00, Sak01]. Il l'applique aussi aux codes géométriques hermitiens, pour obtenir une complexité de $O(s^6 n^{5/2} k^{-1/2})$.

9.3 Algorithme de Sudan : recherche de racines

Rappelons le problème.

Données $X = \{x_1,\ldots,x_n\}$ n points distincts dans $\mathbb{F}_q$, $y = (y_1,\ldots,y_n)$ le mot reçu, $Q(X,Y)$ un polynôme dans $\mathbb{F}_q[X][Y]$, tel que $Q(x_i,y_i) = 0$, avec la multiplicité s.

Question trouver tous les polynômes $f \in \mathbb{F}_q[X]$, $\deg f \leq k$, tels que $Q(X,f) = 0$.

Nous avons contribué dans le cadre de l'algorithme de Sudan simple ($s = 1$) à l'étape de recherche de racines. Notons qu'à l'étape d'interpolation, on peut choisir $Q = Q(X,Y)$ de degré minimal. Soit alors $Q'(X,Y)$ la dérivée de Q par rapport à Y. On la propriété suivante :

Proposition 9. *Soit $X = \{x_1,\ldots,x_n\} \subset \mathbb{F}_q$, et $y = (y_1,\ldots,y_n) \in \mathbb{F}_q^n$ à décoder. Soit $Q(X,Y)$, de degré minimal en Y, vérifiant $Q(x_i,y_i) = 0$, pour $i \in \{1,\ldots,n\}$. Alors*

1. *pour toute solution $f(X)$ du problème de décodage, telle que $Q(X, f) = 0$, il existe une position $i \in \{1, \ldots, n\}$ telle que $f(x_i) = y_i$, et $Q'(x_i, y_i) \neq 0$.*
2. *soit de plus f_1 et f_2 tels que $Q(X, f_1) = Q(X, f_2) = 0$, si de plus il existe i tel que $Q'(x_i, y_i) \neq 0$ et $f_1(x_i) = f_2(x_i)$, alors $f_1 = f_2$.*

Ainsi, quand $Q'(x_i, y_i) \neq 0$, on peut appliquer la méthode de Newton au polynôme $Q(X, Y) \in \mathbb{F}_q[X][Y]$, en itérant

$$f \leftarrow f - \frac{Q(X, f)}{Q'(X, f)} \bmod (X - x_i)^k, \tag{9.11}$$

où f est initialisée par $f \leftarrow y_i$. En effet, les solutions cherchées vérifient $f(x_i) = y_i$ en au moins un indice i tel que $Q'(x_i, y_i) \neq 0$: on peut donc prendre ce point comme initialisation de la méthode de Newton. On peut l'arrêter au bout de $\lceil \log k \rceil$ itérations, car nous ne sommes intéressés que par les solutions de degré au plus k, et la méthode de Newton est à convergence doublement quadratique. L'algorithme que nous proposons est donc :

Recherche de racines dans le cas de l'algorithme de Sudan

Données $X = \{x_1, \ldots, x_n\} \subset \mathbb{F}_q$, $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$, $Q(X, Y)$ de degré minimal vérifiant les conditions de l'étape d'interpolation.

Initialiser $I = \{i \in \{1, \ldots, n\}; Q'(x_i, y_i) \neq 0\}$, $L = \emptyset$

Boucle pour $i \in I$

1. faire les $\lceil \log_2 k \rceil$ itérations de Newton (9.11), en initialisant f par $f \leftarrow y_i$,
2. $L \leftarrow L \cup \{f\}$,
3. enlever de I les indices j tels que $f(x_j) = y_j$.

Vérification retourner les $f \in L$ tels que $d(f, y) < t$.

Les autres travaux traitant ce problème de recherche de racines sont ceux de Roth et Ruckenstein [RR00] (qui s'appliquent au cas où $s = 1$) et ceux de Gao et Shokrollahi [GS00] qui s'appliquent à l'algorithme de Guruswami-Sudan, avec $s > 1$. Dans ces travaux, à une étape donnée de l'algorithme, le polynôme $Q(X, Y)$ est spécialisé en Y, ce qui donne un polynôme univarié que l'on factorise par des méthodes classiques. Le résultat de cette factorisation donne des points de départ pour calculer le développement des solutions. Dans le cas de [RR00], on voit apparaître une méthode d'éclatement des points, alors que dans le cas [GS00], on utilise la méthode du polygone de Newton. Il est à noter que notre proposition évite cette étape de factorisation, car les points de départ de la méthode de Newton sont déjà disponibles, ce sont les y_i, tels que $Q'(x_i, y_i) \neq 0$. Une autre approche est proposée par Nielsen et Høholdt, où le problème est ramené à celui d'une factorisation dans un grand corps fini [NH00, HN99]. Ils traitent à la fois des codes de Reed-Solomon et des codes géométriques. Notre contribution s'applique aux codes de Reed-Solomon et aux codes géométriques, évite la factorisation univariée, ce qui a l'intérêt de rendre le décodage déterministe. En revanche, notre méthode ne permet de traiter le cas où $s > 1$, car il n'y a plus de point où $Q'(x_i, y_i) \neq 0$ (la multiplicité est supérieure à 1).

9.4 Généralisation aux codes géométriques

Une généralisation majeure des codes de Reed-Solomon est celle des *codes géométriques*, due à Goppa [Gop77]. Soit $\mathcal{X}$ une courbe projective, absolument irréductible, non singulière définie sur $\mathbb{F}_q$. Soient $X = \{P_1, \ldots, P_n\}$, n points distincts

$\mathbb{F}_q$-rationnels, et G un diviseur $\mathbb{F}_q$-rationnel de $\mathcal{X}$, tel qu'aucun de P_i ne soit dans le support de G. Soit $L(G)$ l'espace de fonctions sur la courbe :

$$L(G) = \{f \in \mathbb{F}_q(\mathcal{X}); \quad (f) + G \geq 0\}\,. \tag{9.12}$$

Le code géométrique $C(X, G)$, correspondant aux points de X et à l'espace $L(G)$, est

$$C(X, G) = \{(f(P_1), \ldots, f(P_n)); \quad f \in L(G)\}\,. \tag{9.13}$$

Si $\deg G < n$, alors c'est un code $[n, k, d]$, avec $k \geq \deg G + 1 - g$, et $d \geq n - \deg G$, où g est le genre $\mathcal{X}$. Si de plus $2g - 2 < \deg G < n$, alors la dimension du code est exactement $k = \deg G + 1 - g$. Cette généralisation présente de grandes potentialités : alors que dans le cas des codes de Reed-Solomon la longueur n est bornée par la taille q de l'alphabet $\mathbb{F}_q$, dans le cas des codes géométriques, à q fixé, on peut construire des courbes avec un nombre de points arbitrairement grand. De plus, il existe, dans la classe des codes de Goppa, des codes de paramètres asymptotiquement meilleurs que les codes aléatoires [TVZ82]. Enfin, en longueur petite, on peut construire de très bons codes [Bro98].

Nous pouvons reprendre point par point tout ce qui a été dit précédemment sur les codes de Reed-Solomon, et cela se généralise bien au codes géométriques. Cheng [Che05] a montré que le problème du décodage des codes géométriques est difficile, pour la classe des codes géométriques définis sur les courbes elliptiques, mais en utilisant, comme dans le cas des codes de Reed-Solomon, des tailles d'alphabet exponentielles en la la longueur du code.

L'algorithme de Sudan et l'algorithme de Guruswami-Sudan se généralisent bien, dans leur principe, aux codes géométriques. Cela a été dans le cas $s = 1$ (algorithme de Sudan) par Shokrollahi et Wasserman [SW99]. Guruswami et Sudan, en introduisant les multiplicités $s > 1$, ont aussi généralisé aux codes géométriques dits « à un point » : ce sont les codes pour lesquels le diviseur G est de la forme αP_∞, où P_∞ est un point particulier de la courbe, et où α est un entier. Il est frappant que la capacité de correction est encore $n - \sqrt{n(n-d')}$ où d' est la distance construite du code géométrique.

Dans sa thèse Lancelot Pecquet montre comment utiliser l'algorithme de Guruswami-Sudan dans le cas où le diviseur G n'est pas un diviseur à un point. Cela nous semble intéressant car il arrive que l'on puisse obtenir de meilleurs paramètres en considérant des codes à diviseur général, comme par exemple ceux obtenus avec la contruction de Xing [Xin01, Xin05]. Il obtient encore une capacité de correction de $n - \sqrt{n(n-d')}$.

Schématiquement, les algorithmes comportent toujours les deux grandes étapes d'interpolation et de recherche de racines. Plus précisément, la structure est :

constantes $\mathcal{X}$ une courbe définie sur $\mathbb{F}_q$, $P_1, \ldots, P_n$ n points $\mathbb{F}_q$-rationnels de $\mathcal{X}$, D un diviseur dont le support ne contient aucun des P_i.

données $y = (y_1, \ldots, y_n)$ le mot reçu.

résultat tous les $f \in L(D)$ tels que $d(f, y) \leq t$.

étape d'interpolation trouver un polynôme $Q(Y)$ in $\mathbb{F}_q(\mathcal{X})[Y]$ tel que

1. $Q(X) \neq 0$;
2. les coefficients Q_i de $Q(Y)$ sont dans des espaces $L(D_i)$ convenables ;
3. la « multiplicité »de $Q(Y)$ en (P_i, y_i) est supérieure ou égale à s.

étape de recherche de racines trouver les $f \in \mathbb{F}_q(\mathcal{X})$ tels que $Q(f) = 0$, avec $f \in L(D)$, et $d(f, y) \leq t$.

En ce qui concerne l'étape d'interpolation, Høholdt et Nielsen [HN99] étendent leur algorithme aux codes *hermitiens*, c'est-à-dire aux codes géométriques définis

que la courbe d'équation $X^{q+1} + Y^{q+1} + Z^{q+1} = 0$ sur $\mathbb{F}_{q^2}$. Sakata [SNF00, Sak01] a aussi traité le cas des codes hermitiens avec son algorithme BMS.

Avec Lancelot Pecquet, nous avons montré que la méthode de Newton, que nous utilisons avec des points de départ déjà connus, se généralise au cas des codes géométriques. Elle présente encore l'avantage d'éviter une factorisation univariée. souffre du même défaut, qui est de ne s'appliquer qu'au cas où s, l'ordre de multiplicité est égal à 1. D'autre part, Wu et Siegel [WS01] ont généralisé la méthode de factroisation de Roth et Ruckenstein aux codes géométriques, et dans le cadre général de l'algorithme de Guruswami-Sudan ($s \geq 1$). L'algorithme de Gao et Shokrollahi [GS00] s'applique aussi au cas des codes géométriques, et avec $s \geq 1$.

9.5 Généralisations multivariées

Il peut être fructueux de considérer des généralisations multivariées d'algorithmes « univariés ». On peut voir les algorithmes de Sudan et de Guruswami-Sudan comme des algorithmes univariés qui utilisent, à un moment donné de la preuve de leur correction, la propriété qu'un polynôme qui a plus de zéros, comptés avec multiplicités, que son degré, est identiquement nul. Dans le cas univarié, étant donné un ensemble $S = \{x_1, \ldots, x_n\}$ de n points distincts dans $\mathbb{F}_q$, nous considérions la fonction d'évaluation

$$\begin{array}{rccc} \mathrm{ev}: & \mathbb{F}_q[X] & \to & (\mathbb{F}_q)^n \\ & f(X) & \mapsto & \big(f(x_i)_{(x_i \in X)}\big) \end{array} \tag{9.14}$$

nous considérons $L = \{f \in \mathbb{F}_q[X], \deg f < k\}$ et le code de Reed-Solomon de dimension k est l'ensemble

$$\{\mathrm{ev}\, f; \quad f \in L\} \tag{9.15}$$

Dans le cas multivarié, nous construisons $S^m = \{x_1, \ldots, x_n\}^m$, et la fonction d'évaluation

$$\begin{array}{rccc} \mathrm{ev}^m: & \mathbb{F}_q[X_1, \ldots, X_m] & \to & (\mathbb{F}_q)^{n^m} \\ & f(X_1, \ldots, X_m) & \mapsto & (f(x_{i_1}, \ldots, x_{i_m}))_{(x_{i_1 \ldots, x_{i_m}) \in S^m}} \end{array} \tag{9.16}$$

et nous considérons deux possibilités pour l'espace L des polynômes à évaluer.

1. $L = \{f = f(X_1, \ldots, X_m),\ \deg f \leq r\}$, où deg est le degré total. Ce code correspond au code de Reed-Muller d'ordre r à m variables. Nous nous intéressons au cas $q > r$, pour lequel la dimension est $\binom{m+r}{r}$.
2. $L = \{f(X_1, \ldots, X_m),\ \deg_{X_i} f(X_1, \ldots, X_m) < k, i \in \{1, \ldots, m\}\}$, ce qui donnera le code $\otimes^m C_1$ qui est le code produit m fois du code de Reed-Solomon C_1 de dimension k. La dimension du code est k^m.

Enfin, du point de vue multivarié, citons les travaux de Parvaresh et Vardy [PV05b, PV05a], qui permettent de décoder simultanément plusieurs mots de code, en faisant l'hypothèse que les erreurs sur chaque mot sont localisées sur les mêmes positions. Cela revient à traiter un problème multivarié, mais différent du notre, où la fonction d'évaluation est la suivante :

$$\begin{array}{rccc} \mathrm{ev}_m: & \mathbb{F}_q[X] \times \cdots \times \mathbb{F}_q[X] & \to & (\mathbb{F}_q^n)^m \\ & (f_1(X), \ldots, f_m(X)) & \mapsto & (\mathrm{ev}\, f_1(X), \ldots, \mathrm{ev}\, f_m(X)), \end{array}$$

ce qui revient à considérer simultanément m mots du code C_k. De manière similaire à l'algorithme de Sudan, ils construisent un polynôme $Q(X_1, \ldots, X_M, Y)$, avec

des conditions d'interpolation qui conviennent. Si les erreurs, sur les mots reçus, occurrent sur les même positions, ils peuvent corriger un nombre d'erreurs de

$$t \leq n\left(1 - \sqrt[m+1]{\left(\frac{k-1}{n}\right)^m (1+\frac{1}{s})\cdots(1+\frac{m}{s})}\right)$$

dont la limite, quand s croît, est

$$n\left(1 - R^{\frac{m}{m+1}}\right),$$

ce qui est un résultat très favorable, supérieur au rayon $1 - R^{1/2}$ obtenu par Guruswami-Sudan.

9.6 Généralisation aux codes produits de codes de Reed-Solomon

9.6.1 L'algorithme

Nous donnons d'abord l'algorithme qui en fait est un « principe algorithmique », qui consiste, comme l'algorithme original, en deux étapes : une étape d'interpolation, et une étape de factorisation.

Décodage pour le degré partiel

Données $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$, $y = (y_{i_1,\ldots,i_m})_{(i_1,\ldots,i_m)} \in \mathbb{F}_q^{n^m}$, $k, \mu \in \mathbb{N}$.

Paramètres auxiliaires k' et s à déterminer dans l'analyse de l'algorithme.

Étape d'interpolation trouver un polynôme $Q(X_1, \ldots, X_m, Y)$ tel que

1. $Q(X_1, \ldots, X_m, Y) \neq 0$,
2. Si l'on écrit,
$$Q = \sum_{i_1,\ldots,i_m,j} q_{i_1,\ldots,i_m,j} X_1^{i_1} \ldots X_m^{i_m} Y^l \tag{9.17}$$
alors Q n'a pas de terme $q_{i_1,\ldots,i_m,j}$ si au moins un indice i_l vérifient $i_l + kj > k'$.
3. $\mathrm{mult}(Q(x_{i_1}, \ldots, x_{i_m}), y_{i_1,\ldots,i_m}) = s$, pour tout $(i_1, \ldots, i_m) \in \{1, \ldots, n\}^m$.

Factorisation trouver les $f = f(X_1, \ldots, X_m)$ tel que $Q(X_1, \ldots, X_m, f)$.

Vérification retourner les f tels que $\deg f \leq r$, et $d(f, y) < t$.

Lorsque l'on substitue la solution $f(X_1, \ldots, X_m)$ dans $Q(X_1, \ldots, X_m, Y)$, on obtient, par construction, un polynôme $Q_f = Q(X_1 \ldots, X_m, f)$ de degré en chacune de ses variables borné par k'. Mais nous n'avons pas de théorème semblable au cas univarié, qui permettrait de dire qu'un polynôme qui a plus de racines que son degré est identiquement nul. En revanche, nous pouvons prendre avantage de la structure de X^m, pour avoir tout de même un lemme bornant le nombre de zéros d'un tel polynôme.

9.6.2 Un lemme sur le nombre de zéros

Proposition 10. *Soit $Q = Q(x_1, \ldots, x_m)$ de degré au plus k' en chaque variable, soit $X = \{x_1, \ldots, x_m\}$ un ensemble de n points distincts de $\mathbb{F}_q$. La somme des multiplicités de Q sur l'ensemble X^m est au plus*

$$mk'n^{m-1} \tag{9.18}$$

Démonstration. Par récurrence. Le résultat est vrai pour $m = 1$. Supposons vrai le résultat pour $m - 1$. Soit J l'ensemble d'indices suivant :

$$J = \{j \in \{1, \ldots, n\}, Q(X_1, \ldots, X_{m-1}, x_j) = 0\}, \tag{9.19}$$

de telle sorte que l'on peut écrire, pour $j \in J$:

$$Q(X_1, \ldots, X_m) = (X_m - x_j)^{t_j} \tilde{Q}_j(X_1, \ldots, X_m), \tag{9.20}$$

avec $t_j > 0$. On a donc

$$Q(X_1, \ldots, X_m) = \tilde{Q}(X_1, \ldots, X_m) \prod_{j \in J} (X_m - x_j)^{t_j}, \tag{9.21}$$

et on note que $\deg_{X_m} \tilde{Q}(X_1, \ldots, X_m) = k' - \sum_{j \in J} t_j$. Soit S la somme des multiplicités de $Q(X_1, \ldots, X_m)$ sur X^n. Alors

$$\begin{aligned} S &= \sum_{(i_1, \ldots, i_m)} \text{mult}(Q(X_1, \ldots, X_m), (x_{i_1}, \ldots, x_{i_m})) \\ &= \sum_{i_m \in J'} \sum_{(i_1, \ldots, i_{m-1})} \text{mult}(Q(X_1, \ldots, X_m), (x_{i_1}, \ldots, x_{i_m})) \\ &\quad + \sum_{i_m \in J} \sum_{(i_1, \ldots, i_{m-1})} \text{mult}(Q(X_1, \ldots, X_m), (x_{i_1}, \ldots, x_{i_m})) \\ &= S' + S''. \end{aligned} \tag{9.22}$$

On calcule S' :

$$\begin{aligned} S' &= \sum_{i_m \in J'} \sum_{(i_1, \ldots, i_{m-1})} \text{mult}(Q(X_1, \ldots, X_m), (x_{i_1}, \ldots, x_{i_m})), \\ &= \sum_{i_m \in J'} \sum_{(i_1, \ldots, i_{m-1})} \text{mult}(\tilde{Q}(X_1, \ldots, X_m), (x_{i_1}, \ldots, x_{i_m})). \\ &\leq \sum_{i_m \in J'} \sum_{(i_1, \ldots, i_{m-1})} \text{mult}(Q(X_1, \ldots, X_{m-1}, x_{i_m}), (x_{i_1}, \ldots, x_{i_{m-1}})). \end{aligned}$$

L'hypothèse de récurrence nous permet de dire que

$$S' \leq \sum_{i_m \in J'} (m-1)k' n^{m-2} = |J'|(m-1)k' n^{m-2}. \tag{9.23}$$

De même

$$\begin{aligned} S'' &= \sum_{i_m \in J} \sum_{(i_1, \ldots, i_{m-1})} \text{mult}(Q(X_1, \ldots, X_m), (x_{i_1}, \ldots, x_{i_m})) \\ &= \sum_{i_m \in J} \sum_{(i_1, \ldots, i_{m-1})} \Big(t_j + \text{mult}(\tilde{Q}_j(X_1, \ldots, X_m), (x_{i_1, \ldots, i_m})) \Big) \\ &\leq n^{m-1} \sum_{i \in J} t_j + |J|(m-1)k' n^{m-2} \end{aligned} \tag{9.24}$$

Donc

$$\begin{aligned} S &= S' + S'' \\ &\leq n^{m-1} \sum_{j \in J} t_j + (|J| + |J'|)(m-1)k' n^{m-2} \\ &\leq n^{m-1} k' + (m-1)k' n^{m-1} \\ &= m k' n^{m-1} \end{aligned} \tag{9.25}$$

□

9.6.3 Analyse de l'algorithme

Armé de ce lemme, nous pouvons analyser l'algorithme. On en déduit :

Proposition 11. *Soit $Q(X_1, \ldots, X_m, Y)$ comme dans l'algorithme 9.6.1. Soit $f = f(X_1, \ldots, X_m)$ tel que $\deg_{X_i} f \leq k$ pour chaque indéterminée X_i, et tel que*

$$f(x_{i_1}, \ldots, x_{i_m}) = y_{i_1,\ldots,i_m} \tag{9.26}$$

pour au moins μ indices $(i_1, \ldots, i_m)$. Alors si μ est tel que

$$s\mu > mk' n^{m-1}, \tag{9.27}$$

on a $Q(X_1, \ldots, X_m, f) = 0$.

Enfin, pour justifier complètement l'algorithme, il faut s'assurer que le polynôme Q, avec les contraintes sur les degrés partiels, existe pour tout entrée $(y_1, \ldots, y_n)$. Pour cela, nous écrivons encore la condition

$$N_Q > N_{\text{eq}}, \tag{9.28}$$

où N_Q est le nombre de termes de Q et Neq le nombre d'équations. Le nombre d'équations est

$$N_{\text{eq}} = n^m \binom{m+1+s-1}{s-1} = n^m \binom{m+s}{m+1} = n^m \cdot \frac{s \ldots (s+m)}{(m+1)!}. \tag{9.29}$$

Le nombre de termes est

$$\frac{k'^{m+1}}{(m+1)k}$$

L'équation (9.28) donne

$$k' > n \sqrt[m+1]{\frac{k}{n} \cdot \frac{s \ldots (s+m)}{(m+1)!}}. \tag{9.30}$$

Pour finir, en combinant (9.30) et (9.27), on obtient :

Proposition 12. *L'algorithme précédent décode jusqu'à t erreurs, avec*

$$t = n^m - \mu \leq n^m \left(1 - \sqrt[m+1]{\frac{m^{m+1}}{m!} \cdot \frac{r}{n} \cdot (1 + \frac{1}{s} \cdots (1 + \frac{m}{s})}\right) \tag{9.31}$$

dont la limite est, quand s croît :

$$n^m \left(1 - \sqrt[m+1]{\frac{m^{m+1}}{m!} \cdot \frac{r}{n}}\right). \tag{9.32}$$

9.7 Généralisation aux codes de Reed et Muller

Nous pouvons construire l'algorithme suivant, qui est une généralisation très directe de l'algorithme de Guruswami-Sudan.

Décodage pour le degré total

données $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$, $y = (y_{i_1,\ldots,i_m})_{(i_1,\ldots,i_m)} \in \mathbb{F}_q^{n^m}$, $r, \mu \in \mathbb{N}$.

paramètres auxiliaires d et s à déterminer dans l'analyse de l'algorithme.

étape d'interpolation trouver un polynôme $Q(X_1, \ldots, X_m, Y)$ tel que

1. $Q(X_1, \ldots, X_m, Y) \neq 0$,
2. $\mathrm{wdeg}_{1,\ldots,1,r} Q(X_1, \ldots, X_m, Y) \leq d$,
3. $\mathrm{mult}\, Q(x_{i_1}, \ldots, x_{i_m}, y_{i_1,\ldots,i_m}) = s$, pour tout $(i_1, \ldots, i_m) \in \{1, \ldots, n\}^m$.

étape de recherche de racines retourner les facteurs $(Y - f)$ de Q, tels que $\deg f \leq r$ et $f(x_{i_1}, \ldots, x_{i_m}) = y_{i_1,\ldots,i_m}$ pour au moins μ indices $(i_1, \ldots, i_m)$.

On voit tout de suite que le polynôme $Q_f = Q(X_1, \ldots, X_m, f)$ est de degré total borné par d, et que son nombre de racines, comptés avec multiplicités est au moins sm. Encore une fois nous sommes confrontés au problème de borner le nombre de zéros, parmi X^m, d'un tel polynôme Q_f. Wu et Pellikaan [PW04b, PW04a] utilisent le théorie de l'escalier (footprint) d'un l'idéal. Il considère l'idéal

$$I(q, m) = \langle X_i^q - X_i; i \in \{1, \ldots, m\} \rangle$$

des polynômes qui s'annulent sur $\mathbb{F}_q^m$, puis l'idéal $I(q, m)^s$ des polynômes qui s'annulent avec la multiplicité s. Soit $I_{Q_f} = I(q, m)^s + f$, alors Wu et Pellikaan bornent inférieurement par une certaine quantité $b_{\inf}$ le nombre de points sous l'idéal I_{Q_f}, en utilisant que la somme des multiplicités de Q_f est supérieure ou égale à $s\mu$. D'un autre coté, il borne supérieurement par $b_{\sup}$ le nombre de points sous l'idéal I_{Q_f}, en utilisant que le degré de Q_f est borné par d. En écrivent la condition $b_{\inf} > b_{\sup}$, ils en déduisent que Q_f doit être identiquement nul. Au final, ils obtiennent un rayon de décodage

$$t < q^m \left(1 - \sqrt[m+1]{\frac{r}{q}(1 + \frac{1}{s}) \cdots (1 + \frac{m}{s})}\right)^m \underset{s \to +\infty}{\longrightarrow} n^m \left(1 - \sqrt[m+1]{\frac{r}{q}}\right)^m$$

De notre part, nous utiliserons une généralisation du lemme de Schwarz-Zippel [Sch80, Zip79] permet de borner le nombre de zéros, sur $\mathbb{F}_q^m$ d'un polynôme multivarié à m variables de degré inférieur à d par dq^{m-1} (c'est-à-dire que la proportion des zéros parmi tous les éléments de $\mathbb{F}_q^m$ est $\frac{d}{q}$). Nous avons besoin du même genre de lemme, mais en comptant les zéros avec multiplicité.

Lemme 3 (Schwarz-Zippel avec multiplicités). *Soit $Q = Q(X_1, \ldots, X_m) \in \mathbb{F}_q[X_1, \ldots, X_m]$ de degré total au plus d. Soient $X = \{x_1, \ldots, x_n\}$ un ensemble de n éléments distincts de $\mathbb{F}_q$. Alors la somme des multiplicités de Q, prise sur l'ensemble X^n, est bornée par dn^{m-1}.*

La preuve suit les mêmes lignes que la preuve du lemme 10. Nous avons donc :

Proposition 13. *Soit $Q(X_1, \ldots, X_m, Y)$ comme dans l'algorithme 9.7. Soit $f = f(X_1, \ldots, X_m)$ tel que $\deg f \leq r$, et tel que*

$$f(x_{i_1}, \ldots, x_{i_m}) = y_{i_1,\ldots,i_m} \tag{9.33}$$

pour au moins μ indices $(i_1, \ldots, i_m)$. Alors si

$$s\mu > dn^{m-1}, \tag{9.34}$$

on a $Q(X_1, \ldots, X_m, f) = 0$.

Pour finir d'étudier la capacité de correction de l'algorithme, nous devons déterminer quand est-ce que le polynôme Q de l'étape d'interpolation existe pour toute entrée de l'algorithme. Pour être sûr que Q existe quelque soit le mot reçu y, nous devons imposer encore

$$N_Q > N_{\mathrm{eq}} \tag{9.35}$$

où N_Q est le nombre de termes de Q, et N_{eq}. le nombre d'équations données par le point 3. de l'algorithme. Le nombre d'équations est le même que dans le cas des codes produits. Et on calcule que le nombre de termes N_Q d'un polynôme Q vérifiant la condition 2 d'interpolation est au moins.

$$\frac{d^{m+1}}{r(m+1)!}. \tag{9.36}$$

L'équation 9.35 donne la condition suffisante :

$$\frac{d^{m+1}}{r(m+1)!} > n^m \cdot \frac{s \dots (s+m)}{(m+1)!}. \tag{9.37}$$

En combinant (9.34)et (9.37), après calcul, on obtient :

Proposition 14. *L'algorithme précédent décode jusqu'à t erreurs, avec*

$$t \leq n^m(1 - \sqrt[m+1]{\frac{r}{n}(1+\frac{1}{s})\cdots(1+\frac{m}{s})}) \quad \underset{s\to+\infty}{\longrightarrow} \quad n^m\left(1 - \sqrt[m+1]{\frac{r}{n}}\right). \tag{9.38}$$

Ce qui est meilleur que le rayon de décodage obtenue par Pellikaan et Wu avec l'analyse utilisant les bases de Gröbner.

En résumé, les deux algorithmes de décodage, des codes produits de codes de Reed-Solomon, et de codes de Reed-Muller, voient leur performance se dégrader quand le nombre m de variables augmentent (à cause du facteur $\sqrt[m+1]{}$). C'est à l'opposé de l'algorithme de Parvaresh et Vardy, qui voit ses performances s'améliorer quand le nombre de variables augmente.

9.8 Spécificité des corps finis

Remarquons que ces algorithmes n'utilisent en aucun cas le fait que les codes sont définis sur $\mathbb{F}_q$. En réalité, on pourrait définir des codes théoriques sur n'importe quel corps, et ces algorithmes fonctionnerait parfaitement. En revanche, lorsque l'alphabet est $\mathbb{F}_q$, les codes de Reed-Muller prennent une structure particulière, suivant un résultat dû à Kasami, Lin et Peterson [KLP68]. Soit $RM_q(r,m)$ le code de Reed-Muller construit sur $X^m = \mathbb{F}_q^m$. Un mot de $RM_q(u,m)$ est donc du type

$$c = (f(x_{i_1}, \dots, x_{i_m}))_{(i_1,\dots,i_m)\in\{1,\dots,q\}^m} \tag{9.39}$$

où $\{x_1 = 0, \dots, x_q\}$ est une énumération de $\mathbb{F}_q$. Soit $RM_q(u,m)^*$ le code raccourci obtenu en enlevant de chaque mot de $RM_q(r,m)$ la composante correspondant à l'évaluation en $(x_1 = 0, \dots x_m = 0)$, on obtient un code de longueur $q^m - 1$.

Proposition 15. *Soit $d-1$ la distance minimale de $RM_q(r,m)^*$. Le code $RM_q(r,m)$ est le code*

$$C_0 \cap \mathbb{F}_q^{q^m-1}, \tag{9.40}$$

où C_0 est le code BCH primitif au sens strict de distance construite $d-1$ (qui est un cas particulier de code de Reed-Solomon).

Ainsi Pellikaan et Wu [PW04b], en utilisant alors l'algorithme de Guruswami-Sudan « univarié », obtiennent une capacité de correction de

$$n = n - \sqrt{(n-d)n} = \left(1 - \sqrt{r/q}\right) \tag{9.41}$$

Une autre construction utilise le fait que l'ensemble $\mathbb{A}^m(\mathbb{F}_q)$ des points $P \in \mathbb{F}_q^m$ constitue une courbe algébrique, lisse sur sa partie affine $\mathbb{A}^m$, ayant un seul

point à l'infini P_∞ correspondant à une seule place Q_∞, et de genre $g = \frac{1}{2}((q^2 - 1)(q+1)^{m-1} - q^{m+1} + 1)$. Les équations définissant cette courbe sont explicitement connues [PSvW91], et les bases des espace $L(lQ_\infty)$ sont aussi explicitement connues. De cette manière, ils interprètent le code de Reed-Muller comme un code géométrique à un point sur cette courbe. En utilisant encore l'algorithme de Guruswami-Sudan, cette fois pour les codes géométriques, Pellikaan et Wu obtiennent une capacité de correction de

$$n\left(1 - \sqrt{\frac{r(q+1)^{m-1}}{n}}\right) \tag{9.42}$$

qui est sensiblement égale à celle obtenue en (9.41) quand q est grand ($n = q^m$).

En ce qui concerne les codes produits de codes de Reed-Solomon, l'astuce de Parvaresh, El-Khamy, McEliece et Vardy [PEKMV] est de considérer le code produit $\otimes^2 C_1$ comme un sous-code du code de Reed et Muller de paramètre $r = 2k$ (en effet un polynôme de degré k en chacune de ses deux variables est de degré total au plus $2k$). En utilisant le résultat de Wu et Pellikaan, ils obtiennent ainsi une capacité de correction de

$$t < n\left(1 - \sqrt{\frac{r}{q}}\right). \tag{9.43}$$

Cinquième partie

Annexes

Annexe A

Informations annexes

A.1 Étudiants encadrés

A.1.1 Thèse de Lancelot Pecquet : décodage en liste des codes géométriques

Thèse soutenue le 18 décembre 2001, à l'Université Paris VI.

Lancelot Pecquet a travaillé sur les algorithmes de Sudan [Sud97], Shokrollahi-Wasserman [SW99] et Guruswami-Sudan [GS99], de décodage des codes de Reed-Solomon et des codes géométriques. Il a proposé une version de l'algorithme de Guruswami-Sudan pour les codes géométriques dits « à plusieurs points », c'est à dire lorsque l'espace des fonctions à évaluer n'est pas un espace de Riemman-Roch de la forme $L(kP)$, mais un espace de Riemann-Roch $L(D)$, avec D quelconque. Il a implémenté en Magma ces algorithmes.

Il a aussi proposé une méthode de décodage « souple », suivant le principe de Kötter et Vardy [KV03]. Le problème est le suivant : dans la chaîne de transmission, le dispositif physique est capable, pour chaque symbole reçu, de donner une *information de qualité* (soft information), plutôt que de choisir un symbole dans $\mathbb{F}_q$, le plus probable. Il est alors utile, pour améliorer le décodage, de prendre en compte cette information, pour trouver le mot de code le plus vraisemblable.

Les algorithmes de décodage par interpolation construisent un polynôme d'interpolation, dont les facteurs contiennent les solutions du problème de décodage. L'idée est de traduire les informations de qualité en conditions algébriques d'interpolation : les multiplicités. Ce principe a aussi été généralisé aux codes géométriques, et Lancelot Pecquet a proposé une règle différente que celle de Kötter et Vardy pour transformer les informations de qualité en multiplicités.

A.1.2 Thèse de Cédric Tavernier : testeurs, problèmes de reconstruction univariés et multivariés, et application à la cryptanalyse du DES

Thèse soutenue le 14 janvier 2005, à l'École Polytechnique.

La problématique est la suivante : on a accès à une fonction $f : \mathbb{F}_2^n \to \mathbb{F}_2$, avec n grand (par exemple, $n = 64$ ou $n = 128$), grâce à un oracle qui, lorsqu'on lui soumet $x \in \mathbb{F}_2^n$, retourne $f(x)$. Soit l la fonction $\mathbb{F}_2$-linéaire, inconnue, la plus proche de f au sens de la distance de Hamming. C'est-à-dire que l est la fonction $\mathbb{F}_2$-linéaire telle que la distance

$$d = |\{x \in \mathbb{F}_2^n; f(x) \neq l(x)\}|$$

est minimale. On se pose les questions suivantes :

- est-ce que d est plut petit qu'un seuil donné ?
- quelle est la fonction $\mathbb{F}_2$-linéaire la plus proche (au sens de la distance de Hamming) de f ?

On ne peut pas voir accès à la totalité du vecteur des valeurs $(f(x); x \in \mathbb{F}_2^n)$, car n est trop grand. On considère donc des algorithmes qui font un petit nombre de requêtes $f(x)$ pour un certain nombre de valeurs x, si possible polynomial en n.

On aura une réponse probabiliste aux deux questions ci-dessus. Le premier problème a été étudié initialement dans [GLR+91] et on appelle un *testeur* un programme qui teste si d est est en dessous d'un certain seuil. Le deuxième problème, qui s'apparente à de la correction d'erreurs, a connu une solution en temps polynomial avec l'algorithme de Goldreich-Rubinfeld-Sudan [GRS95, GRS00]. On parle de décodage local : on ne connait qu'une partie, infime, des valeurs de $f(x)$, et on veut retrouver la fonction l la plus proche de f.

Cédric Tavernier a montré que répondre au premier problème ne coûte pas moins cher que de déterminer la fonction linéaire la plus proche. Il a donné une complexité précise pour l'algorithme de Goldreich-Rubinfeld-Sudan, et a ensuite amélioré cette complexité, ainsi que le nombre d'appel à la fonction f.

Ces algorithmes peuvent être employés pour étudier certaines combinaisons linéaires des bits de sortie d'une fonction de chiffrement $f : F_2^n \times \mathbb{F}_2^k \to \mathbb{F}_2^n$, où n est la taille en bits des messages, k la taille en bits des clés. Cédric Tavernier a appliqué son algorithme au schéma du DES, et il a retrouvé automatiquement les meilleures approximations linéaires de certaines sur de l'algorithme de chiffrement DES, données par Matsui [Mat94b], ainsi que de nombreuses autres.

A.1.3 Thèse de Magali Bardet : étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie

Thèse soutenue le 8 décembre 2004, à l'université Paris VI. Co-direction avec Jean-Charles Faugère.

Magali Bardet a étudié des outils issus des méthodes de base de Gröbner pour traiter de certains problèmes en cryptographie et en théorie des codes. La partie codage est décrite, de manière détaillée dans le chapitre 8 du présent document. Essentiellement, le problème du décodage est mis en équations, et les propriétés du système d'équations, relativement au problème de décodage posé, sont étudiées. Elle a montré que les résolutions de ces systèmes, avec des algorithmes de calcul de bases de Gröbner, comme ceux que Jean-Charles Faugère a développés [Fau99, Fau02], est efficace. On obtient en effet un coût, en nombre d'opérations arithmétique dans un corps fini, comparable au coût du décodage d'un code BCH de même capacité de correction, mais en ayant la possibilité de corriger des codes cycliques quelconques.

L'autre partie de la thèse traite de l'étude de la complexité algorithmique de la résolution, avec des algorithmes de calcul de base de Gröbner, de systèmes d'équations *surdéterminés* généraux. Cela a de nombreuses applications en cryptographie, comme par exemple la cryptanalyse de HFE [FJ03].

A.1.4 Thèse de Raghav Bhaskar : protocoles cryptographiques pour les réseaux Ad Hoc

Thèse soutenue le 26 juin 2006. Co-direction avec Valérie Issarny.

Le sujet de départ était la sécurisation du système de fichiers distribué AdHocFS, développé dans le projet Arles de l'INRIA Rocquencourt. Alors qu'il semblait clair d'utiliser du chiffrement pour sécuriser les fichiers, le problème de la gestion de la clé restait posé. Il a semblé immédiatement intéressant d'élaborer une solution à base

de protocoles de mise en accord de clé, qui sont une généralisation à n utilisateurs du protocole de Diffie-Hellman [DH76], défini pour deux utilisateurs. Il n'y a pas de telle généralisation directe du protocole à n utilisateurs, et de nombreuses propositions ont été faites, avec de plus ou moins bonnes performances (en terme de la quantité des calculs à effectuer et de la quantité de message échangées) . Nous avons produit une contribution dans ce domaine, qui est meilleure à de nombreux points de vue que l'existant. Raghav Bhaskar a aussi prouvé la sécurité du protocole dans le modèle de sécurité de Bresson, Chevassut, Pointcheval [BCP02]. Une telle *preuve de sécurité* est essentiellement une preuve par l'absurde, où il est montré qu'un attaquant contre le protocole peut être utilisé pour résoudre le problème algorithmique dit de *Diffie-Hellman décisionnel*. À partir du moment où il est admis que ce problème est difficile, on en déduit qu'il n'y a pas d'attaquant efficace contre le protocole.

Enfin, nous avons étudié la mise en œuvre de ce protocole, à travers une collaboration avec le projet Hipercom de Rocquencourt(Paul Mühlethaler et Cédric Adjih) [BMA+06]. En effet, dans le monde cryptographique, les messages arrivent toujours à bon port, ce qui n'est pas le cas notamment dans les réseaux Ad Hoc, où le phénomène de perte de paquets est important. Il fallait donc adapter le protocole au contexte pratique des réseaux Ad Hoc, en utilisant des mécanismes réseaux. Du point de vue de l'application à de tels réseaux, notre protocole présente la qualité d'être très robuste : la défaillance d'un des participants au protocole, n'empêche pas les autres participants restants de mener le protocole à bien.

A.2 Enseignement

1. Cours de codes correcteurs d'erreurs dans le DEA Algorithmique, Paris VI, 2001-2004, 20 heures partagées avec Jean-Pierre Tillich.
2. Mastère MPRI (Mastère Parisien de Recherche en Informatique), 2004-2006, où j'enseigne toujours (51 heures, partagées avec Jean-Charles Faugère et Jean-Pierre Tillich). Ce dernier cours permet de développer les liens entre algorithmes de calcul formel et codage.
3. TD de Caml en license d'informatique à Paris VI (Thèrese Hardin enseignant) en 2000-2001. Ce module a ensuite été arrêté. Je me suis investi dans ce cours à cause de l'intérêt que j'avais pour le projet Focal.
4. TD de java à l'École polytechnique en 2001.
5. Cours de cryptologie dans le DEA Informatique Fondamentale et Applications de l'université de Marne-la-Vallée 2003-2005. Devenu le Mastère Informatique Recherche où j'enseigne toujours.

A.3 Publications

Articles longs dans des revues avec comité de lecture

1. Daniel Augot, Pascale Charpin, and Nicolas Sendrier. Sur une classe de polynômes scindés de l'algèbre $F_{2^m}[Z]$. *Comptes Rendus de l'Académie des Sciences*, 312 :649–751, 1991.
2. Daniel Augot, Pascale Charpin, and Nicolas Sendrier. Studying the locator polynomial of minimum weight codewords of BCH codes. *IEEE Transactions on Information Theory*, 38(3) :960–973, 1992.
3. Daniel Augot and Paul Camion. Forme de Frobenius et vecteurs cycliques. *Comptes rendus de l'Académie des Sciences*, 318 :183–188, 1993.

4. Daniel Augot and Nicolas Sendrier. Idempotents and the BCH bound. *IEEE Transactions on Information Theory*, 40(1) :204–207, January 1994.
5. Daniel Augot. Description of minimum weight codewords of cyclic codes by algebraic systems. *Finite Fields Appl.*, 2(2) :138–152, 1996.
6. Daniel Augot and Françoise Lévy dit Véhel. Bounds on the minimum distance of the duals of BCH code. *IEEE Transactions on Information Theory*, 1996.
7. Daniel Augot and Paul Camion. On the computation of minimal polynomials, cyclic vectors and Frobenius forms. *Linear Algebra and its Applications*, 260 :61–94, 1997.
8. Daniel Augot, Jean-Marc Boucqueau, Jean-François Delaigle, Caroline Fontaine, and Eddy Goray. Secure delivery of images over open networks. *Proceedings of the IEEE*, 87(7) :2605–2614, November 1999.
9. Daniel Augot and Lancelot Pecquet. A Hensel lifting to replace factorization in list-decoding of Algebraic-Geometric and Reed-Solomon codes. *IEEE Transactions on Information Theory*, 46(7) :2605–2614, November 2000.
10. Daniel Augot, Raghav Bhaskar, Valerie Issarny, and Daniele Sacchetti. A three round authenticated group key agreement protocol for adhoc networks. *Pervasive and Mobile Computing*, 3(1) :36–52, 2007.

Articles longs dans des conférences internationales avec comité de lecture

1. Daniel Augot, Pascale Charpin, and Nicolas Sendrier. Weights of some binary cyclic codes throughout the Newton's identities. In Gérard Cohen and Pascale Charpin, editors, *Eurocode 90*. Springer-Verlag, 1990.
2. Daniel Augot. A deterministic algorithm for computing a normal basis in a finite field. In *Eurocodes 94*, Abbaye de la Bussière, France, 1994.
3. Daniel Augot, Jean-Francois Delaigle, and Caroline Fontaine :. A scheme for managing watermarking keys in the Aquarelle multimedia distributed system. In Jean-Jacques Quisquater, Yves Deswarte, Catherine Meadows, and Dieter Gollmann, editors, *ESORICS*, number 1485 in Lecture Notes in Computer Science. Springer-Verlag, 1998.
4. Daniel Augot and Caroline Fontaine. Key issues for watermarking digital images. In *SPIE, EUROPTO - Conference on Electronic Imaging : Processing, Printing, and Publishing in Color, EUROPTO*, pages 176–185, Zürich, Suisse, 1998.
5. Daniel Augot and Matthieu Finiasz. A public key encryption scheme based on the polynomial reconstruction problem. In Eli Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, number 2656 in Lecture Notes in Computer Science, pages 229–240. Springer-Verlag, 2003.
6. Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In Ed Dawson and Serge Vaudenay, editors, *Progress in Cryptology - Mycrypt 2005 : First International Conference on Cryptology in Malaysia*, number 3715 in Lecture Notes in Computer Science, pages 64–83. Springer-Verlag, 2005.
7. Raghav Bhaskar, Daniel Augot, Valérie Issarny, and Daniele Sacchetti. An efficient group key agreement protocol for ad hoc networks. In *WOWMOM '05. IEEE Workshop on Trust, Security and Privacy in Ubiquitous Computing*, Taormina, Italy, 2005.
8. Daniel Augot, Mostafa El-Khamy, Robert J McEliece, Farzad Parvaresh, Mikhail Stepanov, and Alexander Vardy. List decoding of reed-solomon product

codes. In *Proceedings of the Tenth International Workshop on Algebraic and Combinatorial Coding Theory, Zvenigorod, Russia*, pages 210–213, September 2006.

Résumés courts dans des conférences internationales avec comité de lecture

1. Daniel Augot, Pascale Charpin, and Nicolas Sendrier. On the minimum weight of some binary BCH codes. In *IEEE Internation Symposium on Information Theory*, page 7, Budapest, Hungary, 1991.
2. Daniel Augot. Algebraic characterization of minimum weight codewords of cyclic codes. In *IEEE International Symposium on Information Theory 94*, Trondheim, Norway, June 1994.
3. Daniel Augot and Françoise Lévy dit Véhel. Bounds on the minimum distance of the duals of BCH codes. In *IEEE International Symposium on Information Theory*, Trondheim, Norway, 1994.
4. Daniel Augot. Algebraic equations for minimum weight codewords of many codes. In *International Conference on Finite Field and Applications*, Glasgow, Scotland, 1995.
5. Daniel Augot. Newton's identities for minimum codewords of a family of alternant codes. In *IEEE International Symposium on Information Theory (ISIT 95)*, Whistler, Canada, 1995.
6. Daniel Augot. Algebraic solutions of Newton's identities for cyclic codes. In *Proceedings of the 1998 Information Theory Workshop*, Killarney, Irland, 1998.
7. Daniel Augot. A parallel version of a special case of the sudan decoding algorithm. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, Lausanne, 2002.
8. Daniel Augot, Magali Bardet, and Jean-Charles Faugère. Efficient decoding of (binary) cyclic codes above the correction capacity of the code using Gröbner bases. In *IEEE International Symposium on Information Theory*, Yokohama, Japan, 2003.
9. Daniel Augot, Magali Bardet, and Jean-Charles Faugère. On formulas for decoding binary cyclic codes. In *IEEE Internation Symposium on Information Theory (ISIT 2007)*, Nice, France, 2007.

Conférences internationales

1. Daniel Augot and Nicolas Sendrier. Idemptotents and the BCH bound. In *Sixth Joint Swedish -Russian International Workshop on Information Theory*, Mölle, 1993.
2. Daniel Augot. Computing normal bases in finite fields. In Jacques Calmet, editor, *Rhine Workshop on Computer Algebra*, Karlsruhe, 1994.
3. Daniel Augot. Computing groebner bases for finding codewords of small weight. In *The 2nd IMACS Conference on Applications of Computer Algebra*, 1996.
4. Daniel Augot. Protection des droits et marquage d'image. In *Electronic Imaging and The Visual Arts (EVA'96)*, 1996.
5. Daniel Augot and Anne Canteaut. Cryptologie pour la sécurité des communications. In *CARI 98, colloque africain de recherche en informatique*, Dakar, Sénégal, 1998.

6. Nicolas Sendrier, Daniel Augot, Matthieu Finiasz, and Pierre Loidreau. Diversity in public key cryptography using coding theory and related problems. In *STORK cryptography workshop*, Bruges, Belgium, november 2002. workshop préparatoire du réseau d'excellence E-CRYPT.
7. Daniel Augot, Magali Bardet, and Jean-Charles Faugère. Decoding cyclic codes with algebraic systems. In Luxembourg Belgian (BMS), Dutch (KWG) and French (SMF) Mathematical Societies, editors, *Joint BeNeLuxFra Conference in Mathematics*, May 2005.
8. Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In European Network of Excellence in Cryptology ECRYPT, editor, *Ecrypt Conference on Hash Functions, Krakow, Poland*, Krakow, Poland, 2005. `http://www.ecrypt.eu.org/stvl/hfw/`.

Rappors techniques et de recherche

1. Daniel Augot, Pascale CHarpin, and Nicolas Sendrier. Studying the locator polynomials of minimum weight codewords of BCH codes. Technical Report 1488, INRIA, July 1991. `http://www.inria.fr`.
2. Daniel Augot and Paul Camion. The minimal polynomials, characteristic subspaces, normal bases and the Frobenius from. Technical Report 2006, INRIA, 1993. `http://www.inria.fr`.
3. Daniel Augot and Caroline Fontaine. D5.4 : IPR protection for multimedia assets, 1997.
4. Daniel Augot and Lancelot Pecquet. An alternative to factorization : a speedup for Sudan's decoding algorithm and its generalization to algebraic-geometric codes. Technical Report 3532, INRIA, Octobre 1998. `http://www.inria.fr`.
5. Daniel Augot, Magali Bardet, and Jean-Charles Faugère. Efficient decoding of (binary) cyclic codes beyond the correction capacity of the code using Gröbner bases. Technical Report 4652, INRIA, Novembre 2002. `http://www.inria.fr`.
6. Daniel Augot, Matthieu Finiasz, and Pierre Loidreau. Using the trace operator to repair the polynomial reconstruction based cryptosystem, presented at eurocrypt 2003. Cryptology ePrint Archive, Report 2003/209, 2003. `http://eprint.iacr.org/`.
7. Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A fast provably secure cryptographic hash function. `eprint.iacr.org`, 2003.
8. Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. Technical Report 5592, INRIA, Juin 2005. `http://www.inria.fr`.
9. Daniel Augot, François Morain, Caroline Fontaine, Jean Leneutre, Stéphane Maag, Anna Cavalli, and Farid Naït-Abdesselam. Review of vulnerabilities in mobile ad-hoc networks : trust and routing protocols views. Technical report, ACI SERAC, 2005.
10. A. Canteaut (Ed.), D. Augot, A. Biryukov, A. Braeken, C. Cid, H. Dobbertin, H. Englund, H. Gilbert, L. Granboulan, H. Handschuh, M. Hell, T. Johansson, A. Maximov, M. Parker, T. Pornin, B. Preneel, M. Robshaw, and M. Ward. Open research areas in symmetric cryptography and technical trends in lightweight cryptography. Technical report, Réseau d'excellence européen ECRYPT, 2005.

11. Raghav Bhaskar, Paul Mühlethaler, Daniel Augot, Cédric Adjih, and Saadi Boudjit. Efficient and dynamic group key agreement in Ad Hoc networks. Technical Report RR-5915, INRIA, 2006.

Colloques nationaux

1. Daniel Augot. Introduction à la cryptologie des courbes elliptiques. In *27ème École de printemps d'informatique théorique, Codage et cryptographie*, Batz-sur-mer, France, 1999.
2. Daniel Augot. Algorithme de décodage de sudan et cryptanalyse. In *Journées nationales de calcul formel*, CIRM, Luminy, 20-24 janvier 2003 2003.
3. Daniel Augot. Chiffrement et signature. In *École des Jeunes Chercheurs an Algorithmique et Calcul formel*. Institut Gaspard Monge, laboratoire d'Informatique, 31 mars - 4 avril 2003.

Articles dans des revues sans comité de lecture

1. Daniel Augot. Protection of intellectual property rights related to images. *ERCIM News*, April 1998. `http://www.ercim.org`.
2. Daniel Augot. Les travaux de M. Sudan sur les codes correcteurs d'erreurs. *Gazette des Mathematiciens*, 98, octobre 2003.
3. Daniel Augot. Madhu Sudan's work on error-correcting codes. *European Mathematical Society Newsletter*, 51 :8–10, March 2004. `http://www.emis.de/`.

Édition de comptes rendus

1. Daniel Augot and Claude Carlet, editors. *Workshop on Coding and Cryptography*, Paris, France, 1999. INRIA.
2. Daniel Augot and Claude Carlet, editors. *Workshop on Coding and Cryptography*, Paris, France, January 2001. ESAT.
3. Daniel Augot, Pascale Charpin, and Grigory Kabatianski, editors. *Workshop on Coding and Cryptography*, Versailles, France, March 2003. ESAT.

Thèse

1. Daniel Augot. *Étude algébrique des mots de poids minimum des codes cycliques, méthodes d'algèbre linéaire sur les corps finis.* PhD thesis, Université Pierre et Marie Curie, Paris 6, 1993.

A.4 Contrats de recherche

Projets industriels

1. Aquarelle, 1995-1997, coordonné par Alain Michard. Aquarelle était un projet de Recherche et Développement soutenu par le programme « Applications Télématiques » du cinquième PCRD de l'Union Européenne. Il s'agissait de mettre un place un système de partage de l'information sur le patrimoine culturel. Il a été mis en place un consortium européen regroupant des institutions culturelles, des éditeurs, des entreprises industrielles technologiques, et des laboratoires de recherche.

   Les partenaires culturels étaient très attentifs au problème de la protection des droits d'auteurs. J'ai contribué au livrable « IPR protection for multimedia assets », et avec Caroline Fontaine, nous avons défini un protocole de gestion de clés pour le marquage d'images. Le groupe TELE (Benoit Macq) de l'université catholique de Louvain a été sous-contractant, en fournissant un

algorithme de marquage. Nous avons réalisé un prototype intégrant la gestion des clés et l'algorithme de Louvain.

2. Canal+, février-juin 2002. Expertise, avec Anne Canteaut, d'un algorithme de chiffrement symétrique développé en interne à Canal+.
3. Banque de France. décembre 2003-mars 2004. Étude de la possibilité de mettre un authentifiant variable sur chaque billet, dépendant de données inscrites invisibles.

Projets de recherche :

1. Direction de l'Action de Recherche Concertée de l'INRIA « ARC courbes », 1999-2001 : rassemblement du LIX (François Morain), de Limoges (Iwan Duursma), et du projet CODES, en géométrie algébrique et théorie des nombres, pour le codage et la cryptographie.
2. ACI Polycrypt, 2001-2004 : avec le projet Spaces (Jean-Charles Faugère et Guillaume Hanrot) et le LACO de l'université de Limoges (Philippe Gaborit) : étude des systèmes polynomiaux intervenant dans les systèmes cryptographiques. C'est dans ce cadre que Jean-Charles Faugère a cassé le cryptosystème HFE.
3. ACI SERAC, 2004-2007 : models and protocols for SEcuRity in wireless Ad hoC networks. Avec Farid Naït-Abdesselam de l'USTL (Université des Sciences et Techniques de Lille), Jean Leneutre du GET (Groupe des Écoles des Télécommunications) et managé par Caroline Fontaine. L'INRIA est présente avec Codes, Tanc et Hipercom qui proposent des algorithmes et des protocoles cryptographiques bien adaptés aux réseaux Ad Hoc. Je coordonne le pôle INRIA de ce projet.

Bibliographie

[ABD+99] Daniel Augot, Jean-Marc Boucqueau, Jean-François Delaigle, Caroline Fontaine, and Eddy Goray. Secure delivery of images over open networks. *Proceedings of the IEEE*, 87(7) :2605–2614, November 1999.

[ABF02] Daniel Augot, Magali Bardet, and Jean-Charles Faugère. Efficient decoding of (binary) cyclic codes beyond the correction capacity of the code using Gröbner bases. Technical Report 4652, INRIA, Novembre 2002. `http://www.inria.fr`.

[ABF03] Daniel Augot, Magali Bardet, and Jean-Charles Faugère. Efficient decoding of (binary) cyclic codes above the correction capacity of the code using Gröbner bases. In *Proceedings of the 2003 IEEE International Symposium on Information Theory*, pages 362–362, 2003.

[ABF05] Daniel Augot, Magali Bardet, and Jean-Charles Faugère. Decoding cyclic codes with algebraic systems. In Luxembourg Belgian (BMS), Dutch (KWG) and French (SMF) Mathematical Societies, editors, *Joint BeNeLuxFra Conference in Mathematics*, May 2005.

[ABIS05] D. Augot, R. Bhaskar, V. Issarny, and D. Sacchetti. An efficient group key agreement protocol for ad hoc networks. In *World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, pages 576–580, 2005.

[ABIS07] Daniel Augot, Raghav Bhaskar, Valerie Issarny, and Daniele Sacchetti. A three round authenticated group key agreement protocol for adhoc networks. *Pervasive and Mobile Computing*, 3(1) :36–52, 2007.

[ACS90] Daniel Augot, Pascale Charpin, and Nicolas Sendrier. Weights of some binary cyclic codes throughout the Newton's identities. In Gérard Cohen and Pascale Charpin, editors, *Eurocode 90*. Springer-Verlag, 1990.

[ADF98] Daniel Augot, Jean-Francois Delaigle, and Caroline Fontaine :. A scheme for managing watermarking keys in the Aquarelle multimedia distributed system. In Jean-Jacques Quisquater, Yves Deswarte, Catherine Meadows, and Dieter Gollmann, editors, *ESORICS*, number 1485 in Lecture Notes in Computer Science. Springer-Verlag, 1998.

[AEKM+06] Daniel Augot, Mostafa El-Khamy, Robert J McEliece, Farzad Parvaresh, Mikhail Stepanov, and Alexander Vardy. List decoding of reed-solomon product codes. In *Proceedings of the Tenth International Workshop on Algebraic and Combinatorial Coding Theory, Zvenigorod, Russia*, pages 210–213, September 2006.

[AF97] Daniel Augot and Caroline Fontaine. D5.4 : IPR protection for multimedia assets. Technical report, European Commission, 1997.

[AF98] Daniel Augot and Caroline Fontaine. Key issues for watermarking digital images. In *SPIE, EUROPTO - Conference on Electronic Imaging :*

Processing, Printing, and Publishing in Color, EUROPTO, pages 176–185, Zürich, Suisse, 1998.

[AF03] Daniel Augot and Matthieu Finiasz. A public key encryption scheme based on the polynomial reconstruction problem. In *Advances in cryptology—EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 229–240. Springer, 2003.

[AFL03] Daniel Augot, Matthieu Finiasz, and Pierre Loidreau. Using the trace operator to repair the polynomial reconstruction based cryptosystem presented at Eurocrypt 2003. Cryptology ePrint Archive, Report 2003/209, 2003.

[AFS03] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A fast provably secure cryptographic hash function. `eprint.iacr.org`, 2003.

[AFS05a] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In European Network of Excellence in Cryptology ECRYPT, editor, *Ecrypt Conference on Hash Functions, Krakow, Poland*, Krakow, Poland, 2005. `http://www.ecrypt.eu.org/stvl/hfw/`.

[AFS05b] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. In Ed Dawson and Serge Vaudenay, editors, *Progress in Cryptology - Mycrypt 2005 : First International Conference on Cryptology in Malaysia*, number 3715 in Lecture Notes in Computer Science, pages 64–83. Springer-Verlag, 2005.

[AFS05c] Daniel Augot, Matthieu Finiasz, and Nicolas Sendrier. A family of fast syndrome based cryptographic hash functions. Technical Report 5592, INRIA, Juin 2005. `http://www.inria.fr`.

[AKS04] Arshad Ahmed Ahmed, Ralf Kötter, and Naresh R. Shanbhag. VLSI architectures for soft-decision decoding of Reed-Solomon codes. In *2004 IEEE International Conference on Communications*, volume 5, pages 2584–2590 Vol.5, 2004.

[Ale02] Michael Alekhnovich. Linear diophantine equations over polynomials and soft decoding of Reed-Solomon codes. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, Vancouver, Canada*, pages 439–448, 2002.

[Ale05] Michael Alekhnovich. Linear diophantine equations over polynomials and soft decoding of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 51(7) :2257–2265, 2005.

[AP98a] Daniel Augot and Lancelot Pecquet. An alternative to factorization : a speedup for Sudan decoding algorithm and its generalization to algebraic-geometric codes. Technical Report 3532, INRIA, 1998. Avalaible on `http://www.inria.fr`.

[AP98b] Daniel Augot and Lancelot Pecquet. An alternative to factorization : a speedup for Sudan's decoding algorithm and its generalization to algebraic-geometric codes. Technical Report 3532, INRIA, Octobre 1998. `http://www.inria.fr`.

[AP00a] Daniel Augot and Lancelot Pecquet. A Hensel lifting to replace factorization in list-decoding of Algebraic-Geometric and Reed-Solomon codes. *IEEE Transactions on Information Theory*, 46(7) :2605–2614, November 2000.

[AP00b] Daniel Augot and Lancelot Pecquet. A Hensel lifting to replace factorization in list decoding of algebraic-geometric and Reed-Solomon

codes. *IEEE Transactions on Information Theory*, 46(7) :2605–2613, Nov 2000.

[AST00] G. Ateniese, M. Steiner, and G. Tsudik. New multiparty authentication services and key agreement protocols. *Selected Areas in Communications, IEEE Journal on*, 18(4) :628–639, 2000.

[Aug93] Daniel Augot. *Étude algébrique des mots de poids minimum des codes cycliques, méthodes d'algèbre linéaire sur les corps finis.* PhD thesis, Université Pierre et Marie Curie, Paris 6, 1993.

[Aug94] Daniel Augot. Algebraic characterization of minimum weight codewords of cyclic codes. In *IEEE International Symposium on Information Theory 94*, Trondheim, Norway, June 1994.

[Aug96a] Daniel Augot. Description of minimum weight codewords of cyclic codes by algebraic systems. *Finite Fields Appl.*, 2(2) :138–152, 1996.

[Aug96b] Daniel Augot. Protection des droits et marquage d'image. In *Electronic Imaging and The Visual Arts (EVA'96)*, 1996.

[Aug98] Daniel Augot. Algebraic solutions of Newton's identities for cyclic codes. In *Proceedings of the 1998 Information Theory Workshop*, Killarney, Irland, 1998.

[Bar98] A. Barg. Complexity issues in coding theory. In V. S. Pless and W. C. Huffman, editors, *Handbook of Coding Theory*, volume I. North-Holland, 1998.

[Bar04] Magali Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie.* PhD thesis, Université Pierre et Marie Curie, Paris 6, 2004.

[BC04] Emmanuel Bresson and Dario Catalano. Constant round authenticated group key agreement via distributed computation. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *Public Key Cryptography ŽŽ013 PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 115–129, Berlin / Heidelberg, 2004. Springer.

[BCJ+05] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and reduced SHA-1. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, Aarhus, Denmark*, volume 3494 of *Lectures Notes in Computer Science*, pages 36–57. Springer, 2005.

[BCP02] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Dynamic group Diffie-Hellman key exchange under standard assumptions. In Lars Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, Amsterdam, The Netherlands,*, volume 2332 of *Lecture Notes in Computer Science*, pages 321–336. Springer, 2002.

[BCPQ01] Emmanuel Bresson, Olivier Chevassut, David Pointcheval, and Jean-Jacques Quisquater. Provably authenticated group Diffie-Hellman key exchange. In *CCS '01 : Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 255–264, New York, NY, USA, 2001. ACM Press.

[BD95] Mike V. D. Burmester and Yvo Desmedt. A secure and efficient conference key distribution system. In Alfredo De Santis, editor, *Advances in Cryptology ŽŽ014 EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 275–286, Berlin / Heidelberg, 1995. Springer.

[Ber68] Elwyn R. Berlekamp. *Algebraic coding theory.* McGraw-Hill, 1968.

[Bha06] Raghav Bhaskar. *Protocoles Cryptographiques Pour Les Réseaux Mobiles Ad Hoc*. PhD thesis, École Polytechnique, June 2006.

[BM06] Emmanuel Bresson and Mark Manulis. Extended definitions of AKE- and MA-security for group key exchange protocols. Cryptology ePrint Archive, November 2006. `http://eprint.iacr.org/`.

[BMA+06] Raghav Bhaskar, Paul Mühlethaler, Daniel Augot, Cédric Adjih, and Saadi Boudjit. Efficient and dynamic group key agreement in Ad Hoc networks. Technical Report RR-5915, INRIA, 2006.

[BMvT78] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3) :384–386, 1978.

[BN90] J. Bruck and M. Naor. The hardness of decoding linear codes with preprocessing. *IEEE Transactions on Information Theory*, 36(2) :381–385, 1990.

[BR94] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, Berlin / Heidelberg, 1994. Springer.

[Bro98] A. E. Brouwer. Bounds on the size of linear codes. In V. S. Pless and W. C. Huffman, editors, *Handbook of Coding Theory*, volume I, pages 295–461. North-Holland, 1998.

[BS04] Andrew Brown and Amin Shokrollahi. Algebraic-geometric codes on the erasure channel. In *Proceedings of the International Workshop on Information Theory, 2004, ISIT 2004, Chicago*, page 76, 2004.

[BW98] Klaus Becker and Uta Wille. Communication complexity of group key distribution. In Li Gong and Michael Reiter, editors, *CCS '98 : Proceedings of the 5th ACM conference on Computer and communications security*, pages 1–6, New York, NY, USA, 1998. ACM Press.

[CDF06] Ingemar J. Cox, Gwenaël Doërr, and Teddy Furon. Watermarking is not cryptography. In *Digital Watermarking*, volume 4283 of *Lecture Notes in Computer Science*, pages 1–15, Berlin / Heidelberg, 2006. Springer.

[CFS01] Nicolas T. Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a Mceliece-based digital signature scheme. In C. Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 157–174. Springer, 2001.

[Che05] Qi Cheng. Hard problems of algebraic geometry codes. `http://arxiv.org/abs/cs.IT/0507026`, July 2005.

[Chi64] R. T. Chien. Cyclic decoding procedures for Bose-Chaudhuri-Hocquenghem codes. *IEEE Transactions on Information Theory*, 10(4) :357–363, 1964.

[CI90] A. B. Cooper III. Direct solution of bch syndrome equations. In E. Arkian, editor, *Communications, Control, and Signal Processing*, pages 281–286. Elsevier, 1990.

[CI91a] A. B. Cooper III. Finding BCH error locator polynomials in one step. *Electronics Letters*, 27(22) :2090–2091, 1991.

[CI91b] A. B. Cooper III. A one-step algorithm for finding BCH error locator polynomials. In *International Symposium ion Information Theory, ISIT 1991*, pages 93–93, 1991.

[CJ04] Jean-Sébastien Coron and Antoine Joux. Cryptanalysis of a provably secure cryptographic hash function. Cryptology ePrint Archive, 2004. `http://eprint.iacr.org/`.

[CLO05] David A. Cox, John Little, and Donal O'Shea. *Using Algebraic Geometry*. Graduate Texts in Mathematics. Springer, March 2005.

[CLS06] Scott Contini, Arjen K. Lenstra, and Ron Steinfeld. VSH, an efficient and provable collision-resistant hash function. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 165–182. Springer, 2006.

[CM02] Massimo Caboara and Teo Mora. The Chen-Reed-Helleseth-Truong decoding algorithm and the Gianni-Kalkbrenner Gröbner shape theorem. *Applicable Algebra in Engineering, Communication and Computing*, 13(3) :209–232, 2002.

[Cor03a] Jean-Sébastien Coron. Cryptanalysis of the repaired public-key encryption scheme based on the polynomial reconstruction problem. Cryptology ePrint Archive, Report 2003/219, October 2003.

[Cor03b] Jean-Sébastien Coron. Cryptanalysis of the repaired public-key encryption scheme based on the polynomial reconstruction problem. Cryptology ePrint Archive, 2003. `eprint.iacr.org`.

[Cor04] Jean-Sébastien Coron. Cryptanalysis of a public-key encryption scheme based on the polynomial reconstrution problem. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *Public key cryptography—PKC 2004, Singapore*, volume 2947 of *Lecture Notes in Computer Science*, pages 14–27. Springer, 2004.

[Cov73] T. Cover. Enumerative source encoding. *IEEE Transactions on Information Theory*, 19(1) :73–77, 1973.

[CRHT94a] X. Chen, I. S. Reed, T. Helleseth, and T. K. Truong. General principles for the algebraic decoding of cyclic codes. *IEEE Transactions on Information Theory*, 40(5) :1661–1663, September 1994.

[CRHT94b] Xuemin Chen, I. S. Reed, T. Helleseth, and T. K. Truong. Algebraic decoding of cyclic codes : a polynomial ideal point of view. In Gary L. Mullen and Peter J. Shiue, editors, *Finite fields : theory, applications, and algorithms*, number 168 in Contemporary Mathematics, pages 15–22. American Mathematical Society, 1994.

[CRHT94c] Xuemin Chen, I. S. Reed, T. Helleseth, and T. K. Truong. Algebraic decoding of cyclic codes : a polynomial ideal point of view. In *Finite fields : theory, applications, and algorithms (Las Vegas, NV, 1993)*, volume 168 of *Contemporary Mathematics*, pages 15–22. American Mathematical Society, 1994.

[CRHT94d] Xuemin Chen, I. S. Reed, T. Helleseth, and T. K. Truong. Use of Gröbner bases to decode binary cyclic codes up to the true minimum distance. *IEEE Transactions on Information Theory*, 40(5) :1654–1661, 1994.

[CRT94] Xiaowei Chen, Irving S. Reed, and Trieu-Kien Truong. Decoding the (73, 37, 13) quadratic residue code. *IEE Proceedings on Computers and Digital Techniques*, 141(5) :253–258, 1994.

[CTR+03] Yaotsu Chang, Trieu-Kin Truong, Irving S. Reed, H. Y. Cheng, and C. D. Lee. Algebraic decoding of (71, 36, 11), (79, 40, 15), and (97, 49, 15) quadratic residue codes. *IEEE Transactions on Communications*, 51(9) :1463–1473, 2003.

[DB05] R. Dutta and R. Barua. Dynamic group key agreement in tree-based setting. In C. Boyd and González J. M. Nieto, editors, *Proc. 10th Australasian Conference on Information Security and Privacy (ACISP 2005)*, volume 3574 of *Lecture Notes in Computer Science*, pages 101–112, Berlin / Heidelberg, 2005. Springer.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6) :644–654, 1976.

[DN92] Stefan M. Dodunekov and Jan E. Nilsson. Algebraic decoding of the Zetterberg codes. *IEEE Transactions on Information Theory*, 38(5) :1570–1573, 1992.

[DN93] Stefan M. Dodunekov and Jan E. Nilsson. Algebraic decoding of Zetterberg and Dumer-Zinoviev codes. In *Proceedings of the 1993 IEEE International Symposium on Information Theory*, pages 306–306, 1993.

[DN94] Stefan M. Dodunekov and Jan E. Nilsson. Parallel decoding of the [23, 12, 7] binary Golay code. *IEE Proceedings on Computers and Digital Techniques*, 141(2) :119–122, 1994.

[Dr90] Ivan Bjerre Damgå rd. A design principle for hash functions. In G. Brassard, editor, *Advances in cryptology—CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 1990.

[Eli57] Peter Elias. List decoding for noisy channels. Technical report, Research Lab. of Electronics, MIT, 1957.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1-3) :61–88, June 1999.

[Fau02] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, Lille, France*. ACM, 2002.

[Fed06] Sergei V. Fedorenko. A method for computation of the discrete Fourier transform over a finite field. *Problems of Information Transmission*, 42(2) :139–151, June 2006.

[FGLM93] J. C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16(4) :329–344, 1993.

[FGT01] Elisabetta Fortuna, Patrizia Gianni, and Barry Trager. Degree reduction under specialization. *Journal of Pure Applied Algebra*, 164(1-2) :153–163, 2001.

[Fin04] Matthieu Finiasz. *Nouvelles constructions utilisant des codes correcteurs d'erreurs en cryptographie à clef publique,*. PhD thesis, École Polytechnique, October 2004.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases. In Dan Boneh, editor, *Advances in cryptology—CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Springer, 2003.

[Fon98] Caroline Fontaine. *Contribution à la recherche de fonctions booléennes hautement non linéaires, et au marquage d'images en vue de la protec-*

tion des droits d'auteur. PhD thesis, Université Paris VI, November 1998.

[FT91] G. L. Feng and K. K. Tzeng. Decoding cyclic and BCH codes up to actual minimum distance using nonrecurrent syndrome dependence relations. *IEEE Transactions on Information Theory*, 37(6) :1716–1723, 1991.

[Fur05] Teddy Furon. A survey of watermarking security. In *Digital Watermarking*, volume 3710 of *Lecture Notes in Computer Science*, pages 201–215, Berlin/Heidelberg, 2005. Spinger.

[Gia89] Patrizia Gianni. Properties of Gröbner bases under specializations. In James Davenport, editor, *Eurocal '87 : European Conference on Computer Algebra, Leipzig GDR*, volume 378 of *Lecture Notes in Computer Science*, pages 293–297. Springer, 1989.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. Series of Books in the Mathematical Sciences. W.H. Freeman & Company, January 1979.

[GLR+91] Peter Gemmell, Richard Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In Cris Koutsougeras and Jeff Vitter, editors, *STOC'91 : Proceedings of the twenty-third annual ACM symposium on Theory of computing, New Orleans, United States*, pages 33–42. ACM Press, 1991.

[Gop77] V. D. Goppa. Codes that are associated with divisors. *Problemy Pereda ci Informacii*, 13(1) :33–39, 1977.

[Gop81] V. D. Goppa. Codes on algebraic curves. *Dokl. Akad. Nauk SSSR*, 259(6) :1289–1290, 1981.

[Gri86] Dima Grigoriev. Polynomial factoring over a finite field and solving systems of algebraic equations. *J. Soviet Math.*, 34 :1762–1803, 1986.

[GRS95] Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries : the highly noisy case. In *FOCS : 36th Annual Symposium on Foundations of Computer Science, Milwaukee, United States*, pages 294–303. IEEE Computer Society Press, 1995.

[GRS00] Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries : the highly noisy case. *SIAM Journal on Discrete Mathematics*, 13(4), 2000.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6) :1757–1767, 1999.

[GS00] Shuhong Gao and Amin M. Shokrollahi. Computing roots of polynomials over function fields of curves. In David Joyner, editor, *Proceedings of the Annapolis Conference on Number Theory, Coding Theory, and Cryptography 1999*, pages 214–228. Springer, 2000.

[Gur04] Venkatesan Guruswami. *List Decoding of Error-Correcting Codes - Winning Thesis of the 2002 ACM Doctoral Dissertation Competition*, volume 3282 of *Lectures Notes in Computer Science*. Springer, December 2004.

[GV05] Venkatesan Guruswami and Alexander Vardy. Maximum-likelihood decoding of Reed-Solomon codes is NP-hard. *IEEE Transactions on Information Theory*, 51(7) :2249–2256, 2005.

[HH93] Russel J. Higgs and John F. Humphreys. Decoding the ternary Golay code. *IEEE Transactions on Information Theory*, 39(3) :1043–1046, 1993.

[HH95] Russel J. Higgs and John F. Humphreys. Decoding the ternary (23, 12, 8) quadratic residue code. *IEE Proceedings on Communications*, 142(3) :129–134, 1995.

[HH98] Russell J. Higgs and John F. Humphreys. Decoding the quintic $(11, 6, 5)$ quadratic residue code. *Mathematical Proceedings of the Royal Irish Academy*, 98A(1) :47–51, 1998.

[HN99] T. Hoeholdt and Refslund R. Nielsen. Decoding hermitian codes with sudan's algorithm. In M. Fossorier, H. Imai, S. Lin, and A. Poli, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes : 13th International Symposium, AAECC-13, Honolulu, USA*, volume 1719 of *Lecture Notes in Computer Science*, pages 260–269, Berlin, 1999. Springer.

[HP95] Tom Høholdt and Ruud Pellikaan. On the decoding of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 41(6) :1589–1614, 1995.

[HRTC01] Ruhua He, Irving S. Reed, Trieu-Kien Truong, and Xuemin Chen. Decoding the (47,24,11) quadratic residue code. *IEEE Transactions on Information Theory*, 47(3) :1181–1186, 2001.

[HT72] Carlos R. Hartmann and Kenneth K. Tzeng. Generalizations of the BCH bound. *Information and Control*, 20(5) :489–498, 1972.

[Hug94] E. Hughes. An encrypted key transport protocol. CRYPTO Rump Session, August 1994.

[Hum92] John F. Humphreys. Algebraic decoding of the ternary (13,7,5) quadratic residue code. *IEEE Transactions on Information Theory*, 38(3) :1122–1125, 1992.

[ITW82] I. Ingemarsson, D. Tang, and C. Wong. A conference key distribution system. *IEEE Transactions on Information Theory*, 28(5) :714–720, 1982.

[Jak98] Thomas Jakobsen. Cryptanalysis of block ciphers with probabilistic non-linear relations of low degree. In *Advances in Cryptology, CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 212–222, 1998.

[JH04] Jom Justesen and Tom Høholdt. *A Course in Error-Correcting Codes*. EMS Textbooks in Mathematics. European Mathematical Society, February 2004.

[K̈96] Ralf Kötter. *On Algebraic Decoding of Algebraic-Geometric and Cyclic Codes*. PhD thesis, University of Linköping, 1996.

[Kal85] Erich Kaltofen. Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM Journal on Computing (SICOMP)*, 14(2) :469–489, 1985.

[KLL04] Hyun-Jeong Kim, Su-Mi Lee, and Dong H. Lee. Constant-round authenticated group key exchange for dynamic groups. In P. J. Lee, editor, *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 245–259, Berlin / Heidelberg, 2004. Springer.

[KLP68] T. Kasami, Shu Lin, and W. Peterson. New generalizations of the Reed-Muller codes–I : Primitive codes. *IEEE Transactions on Information Theory*, 14(2) :189–199, 1968.

[KPT00] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *CCS '00 : Proceedings of the 7th ACM conference on Computer and communications security*, pages 235–244, New York, NY, USA, 2000. ACM Press.

[KPT04] Y. Kim, A. Perrig, and G. Tsudik. Group key agreement efficient in communication. *IEEE Transactions on Computer*, 53(7) :905–921, 2004.

[KV03] Ralf Kötter and Alexander Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 49(11) :2809–2825, 2003.

[KY01a] Aggelos Kiayias and Moti Yung. Polynomial reconstruction based cryptography (a short survey). In S. Vaudenay and A. M. Youssef, editors, *Selected Areas in Cryptography : 8th Annual International Workshop, SAC 2001 Toronto, Canada*, volume 2259 of *Lecture Notes in Computer Science*, pages 129–133, Berlin, 2001. Springer.

[KY01b] Aggelos Kiayias and Moti Yung. Secure games with polynomial expressions. In F. Orejas, P. G. Spirakis, and J. van Leeuwen, editors, *Automata, Languages and Programming : 28th International Colloquium, ICALP 2001*, volume 2076 of *Lecture Notes in Computer Science*, pages 393–950. Springer, July 2001.

[KY02a] Aggelos Kiayias and Moti Yung. Cryptographic hardness based on the decoding of Reed-Solomon codes. In P. Widmayer, F. Triguero, R. Morales, M. Hennessy, S. Eidenbenz, and R. Conejo, editors, *Automata, Languages and Programming : 29th International Colloquium, ICALP 2002, Malaga, Spain*, volume 2380 of *Lecture Notes in Computer Science*, pages 232–243. Springer, July 2002.

[KY02b] Aggelos Kiayias and Moti Yung. Cryptographic hardness based on the decoding of Reed-Solomon codes with applications. `http://eccc.uni-trier.de/eccc-reports/2002/TR02-017/`, 2002.

[KY03] Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125, Berlin / Heidelberg, 2003. Springer.

[KY04] Aggelos Kiayias and Moti Yung. Cryptanalyzing the polynomial-reconstruction based public-key system under optimal parameter choice. In Pil J. Lee, editor, *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 401–416. Springer, December 2004.

[KY05] Aggelos Kiayias and Moti Yung. Cryptography and decoding Reed-Solomon codes as a hard problem. In *IEEE Information Theory Workshop on Theory and Practice in Information-Theoretic Security*, pages 48–54, 2005.

[Len85] Arjen K. Lenstra. Factoring multivariate polynomials over finite fields. *Journal of Computer System Sciences*, 30(2) :235–248, 1985.

[LN96] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, October 1996.

[LO06] Kwankyu Lee and Michael E. O'Sullivan. Sudan's list decoding of Reed-Solomon codes from a Groebner basis perspective. `arxiv.org/abs/math.AC/0601022`, Jan 2006.

[LWCL95] Erl H. Lu, Hsiao P. Wuu, Y. C. Cheng, and P. C. Lu. Fast algorithms for decoding the (23,12) binary Golay code with four-error-correcting capability. *International Journal of Systems Science*, 26(4) :937–945, 1995.

[LY97] Philippe Loustaunau and Eric V. York. On the decoding of cyclic codes using Gröbner bases. *Applicable Algebra in Engineering, Communication and Computation*, 8(6) :469–483, 1997.

[Mas69] Jim Massey. Shift-register synthesis and BCH decoding. *IEEE Transactions on Information Theory*, 15(1) :122–127, 1969.

[Mat94a] Mitsuru Matsui. The first experimental cryptanalysis of the data encryption standard. In Y. G. Desmedt, editor, *Advances in Cryptology - CRYPTO '94*, number 839 in Lecture Notes in Computer Science, pages 1–11. Springer-Verlag, 1994.

[Mat94b] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93*, number 765 in Lecture Notes in Computer Science, pages 386–397, Berlin-Heidelberg-New York, 1994. Springer-Verlag.

[Mat01] Gretchen L. Matthews. Weierstrass pairs and minimum distance of Goppa codes. *Designs, Codes and Cryptography*, V22(2) :107–121, March 2001.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. Technical report, Jet Propulsion Lab Deep Space Network Progress report, 1978.

[Mer90] Ralph C. Merkle. One way hash functions and DES. In G. Brassard, editor, *Advances in cryptology—CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446. Springer, 1990.

[MM05] Seiichi Mita and Hajime Matsui. An effective error correction using a combination of algebraic geometric codes and parity codes for HDD. *IEEE Transactions on Magnetics*, 41(10) :2992–2994, 2005.

[MS88] James L. Massey and Thomas Schaub. Linear complexity in coding theory. In Gérard D. Cohen and Philippe Godlewski, editors, *Proceedings of the second Interpolation Colloquium on Coding Theory and Applications, Cachan-Paris, France, 1986*, volume 311 of *Lecture Notes in Computer Science*, pages 19–32. Springer, 1988.

[MS03] Teo Mora and Massimiliano Sala. On the Gröbner bases of some symmetric systems and their application to coding theory. *Journal of Symbolic Computation*, 35(2) :177–194, 2003.

[MTV04] Jun Ma, Peter Trifonov, and Alexander Vardy. Divide-and-conquer interpolation for list decoding of Seed-Solomon codes. In *Proceedings of the 2004 International Symposium on Information Theory*, pages 386–386, 2004.

[NH00] Rasmus R. Nielsen and Tom Høholdt. Decoding Reed-Solomon codes beyond half the minimum distance. In Johannes Buchmann, Tom Høholdt, Henning Stichtenoth, and Horacio Tapia-Recillas, editors, *Coding Theory, Cryptography and Related Areas : Proceedings of an International Conference on Coding Theory, Cryptography and Related Areas, Guanajuato, 1998*. Springer, 2000.

[NK93] Kaisa Nyberg and Lars R. Knudsen. Provable security against differential cryptanalysis. In E. F. Brickell, editor, *Advances in Cryptology - CRYPTO '92*, number 740 in Lecture Notes in Computer Science, pages 566–574, Berlin-Heidelberg-New York, 1993. Springer-Verlag.

[NK95] Kaisa Nyberg and Lars R. Knudsen. Provable security against a differential attack. *Journal of Cryptology*, V8(1) :27–37, December 1995.

[NLKW04] J. Nam, J. Lee, S. Kim, and D. Won. Ddh-based group key agreement for mobile computing. Cryptology ePrint Archive, 2004. `http://eprint.iacr.org/`.

[OF02] Henry O'Keeffe and Patrick Fitzpatrick. Gröbner basis solutions of constrained interpolation problems. *Linear Algebra and its Application*, 351/352 :533–551, 2002.

[OS99] Vadim Olshevsky and Amin M. Shokrollahi. A displacement approach to efficient decoding of algebraic-geometric codes. In *STOC '99 : Proceedings of the thirty-first annual ACM symposium on theory of computing, Atlanta, United States*, pages 235–244. ACM Press, 1999.

[OS03a] Vadim Olshevksy and Amin M. Shokrollahi. A displacement approach to decoding algebraic codes. *Contemporary Mathematics*, 323, May 2003.

[OS03b] Vadim Olshevsky and Amin M. Shokrollahi. Efficient list decoding of Reed-Solomon codes for message recovery in the presence of high noise levels. US PATENT 6,631,172, 2003.

[Pec01] Lancelot Pecquet. *Décodage en liste des codes géométriques.* PhD thesis, Université Pierre et Marie Curie, Paris 6, November 2001.

[PEKMV] Farzad Parvaresh, Mostafa El-Khamy, Robert J. McEliece, and Alexander Vardy. Algebraic list decoding of Reed-Solomon product codes. submitted to ISIT 2006.

[Per99] Adrian Perrig. Efficient collaborative key management protocols for secure autonomous group communication. In *International Workshop on Cryptographic Techniques and E-Commerce CrypTEC '99*, pages 192–202, 1999.

[PHB98] Vera S. Pless, Cary W. Huffman, and Richard A. Brualdi. An introduction to algebraic codes. In Vera S. Pless and Cary W. Huffman, editors, *Handbook of Coding Theory*, volume 1. Elsevier, 1998.

[PSvW91] R. Pellikaan, B. Z. Shen, and G. J. M. van Wee. Which linear codes are algebraic-geometric ? *IEEE Transactions on Information Theory,*, 37(3) :583–602, 1991.

[PV05a] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *FOCS 2005 : Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, Pittsburgh, United States*, pages 285–294, 2005.

[PV05b] Farzad Parvaresh and Alexander Vardy. Multivariate interpolation decoding beyond the Guruswami-Sudan radius. In *Allerton Conference on Communication, Control and Computing.* CDROM only, 2005. available from authors.

[PW04a] Ruud Pellikaan and Xin-Wen Wu. List decoding of q-ary Reed-Muller codes. preprint available form authors, 2004.

[PW04b] Ruud Pellikaan and Xin-Wen Wu. List decoding of q-ary Reed-Muller codes. *IEEE Transactions on Information Theory*, 50(4) :679–682, 2004.

[Roo83] Cornelis Roos. A new lower bound for the minimum distance of a cyclic code. *IEEE Transactions on Information Theory*, 29(3) :330–332, 1983.

[RR00] R. M. Roth and G. Ruckenstein. Efficient decoding of Reed-Solomon codes beyond half the minimum distance. *IEEE Transactions on Information Theory*, 46(1) :246–257, 2000.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, February 1978.

[RTCY92] Irving S. Reed, Trieu-Kien Truong, Xuemin Chen, and Xiaowei Yin. The algebraic decoding of the (41, 21, 9) quadratic residue code. *IEEE Transactions on Information Theory*, 38(3) :974–986, 1992.

[RYT90] Irving S. Reed, Xiaowei Yin, and Trieu-Kien Truong. Algebraic decoding of the (32, 16, 8) quadratic residue code. *IEEE Transactions on Information Theory*, 36(4) :876–880, 1990.

[RYTH90] Irving S. Reed, Xiaowei Yin, Trieu-Kien Truong, and J. K. Holmes. Decoding the (24,12,8) golay code. *Computers and Digital Techniques, IEE Proceedings-*, 137(3) :202–206, 1990.

[Sak90] Shojiro Sakata. Extension of the Berlekamp-Massey algorithm to n dimensions. *Information and Computation*, 84(2) :207–239, 1990.

[Sak01] Shojiro Sakata. On fast interpolation method for Guruswami-Sudan list decoding of one-point algebraic-geometry codes. In S. Bozta and I. Sphparlinski, editors, *Applied algebra, algebraic algorithms and error-correcting codes, AAECC-14, Melbourne, 2001*, volume 2227 of *Lecture Notes in Computer Science*, pages 172–181. Springer, 2001.

[Sal02] Massimiliano Sala. Groebner bases and distance of cyclic codes. *Applied Algebra in Engineering, Communications and Computing*, 13(2) :137–162, 2002.

[Sal03] Massimiliano Sala. Upper bounds on the dual distance of BCH(255, k). *Designs Codes and Cryptography*, 30(2) :159–168, September 2003.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4) :701–717, October 1980.

[Sch95] Bruce Schneier. *Applied Cryptography : Protocols, Algorithms, and Source Code in C*. Wiley, 2 edition, October 1995.

[SNF00] S. Sakata, Y. Numakami, and M. Fujisawa. A fast interpolation method for list decoding of RS and algebraic-geometric codes. In *Proceedings of the IEEE International Symposium on Information Theory, Sorento, Italy*, page 479, 2000.

[Sti93] Henning Stichtenoth. *Algebraic Function Fields and Codes*. Springer, 1993.

[STW96] Michael Steiner, Gene Tsudik, and Michael Waidner. Diffie-Hellman key distribution extended to group communication. In *CCS '96 : Proceedings of the 3rd ACM conference on Computer and communications security*, pages 31–37, New York, NY, USA, 1996. ACM Press.

[STW00] M. Steiner, G. Tsudik, and M. Waidner. Key agreement in dynamic peer groups. *Parallel and Distributed Systems, IEEE Transactions on*, 11(8) :769–780, 2000.

[Sud97] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1) :180–193, March 1997.

[SW99] Mohammad A. Shokrollahi and Hal Wasserman. List decoding of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45(2) :432–437, 1999.

[Tav04] Cédric Tavernier. *Testeurs, problèmes de reconstruction univariés et multivariés, et application à la cryptanalyse du DES*. PhD thesis, École Polytechnique, January 2004.

[TCCL05] Trieu-Kien Truong, Yaotsu Chang, Yan-Haw Chen, and C. D. Lee. Algebraic decoding of (103, 52, 19) and (113, 57, 15) quadratic residue codes. *IEEE Transactions on Communications*, 53(5) :749–754, 2005.

[TVZ82] Michael A. Tsfasman, Serguei G. Vlăduţ, and Th. Zink. Modular curves, Shimura curves, and Goppa codes, better than Varshamov-Gilbert bound. *Math. Nachr.*, 109 :21–28, 1982.

[UUC+05] UVIGO, UNIGE, CNIT, GAUSS, and CNRS. First summary report on fundamentals. Technical report, ECRYPT Network of Excellence (NoE), March 2005.

[Val95] Annick Valibouze. Modules de cauchy, polynômes caractéristiques et résolvantes. Technical Report 1995-131, LIAFA, 1995.

[Var97] Alexander Vardy. Algorithmic complexity in coding theory and the minimum distance problem. In *STOC '97 : Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, El Paso, United States*, pages 92–109. ACM Press, 1997.

[VK00] Alexander Vardy and Ralf Kötter. Algebraic soft-decision decoding of Reed-Solomon codes. `http://www.comm.csl.uiuc.edu/~koetter/publications/RSsoft.ps`, May 2000.

[Wag02] David Wagner. A generalized birthday problem. In *Advances in cryptology—CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002.

[WB86] Loyd R. Welch and Elwyn R. Berlekamp. Error correction for algebraic block codes. US Patent 4 633 470, 1986.

[WFLY04] Xiaoyun Wang, Dengguo Feng, Xuejia Lai, and Hongbo Yu. Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. Cryptology ePrint Archive, 2004. `http://eprint.iacr.org/`.

[WLF+05] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the hash functions MD4 and RIPEMD. In Ronald Cramer, editor, *Advances in Cryptology—EUROCRYPT 2005, Aarhus, Denmark*, volume 3494 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2005.

[Woz58] J. M. Wozencraft. List decoding. Technical report, MIT, Cambridge, MA, 1958.

[WS01] Xin-Wen Wu and P. H. Siegel. Efficient root-finding algorithm with application to list decoding of algebraic-geometric codes. *IEEE Transactions on Information Theory*, 47(6) :2579–2587, 2001.

[WY05] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *Advances in Cryptology, EUROCRYPT 2005, Aarhus, Denmark*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.

[Xin01] Chaoping Xing. Algebraic-geometry codes with asymptotic parameters better than the Gilbert-Varshamov and the Tsfasman-Vlăduţ-Zink bounds. *IEEE Transactions on Information Theory*, 47(1) :347–352, 2001.

[Xin05] Chaoping Xing. Goppa geometric codes achieving the Gilbert-Varshamov bound. *IEEE Transactions on Information Theory*, 51(1) :259–264, January 2005.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *Symbolic and Algebraic Computation : EUROSM '79, An International Symposium on Symbolic and Algebraic Manipulation, Marseille, France*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.