Polynômes de permutation et applications en cryptographie - Cryptanalyse de registres combinés

Yann Laigle-Chapuy

Thèse de doctorat de l'université Pierre et Marie Curie

Spécialité Informatique

edite de Paris

présentée par

Yann Laigle-Chapuy

pour obtenir le grade de

Docteur de l'université Pierre et Marie Curie

Polynômes de permutation et applications en cryptographie

Cryptanalyse de registres combinés

soutenue le 19 juin 2009

devant le jury composé de

Rapporteurs

Jean-Marc Couveignes	Université de Toulouse II, France
Pierre Loidreau	Université de Rennes I, France

Directrice de thèse

Pascale Charpin	INRIA Paris-Rocquencourt, France

Éxaminateurs

Anne Canteaut	INRIA Paris-Rocquencourt, France
Gohar Kyureghyan	Universität Magdeburg, Allemagne
Gary McGuire	University College Dublin, Irlande
Annick Valibouze	Université Pierre et Marie Curie, France

Yann Laigle-Chapuy

Polynômes de permutation et applications en cryptographie

Cryptanalyse de registres combinés

INRIA-équipe projet SECRET
Domaine de Voluceau
78153 Le Chesnay

Remerciements

Cette thèse a été effectuée au projet CODES de l'INRIA Rocquencourt, puis dans l'équipe SECRET et enfin dans l'équipe projet SECRET de l'INRIA Paris-Rocquencourt. Elle n'aurait certainement pas été possible sans l'aide de nombreuses personnes que je vais tenter ici de remercier.

Pour commencer, je tiens à remercier mon professeur de mathématiques de classe préparatoire qui a certainement contribué à me motiver en m'annonçant "faites plus de physique, il n'y a pas que l'informatique quand même". Plus sérieusement, je remercie Daniel Augot et Jean-Pierre Tillich qui ont été mes premiers contacts dans le projet lorsque je les ai eu en cours.

Vient ensuite tout naturellement Pascale Charpin, qui après avoir encadré mon stage est devenue ma directrice de thèse. Elle a su me faire partager son expérience, et sa bienveillance ainsi que sa patience m'ont beaucoup aidé.

Je tiens aussi bien sûr à remercier Jean-Marc Couveignes et Pierre Loidreau pour l'honneur qu'ils m'ont fait d'accepter d'être rapporteurs pour cette thèse, pour la lecture attentive qu'ils en ont fait, ainsi que pour les commentaires qui ont permis de l'améliorer.

Je remercie aussi tous les autres membres de mon jury : Anne Canteaut, Gohar Kyureghyan, Gary McGuire et Annick Valibouze.

Je remercie Anne Canteaut pour les nombreuses discussions que nous avons pu avoir aussi bien sur les serpents des neiges que sur les otaries.

Je remercie Gohar Kyureghyan d'avoir fait le déplacement depuis l'Allemagne pour assister à cette soutenance. Sa présence ici est un réel plaisir.

Je remercie Gary McGuire d'avoir trouvé le temps, malgré la lourde charge que constitue l'organisation de $\mathbb{F}_q 9$, de me l'honneur de faire partie de mon jury.

Je remercie Annick Valibouze pour l'intérêt porté à mon travail et les nombreux commentaires sur le manuscrit.

Je tiens à remercier tous les gens que j'ai pu rencontrer au projet durant ces années. Tout d'abord les membres permanents, Daniel Augot, Anne Canteaut, Pascale Charpin, Jean-Pierre Tillich et Nicolas Sendrier pour l'ambiance générale qui règne au bâtiment 25 et pour leur disponibilité. Un grand merci aussi à tous les membres extérieurs, thésards ou stagiaires que j'ai pu côtoyer : Alexander, Grisha, Christelle, Claude, Françoise, Pierre, Ludovic, Bassem, Racem, Caroline, Ayoub, Fabien, Marine, Carmen, Matthieu, Harold, Raghav, Cédric, Scarab, Fred, Thomas, Manu, Andrea, Stéphane, Christophe, Michael, Maria, Domitille, Marion, Raphael, Iryna, Bhaskar, Avishek, Maxime, Céline, Benoît, Sumanta, Deepak, Vincent, Denise, Stéphane, Anne. Merci aussi à Jean-Charles Faugères et aux gens du LIP6 de m'avoir accueilli pendant un an. Merci Mohab et Valérie d'avoir partagé votre bureau avec moi, merci Ludovic, Guénaël, et merci Luk pour la relève. Enfin pour le monde de la cryptographie, merci aussi à toutes les personnes avec qui j'ai

pu lier amitié lors de colloques où conférences. Un grand merci aussi à tous ceux que je viens d'oublier de me pardonner, il est 2h du matin... Si vous m'envoyez un mail, je corrigerai dans la version électronique.

Parmi tous ces gens, certains méritent une attention particulière. Merci à Mathieu et Geneviève pour vos invitations, merci Fred (ou pas) pour m'avoir fait découvrir KadoKado et le contrat du CELAR, merci Scarab de m'avoir laissé ton bureau, merci Christelle pour les ragots, merci à Matthieu pour cet amphi, merci à Louisy Joseph de nous avoir donné des nouvelles, merci à Pierre-Louis et Cédric pour le cadeau d'anniversaire, merci à Daniel pour le barbecue, merci à Maria pour les gâteaux.

Je remercie aussi tous les gens qui m'ont permis d'oublier le travail de temps en temps (et oui ça m'arrive). Merci à Luc pour toutes ces soirées et journées passées ensemble, merci a toute l'équipe du *Snook'n pool* Nico, Nak, Emile, Greg et Ben en particulier, que j'ai encore plaisir à retrouver. Merci à l'AVE même si je vous ai un peu perdu de vue. Merci à Benoît et Anne pour l'hospitalité. Un grand merci à tous les Pumas, Joana, Béa, Chris, Sylvain, Lore, Gaëtan, Marcio et Axel en tête, et à tous ceux qui rêvent de le devenir. J'ai réservé une bouteille de champagne pour vous aujourd'hui.

Merci enfin à ma famille. Ils arrivent en dernier dans cette liste mais c'est bien sur une place d'honneur. Leur soutien au cours de ces années a toujours été essentiel. Merci d'être ici aujourd'hui encore pour assister à cet exposé malgré les formules soporifiques. Ce paragraphe est court mais je suis certain que vous savez à quel point votre présence me fait plaisir. Enfin merci Andrea, tu as gagné une place avec eux.

Notice

$\mathbb{F}_q$	le corps à q élément
$\mathbb{F}_q\star$	les éléments non nuls de $\mathbb{F}_q$
$\mathbb{K}[X]$	l'anneau des polynômes à une indéterminée X à coefficients dans $\mathbb{K}$
$\mathcal{P}oly\mathcal{P}erm$	l'ensemble des polynômes de permutation
Tr_ℓ^k	l'application trace, de $\mathbb{F}_{2^k}$ dans $\mathbb{F}_{2^\ell}$
$\mathcal{S}_n$	le groupe cyclique d'ordre n
$\deg(P)$	le degré du polynôme P
$\mathbb{P}$	l'ensemble des nombres premiers
$\mathcal{M}_{n,k}(A)$	l'ensemble des matrices $n \times k$ à coefficients dans l'anneau A
ℓog_g	le logarithme discret dans un groupe de générateur g
$\mathcal{F}[F]$	la transformée de Walsh de la fonction booléenne F
$\mathcal{N}(F)$	la nonlinéarité de la fonction booléenne F

Aperçu de la thèse

Cette thèse est le fruit de mes quatre années à l'INRIA, de septembre 2004 à décembre 2008. Les résultats qui sont présentés ont fait l'objet de plusieurs publications dans des revues ou dans des actes de conférences internationales :

[LC07a] Yann Laigle-Chapuy, A note on a class of quadratic permutations *Applied Algebra, Algebraic Algorithms, and Error Correcting Codes, AAECC17*, LNCS 4851, pp. 130-137, décembre 2007 ;

[CDLCT07] Anne Canteaut, Frédéric Didier, Yann Laigle-Chapuy et Jean-Pierre Tillich, Veille technologique dans le domaine du décodage itératif, *Contrat de recherche 06.42.113 avec le CELAR*, novembre 2007 ;

[DLC07] Frédéric Didier et Yann Laigle-Chapuy, Finding low-weight polynomial multiples using discrete logarithm, *IEEE International Symposium on Information Theory, ISIT 07*, juin 2007 ;

[LC07b] Yann Laigle-Chapuy, Permutation Polynomials and applications to coding theory, *Finite fields and their applications*, vol. 13, Issue 1, pp.58-70, janvier 2007 (accepté en 2005) ;

[BCCLC06] Thierry P. Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy, On Almost Perfect Nonlinear mappings over $\mathbf{F}_2^n$, *IEEE Trans. Inform. Theory*, vol. 52, n. 9, pp.4160-70, septembre 2006 ;

[BCCLC05] Thierry P. Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy, On Almost Perfect Nonlinear mappings, *IEEE International Symposium on Information Theory, ISIT 05*, septembre 2005.

On peut les regrouper en deux thématiques correspondant aux deux parties de ce document, à savoir les polynômes de permutation et l'étude des registres combinés. Ces deux thèmes de recherche peuvent a priori sembler éloignés mais ils s'inscrivent tous deux au sein d'une recherche axée sur la cryptographie symétrique. Le document est organisé de la manière suivante.

La première partie sera consacrée aux polynômes de permutation. Le chapitre 1 introduira les diverses notations et présentera les premières propriétés. Nous ferons ensuite dans le chapitre 2 une étude détaillée des différentes méthodes existantes pour tester si un polynôme induit une permutation d'un corps fini.

Le chapitre 3 dressera une liste de différentes classes de polynômes de permutation classées selon leur complexité croissante au regard de différents critères. Nous commencerons par les polynômes de degré très faible et les polynômes linéaires. Nous nous

intéresserons tout spécialement aux polynômes du type $X^r P\left(X^{(q-1)/m}\right)$ ainsi qu'au cas des binômes. En particulier nous verrons dans ce chapitre quelques résultats du début de ma thèse correspondant à l'article [LC07b]. La section 3.3 étudiera les polynômes de degré algébrique 2 (quadratiques) et présentera les résultats de l'article présenté à *AAECC-17* [LC07a]. Enfin la répartition des polynômes de permutation sera l'objet du chapitre 4 avec là aussi des résultats provenant de l'article [LC07b].

Nous conclurons cette première partie avec une liste d'applications possibles de ces résultats en cryptographie, et en particulier en cryptographie symétrique. On trouvera entre autre les résultats de l'article [BCCLC06].

La deuxième partie sera quant à elle consacrée à l'étude des registres combinés. Il s'agit d'une des construction les plus simple de chiffrement à flots.

Nous commencerons pour cela dans le chapitre 6 par donner quelques rappels sur les récurrences linéaires sur les corps finis. Nous serons alors prêts à introduire dans le chapitre 7 le modèle de chiffrement à flot que l'on considérera. Un bref état de l'art des attaques existantes sera aussi présenté. Enfin le chapitre 8 sera une présentation détaillée de notre attaque sur les registres combinés, présente dans le rapport de recherche [CDLCT07].

Enfin dans le chapitre 9, nous présenterons les différentes méthodes pour calculer les multiples de petit poids d'un polynôme. Ce problème est en effet un problème récurrent en cryptanalyse et constitue la première étape des attaques présentées. Nous verrons en particulier une méthode présentée dans [DLC07] qui est spécialement adaptée à ce cas.

Overview

This document is the result of my four years spent at INRIA, from september 2004 to december 2008. The results presented here were also published in different journals or conference proceedings :

[LC07a] Yann Laigle-Chapuy, A note on a class of quadratic permutations *Applied Algebra, Algebraic Algorithms, and Error Correcting Codes, AAECC17*, LNCS 4851, pp. 130-137, december 2007 ;

[CDLCT07] Anne Canteaut, Frédéric Didier, Yann Laigle-Chapuy et Jean-Pierre Tillich, Veille technologique dans le domaine du décodage itératif, *Contrat de recherche 06.42.113 avec le CELAR*, november 2007 ;

[DLC07] Frédéric Didier et Yann Laigle-Chapuy, Finding low-weight polynomial multiples using discrete logarithm, *IEEE International Symposium on Information Theory, ISIT 07*, june 2007 ;

[LC07b] Yann Laigle-Chapuy, Permutation Polynomials and applications to coding theory, *Finite fields and their applications*, vol. 13, Issue 1, pp.58-70, january 2007 (accepted en 2005) ;

[BCCLC06] Thierry P. Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy, On Almost Perfect Nonlinear mappings over $\mathbf{F}_2^n$, *IEEE Trans. Inform. Theory*, vol. 52, n. 9, pp.4160-70, september 2006 ;

[BCCLC05] Thierry P. Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy, On Almost Perfect Nonlinear mappings, *IEEE International Symposium on Information Theory, ISIT 05*, september 2005.

We can divide this work between the two following themes : permutation polynomials, and the study of the combination generator. It might seem at first sight that those are two completely different topics however they in fact fit in a broader interest for symmetric cryptography answering each to different problems, either in the design of new systems or in the cryptanalisis of classical ones. The document is organised as follows.

The first part deals with permutation polynomials. The Chapter 1 will introduce some notations and some basic properties. We will then, in Chapter 2, survey the existing techniques to test wether a given polynomial induces a permutation of a finite field.

The Chapter 3 is constructed as a list of classes of parmutation polynomials ordered with respect to different criteria. First, we will consider polynomials of small total degree, then linearized polynomials. We will then focus on spars polynomials of the form

$X^r P\left(X^{(q-1)/m}\right)$ and especially binomials. Some of the results from [LC07b] will be presented in this chapter. In Section 3.3 we will study polynomials of algebraic degree 2 and detail the results from [LC07a]. Finally, the distribution of permutation polynomials will be the main topic of Chapter 4 with in particular some results based on [LC07b].

Some possible applications in different areas of cryptography will then conclude this part, with some contributions for symmetric cryptography contained in [BCCLC06].

The second part is devoted to the study of the combination generator, one of the simplest and most studied stream cipher.

Chapitre 6 contains the necessary basic results on linear reccurences on finite fields and LFSRs. The presentation of the combination generator itself occurs in the Chapter 7, followed by a short state of the art of existing attacks. Our new attack on the combination generator is the core of Chapitre 8 and corresponds to [CDLCT07].

The remaining of this part deals with the problem of finding low weight multiples of a given polynomial. This is an important problem with many applications in cryptanalysis and we will also present our result from [DLC07] which is especially well suited for attacking the combination generator.

Première partie

Les polynômes de permutation

L'étude systématique des polynômes de permutation est un sujet datant du XIX^{e}siècle. Elle a été initiée par Hermite [Her63] en 1863 en ce qui concerne les corps premiers. Par la suite, les premiers résultats concernant les corps finis généraux ont été énoncés par Dickson [Dic97]. Il s'agit d'un vaste domaine qui a connu une nouvelle vague d'engouement il y a vingt ans lorsque des propositions de cryptosystèmes basés sur ces polynômes ont émergées (voir [LM84b, LM84a, Lid85, LM88, LM93, MN81, Pie93]). De nombreuses personnes ont travaillé sur ce sujet : Carlitz, Dickson, von zur Gathen, Dobbertin, Shparlinski, Lidl, Niederreiter, Cohen... Pourtant, il reste encore beaucoup de questions ouvertes. En particulier, on dispose d'assez peu de classes infinies explicites de polynômes de permutation.

Une liste de neuf problèmes ouverts concernants les polynômes de permutation avait été donnée par Lidl et Mullen dans [LM88], puis mise à jour et étendue à 17 dans [LM93]. Nous reprenons ici cette liste en guise d'introduction. Les conjectures ne sont pas détaillées ici mais nous les verrons en détail par la suite.

P1 *Trouver de bons algorithmes pour tester si un polynôme est de permutation.* Ce problème est pratiquement résolu. On connait des algorithmes de complexité en temps polynomiale aussi bien probabilistes que déterministes. Cela sera détaillé au chapitre 2.

P2 *Trouver de nouvelles classes de polynômes de permutation.* Les résultats sont en constante évolution et plusieurs contributions font parties de cette thèse [LC07b, LC07a]. Nous essaierons de donner une liste de ce qui est connu dans le chapitre 3.

P3 *Trouver de nouvelles classes de polynômes de permutation avec une utilité cryptographique.* La formulation de ce problème est assez vague et peut se scinder en deux : trouver de nouvelles applications, et trouver de nouvelles classes pour ces applications. Dans une certaine mesure, il s'agit du sujet de la première partie de cette thèse.

P4 *Énoncer des conditions nécessaires et suffisantes pour que les polynômes de Dickson de deuxième espèce soient de permutation.* Des progrès ont été faits, mais la classification n'est pas encore complète [Coh94, HM95, HM98, Mat82].

P5 *Étendre la liste de Dickson des polynômes de permutation normalisés 3.1 jusqu'à un degré plus élevé.*

P6 *Compter le nombre de polynômes de permutation de degré d fixé.* Ce problème, ainsi que d'autre problèmes de dénombrement, sera traité dans le chapitre 4.

P7 *Établir une conjecture de Dickson concernant les polynômes de la forme* $x(x^d - \alpha)^{(p-1)/d}$ *avec p premier impair.*

P8 *Établir la conjecture de Chowla et Zassenhaus.* Ce résultat a été établi par Cohen [Coh90]. Nous en reparlerons au chapitre 4.

P9 *Établir la conjecture de Carlitz.* Ce résultat découle d'un résultat plus profond de Fried, Guralnick et Saxl [FGS93], on pourra aussi consulter [Wan87, CF95].

P10 *Établir la conjecture de Dembowski-Ostrom.* Coulter a trouvé un contre exemple à cette conjecture [CM97].

P11 Caractériser les paires de polynômes de permutation correspondant aux automorphismes du design combinatoire $D(q, k)$.

P12 *Prouver la conjecture de Evans, Greene et Niederreiter [EGN92] concernant les polynômes de permutation complets.* La définition de cette conjecture sera donnée dans la section 4.3.

P13 *Déterminer les conditions pour qu'un binôme soit de permutation.* Beaucoup de résultats dans cette direction ont été apportés. Nous consacrerons une partie du chapitre 3 à ce problème.

P14 *Déterminer les conditions pour qu'un trinôme soit de permutation.*

P15 *Caractériser les polynômes de permutation de la forme* $1 + x + \cdots + x^k$ *sur les corps de caractéristique deux.*

P16 *Déterminer sur les corps de caractéristique deux tous les polynômes de permutation* f *tels que* $(f(x+a) - f(a))/x$ *soit de permutation et* $f(0) = 0$. Pour connaître l'état de l'art sur ce sujet on consultera le site maintenu par Bill Cherowitzo [Che]. Nous en reparlerons aussi dans la section 3.7.

P17 Enfin le dernier problème est en lien avec les polynômes de permutation locaux, c'est-à-dire des polynômes à n variables tels que toute évaluation de $n-1$ variables donne un polynôme univarié de permutation [Mul80a, Mul80b, DHK04].

La majeure partie de cette thèse est consacrée à ce sujet et aux applications possibles en cryptographie. Il s'agit donc principalement des problèmes P2, P3 et P13.

Le chapitre 1 est une introduction présentant les premières définitions et propriétés que nous utiliserons ensuite. Nous essaierons aussi de donner une liste de références auquelles se reporter. Nous y verrons aussi le lien entre les polynômes de permutation et les polynômes exceptionnels. Ce lien permet en particulier d'utiliser des théorèmes de géométrie algébrique pour obtenir des propriétés asymptotiques. Il est aussi à la base de l'algorithme polynomial déterministe permettant de tester si un polynôme induit une permutation sur un corps fini.

Le chapitre 2 est consacré à ce problème décisionnel. Nous verrons les différentes approches possibles, probabilistes ou déterministes, ainsi que des critères simples permettant de faire un tri préalable. Nous comparerons aussi l'efficacité des différentes méthodes sur des exemples.

Nous établirons dans le chapitre 3 une liste de classes de polynômes de permutation. Nous porterons un intérêt tout particulier au cas de la caractéristique 2. Les principaux critères de classification seront le degré, le degré algébrique et le nombre de termes.

La répartition des polynômes de permutation parmi l'ensemble des polynômes sera traitée dans le chapitre 4. Nous verrons en particulier comment évaluer le nombre de polynômes de forme prescrite, c'est-à-dire en fixant les exposants pouvant apparaître.

Enfin, nous verrons une série d'applications possibles de nos résultats dans les domaines de la cryptographie dans le chapitre 5. En cryptographie à clé secrète, on utilise principalement la nature combinatoire de ces objets afin de construire des fonctions aux propriétés optimales, telles que les fonctions courbes ou APN. En cryptographie à clé publique, on cherche plutôt à définir des permutations lacunaires de corps suffisamment grands.

Chapitre 1

Définitions et premières propriétés

Avant toute chose précisons l'objet de notre étude. Dans la suite, nous noterons $\mathbb{F}_q$ le corps fini à q éléments, et nous allons nous placer dans l'anneau des polynômes à une indéterminée X et à coefficients dans ce corps que nous noterons $\mathbb{F}_q[X]$.

Pour une introduction aux corps finis, le lecteur pourra se reporter au livre de Robert J. McEliece [McE87]. Pour une étude plus poussée on pourra consulter celui de Rudolf Lidl et Harald Niederreiter [LN97].

1.1 Les polynômes de permutation

Définition 1.1. Un polynôme $P \in \mathbb{F}_q[X]$ est dit polynôme de permutation de $\mathbb{F}_q$ si et seulement si la fonction associée

$$\left\{ \begin{array}{ccc} \mathbb{F}_q & \to & \mathbb{F}_q \\ x & \mapsto & P(x) \end{array} \right.$$

est une permutation, c'est-à-dire si elle est bijective.

Remarque 1.2. Lorsque le contexte est clair, nous dirons seulement que P est un polynôme de permutation, sans préciser le corps de base.

Les deux exemples les plus simples de polynômes de permutation sont les suivants :

Proposition 1.3.
- *Pour tout* $(a, b) \in \mathbb{F}_q^\star \times \mathbb{F}_q$, *le polynôme* $aX + b$ *est un polynôme de permutation.*
- X^k *est un polynôme de permutation de* $\mathbb{F}_q$ *si et seulement si* $\operatorname{pgcd}(k, q-1) = 1$.

Preuve :
- $ax + b = y \Leftrightarrow x = (y - b)/a$, la fonction est donc bien bijective.
- 0 a pour antécédent unique 0. $\mathbb{F}_q^\star$ est un groupe cyclique d'ordre $q - 1$. Soit α un générateur ; le groupe engendré par α^k est d'ordre $(q-1)/\operatorname{pgcd}(k, q-1)$ d'où la propriété.

□

À toute application f de $\mathbb{F}_q$ dans lui-même, on peut associer un unique polynôme de $\mathbb{F}_q[X]$ de degré inférieur strictement à q, P_f (en utilisant les polynômes d'interpolation de Lagrange par exemple). Ceci est donc aussi vrai pour les permutations. On pourra donc se limiter aux polynômes de degré inférieur à q. Plus précisément, en considérant $\mathbb{F}_q$ comme l'ensemble des zéros du polynôme $X^q - X$, on remarque que l'on peut se contenter de travailler modulo ce polynôme.

La notion de groupe des permutations se traduit alors naturellement dans le cadre des polynômes. En effet, à toute permutation $\sigma \in \mathcal{S}_q$ on peut associer un unique polynôme P_σ. De plus, deux polynômes prennent les mêmes valeurs sur $\mathbb{F}_q$ si et seulement si ils sont égaux modulo $X^q - X$, on a donc

$$P(\sigma_1 \circ \sigma_2) \equiv P(\sigma_1) \circ P(\sigma_2) \pmod{X^q - X}.$$

On peut donc énoncer le résultat suivant :

Proposition 1.4. *L'ensemble des polynômes de permutation modulo* $X^q - X$ *muni de la composition est un groupe isomorphe au groupe des permutations* $\mathcal{S}_q$.

Une des difficultés provient du fait qu'une permutation simple correspond souvent à un polynôme relativement compliqué. Ainsi, la transposition (01) dans le corps à sept éléments $\mathbb{F}_7$ à pour polynôme associé :

$$X^5 + X^4 + X^3 + X^2 + 2X + 1.$$

C'est aussi un des grands intérêts de cette formulation car réciproquement, des polynômes très simples conduisent à des permutations complexes, et cela nous donne une manière efficace de les implémenter.

De même que le groupe des permutations est engendré par un cycle et une transposition adaptés, on peut exhiber différentes familles simples de générateurs du groupe des polynômes de permutation.

1.1.1 Générateurs du groupe des polynômes de permutation

Proposition 1.5. *Soit α un élément primitif de $\mathbb{F}_q^\star$. Le groupe des polynômes de permutation est engendré par $\{\alpha X, X + 1, X^{q-2}\}$.*

On obtient ce résultat en considérant la forme générale du polynôme correspondant à la transposition $(0a)$, pour a non nul :

$$f_a(X) = -a^2 \left[\left((X-a)^{q-2} + a^{q-2} \right)^{q-2} - a \right]^{q-2}.$$

Encore une fois, l'intérêt de ce résultat est la simplicité de la forme polynomiale de ces générateurs.

Pour plus de références sur ce sujet, on se référera au livre de Lidl et Niederreiter [LN97]. Un article récent de Michael Zieve sur le sujet encore à paraître est aussi disponible [Ziea].

1.1.2 Polynômes linéarisés

Une autre famille importante de polynômes que nous reverrons plus tard est la suivante.

Définition 1.6. Soit $\mathbb{F}_{q^r}$ une extension de $\mathbb{F}_q$ de degré r. On appelle polynômes linéarisés les polynômes de la forme

$$\sum_{i=0}^{r-1} a_i X^{q^i} \in \mathbb{F}_{q^r}[X].$$

Ces polynômes correspondent aux applications $\mathbb{F}_q$-linéaires. Le plus souvent, on prendra comme corps de base le corps premier.

Pour beaucoup d'applications cryptographiques, les propriétés seront conservées par composition avec une application linéaire et l'on sera amené à considérer la relation d'équivalence suivante.

Définition 1.7. Deux polynômes P et Q sont dits linéairement équivalents si et seulement si il existe L_1 et L_2 des polynômes linéarisés tels que

$$L_1 \circ P \circ L_2 = Q.$$

Lorsque P et Q sont des polynômes de permutation, il est clair que cela implique que L_1 et L_2 le sont aussi, sinon la composée $L_1 \circ P \circ L_2$ n'est pas bijective. On a donc la propriété suivante.

Proposition 1.8. *Deux polynômes de permutation P et Q sont linéairement équivalents si et seulement si il existe L_1 et L_2 des polynômes linéarisés de permutation tels que*

$$L_1 \circ P \circ L_2 = Q.$$

1.2 Les polynômes exceptionnels

Nous allons voir ici comment relier la notion de polynômes de permutation avec une notion beaucoup plus géométrique, à savoir celle de polynômes exceptionnels. Ce lien étroit nous permettra en particulier d'utiliser des résultats classiques pour évaluer le nombre de points sur une courbe algébrique.

Nous allons maintenant considérer des polynômes à deux indéterminées.

Définition 1.9. Soit $\mathbb{K}$ un corps. Un polynôme de $\mathbb{K}[X, Y]$ est dit absolument irréductible s'il est irréductible sur toute extension algébrique de $\mathbb{K}$.

Par exemple, le polynôme $(Y^2 + X^2) \in \mathbb{F}_7[X, Y]$ est irréductible sur $\mathbb{F}_7$ mais n'est pas absolument irréductible car il se décompose en $(Y + \alpha X)(Y - \alpha X)$ sur $\mathbb{F}_{7^2} = \mathbb{F}_7[\alpha]/ < \alpha^2 + 1 >$.

Définition 1.10. Un polynôme $P \in \mathbb{F}_q[X]$ est dit exceptionnel sur $\mathbb{F}_q$ si aucun des facteurs irréductibles de

$$\Phi(X, Y) = \frac{P(X) - P(Y)}{X - Y}$$

n'est absolument irréductible.

On peut alors établir le lien entre polynômes de permutation et polynômes exceptionnels de la manière suivante.

Considérons la courbe $C_\Phi = \{(a, b) \in \mathbb{F}_q \times \mathbb{F}_q | \Phi(a, b) = 0\}$. P est un polynôme de permutation si et seulement si C_Φ n'admet aucun point rationnel en dehors de la ligne $x = y$.

On peut alors utiliser les résultats de Weil pour estimer le nombre de points de cette courbe (cf. [LN97]) et obtenir par exemple le théorème 1.12 suivant.

Citons tout d'abord une conjecture énoncée par Carlitz et généralisée et prouvée par la suite par Lenstra et Wan de la manière suivante.

Théorème 1.11 (Carlitz-Lenstra-Wan)**.** *Soit $\mathbb{F}_q$ un corps et k un entier naturel tel que* $\mathrm{pgcd}(k, q - 1) > 1$. *Alors il n'existe pas de polynôme exceptionnel de degré k sur $\mathbb{F}_q$.*

D'après la propriété vue en introduction, cela implique que tout polynôme de degré k ne peut être de permutation que sur un nombre fini d'extensions de $\mathbb{F}_q$.

Théorème 1.12. *Pour tout entier k, il existe une constante c_k telle que pour tout corps $\mathbb{F}_q$ avec* $\mathrm{pgcd}(k, q - 1) > 1$ *et $q > c_k$, il n'existe pas de polynôme de permutation sur $\mathbb{F}_q$ de degré k.*

En notant $q = p^n$, le cas où p ne divise pas n a été traité dès 1967 par Hayes [Hay67]. Par la suite, divers résultats partiels ont été démontrés par Wan [Wan87] et Cohen [Coh91]. La fin de la démonstration est due à Fried, Guralnick et Saxl [FGS93] (voir aussi [CF95]).

Définition 1.13. Un polynôme $P \in \mathbb{F}_q[X]$ est dit séparable si son polynôme dérivé P' est non nul.

Théorème 1.14 (von zur Gathen [vzG91b])**.** *Soient $P \in \mathbb{F}_q[X]$ un polynôme séparable de degré n, σ le nombre de facteurs absolument irréductibles de $\Phi(X,Y) = \frac{P(X)-P(Y)}{X-Y}$, $V(P)$ le cardinal de $\{P(a); a \in \mathbb{F}_q\}$, $\rho = q - V(P)$ et $0 < \varepsilon \leqslant 8$.*

(i) Si $q \geqslant n^4$ alors P est un polynôme de permutation si et seulement si il est exceptionnel.

(ii) Si $q \geqslant \varepsilon^{-2} n^4$ alors $\rho > (\sigma - \varepsilon)q/n$.

On peut par exemple caractériser par ce biais les degrés possibles pour les polynômes de permutation.

Théorème 1.15. *Soient n et q deux entiers tels que $q \geqslant n^4$ et* $\operatorname{pgcd}(n,q) = 1$*. Alors il existe un polynôme de permutation de $\mathbb{F}_q$ de degré n si et seulement si* $\operatorname{pgcd}(n, q-1) = 1$*.*

Pour plus de détails, se reporter au livre de Niederreiter et Lidl [LN97], ainsi qu'à l'article de von zur Gathen [vzG91b].

La classification des polynômes exceptionnels est maintenant très avancée. Le résultat primordial dans ce domaine a été celui de Fried, Guralnick et Saxl [FGS93] qui ont énoncé les groupes de monodromie possibles pour de tels polynômes. Nous n'exposerons pas ici les détails de ces théorèmes qui nous amèneraient trop loin, mais nous nous contenterons de dire que depuis de nombreux travaux ont apporté des exemples pour chacun des cas listés dans ce théorème.

Plus précisément, la majorité des cas sont parfaitement connus dans le sens où il a été établi une classification complète des classes d'équivalence possibles pour les polynômes exceptionnels. On pourra en particulier consulter les nombreux travaux de Michael Zieve concernant ce sujet [LJZ96, GZ, GRZ].

Les différentes classes de polynômes exceptionnels seront l'objet une des section du chapitre 3 consacré aux familles de polynômes de permutation.

Chapitre 2

Le problème décisionnel

Afin de pouvoir étudier ces polynômes, il est important de disposer de moyens théoriques aussi bien qu'algorithmiques pour tester si un polynôme est de permutation. On notera $\mathcal{P}oly\mathcal{P}erm$ l'ensemble des polynômes de permutation. Les complexités que nous donnons ici seront comptées en opérations élémentaires dans le corps de base (multiplication, addition, inversion).

Nous verrons qu'il est assez facile d'obtenir des algorithmes probabilistes efficaces. On peut même arriver à tester si un polynôme est de permutation en un nombre quasi-linéaire d'opérations dans le corps de base grâce à une méthode proposée par von zur Gathen en 1991 [vzG91a].

La cas déterministe est cependant beaucoup plus compliqué. En 1988 Lidl et Mullen [LM88] posaient le problème de trouver un algorithme déterministe de complexité inférieure à $\mathcal{O}(nq)$ (où n est le degré). Shparlinski fut un des premier a énoncer une solution en utilisant le lien avec les polynômes exceptionnels [Shp92], mais la complexité obtenue était toujours exponentielle. Par la suite trouver un algorithme déterministe de complexité polynomiale est resté longtemps un problème ouvert.

Il a cependant fallu attendre 2005 pour que Kayal [Kay05] propose un algorithme polynomial déterministe en utilisant les avancées dans le domaine de la factorisation des polynômes.

Nous commencerons ici par exposer des critères simples qui bien qu'inefficaces du point de vue algorithmique permettent d'obtenir de nombreux résultats théoriques. Nous expliquerons ensuite brièvement la méthode de Kayal sans rentrer les détails techniques qui dépassent notre propos. Nous décrirons ensuite les algorithmes probabilistes afin d'aboutir à un test efficace. Enfin, nous nous confronterons à l'expérience pour juger de l'application pratique de ces méthodes.

2.1 Modèle de complexité

Avant tout précisons le modèle de complexité utilisé. On considère ici des représentations denses. Cela signifie que les polynômes sont représentés par un tableau d'éléments du corps de base contenant tous leurs coefficients. Ainsi, la taille de l'entrée de nos algorithmes ne dépend que du degré du polynôme. Plus précisément, pour un polynôme de degré n défini sur $\mathbb{F}_q[X]$, l'entrée est de taille $n \log q$.

Il est important de noter que cette hypothèse ne correspond pas toujours à la réalité de nos besoins, en effet, on cherchera plutôt dans la pratique à tester des polynômes avec très peu de coefficients et il serait intéressant d'obtenir des résultats sur les représentations creuses.

Un algorithme fondamental de manipulation des polynômes est l'algorithme de multiplication. La complexité de cette opération intervient très souvent et nous utiliserons la notation classique $M(n)$ pour désigner le nombre d'opérations nécessaire pour multiplier deux polynômes de degré n. Les algorithmes les plus performants pour cela sont basés sur la transformée de Fourier et donnent $M(n) = \mathcal{O}(n \log(n) \log\log(n))$ [GG03].

2.2 Critères simples

Bien que ne donnant pas lieu à des algorithmes efficaces, les critères présentés ici sont souvent utilisés pour prouver que des familles de polynômes induisent des permutations. Ils peuvent aussi se révéler utiles pour réaliser un premier tri rapide et éliminer ainsi les polynômes qui sont "loin" d'être de permutation.

2.2.1 Critère de Hermite

Un premier critère, probablement le plus utilisé dans la littérature, est le critère de Hermite.

Théorème 2.1 (Critère de Hermite [LN97])**.** *Soit $\mathbb{F}_q$ un corps de caractéristique p. Un polynôme $P \in \mathbb{F}_q[X]$ est de permutation si et seulement si les conditions suivantes sont satisfaites :*

(i) Pour tout k compris entre 1 et $q-2$ tel que $p \nmid k$, $P(X)^k \pmod{X^q - X}$ est de degré inférieur strictement à $q-1$.

(ii) $P(X)^{q-1} \pmod{X^q - X}$ est de degré $q-1$.

Cependant, si l'on implémente directement ce test, la complexité obtenue est en $\mathcal{O}(q^2)$ ce qui est beaucoup trop élevé en pratique. Par contre, ce critère est très utilisé pour prouver que des familles polynômes sont de permutation. Nous verrons en détail un exemple d'utilisation de ce résultat tiré d'un article consacré aux fonctions APN [BCCLC06] dans le chapitre 5.2.1 consacré aux applications en cryptographie symétrique.

2.2.2 Calculer les images

L'algorithme le plus simple consiste à calculer toutes les valeurs $P(a)$ pour a dans $\mathbb{F}_q$, et regarder si elles sont distinctes. Pour un polynôme de degré n cela nécessite $\mathcal{O}(qn)$ opérations puisqu'une évaluation seule est effectuée en $\mathcal{O}(n)$ opérations, plus $\mathcal{O}(q)$ pour tester si les valeurs sont distinctes (avec l'hypothèse la plus optimiste). Puisqu'on peut s'arrêter à la première valeur rencontrée deux fois, la complexité moyenne sera en fait en $\mathcal{O}(n\sqrt{q})$, mais les cas où P est de permutation sont précisément ceux pour lesquels on doit calculer toutes les valeurs.

De manière équivalente, on peut tester la propriété suivante :

Lemme 2.2. *Le polynôme $P \in \mathbb{F}_q[X]$ est de permutation si et seulement si*

$$\prod_{a \in \mathbb{F}_q} (X - P(a)) = X^q - X.$$

Cette reformulation permet d'utiliser des algorithmes de calcul polynomiaux évolués et on peut espérer faire ce calcul avec une complexité dans le pire cas en $\mathcal{O}(\frac{q}{n} M(n) \log n)$ avec des méthodes de type diviser pour régner [BM75]. La place mémoire nécessaire étant en $\mathcal{O}(q \log q)$.

2.2.3 Calculer les antécédents

On peut aussi préférer vérifier le critère suivant, qui permet d'effectuer moins d'opérations si le polynôme n'est pas de permutation

Lemme 2.3. *Le polynôme $P \in \mathbb{F}_q[X]$ est de permutation si et seulement si*

$$\forall a \in \mathbb{F}_q, \deg(\text{pgcd}(X^q - X, P(X) - a)) = 1.$$

On trouve alors une complexité équivalente en utilisant un algorithme de calcul de pgcd sous-quadratique.

Tel quel, ce test a encore une complexité trop élevée mais nous allons voir que la version probabiliste correspondante est assez efficace.

2.3 Algorithme polynomial déterministe

Nous allons suivre l'exposé de Kayal [Kay05], qui poursuit les travaux de Ma et von zur Gathen [MvzG95].

Le point de départ est alors le théorème 1.14 reliant les polynômes de permutation et les polynômes exceptionnels.

Si le cardinal de notre corps est suffisamment petit par rapport au degré de notre polynôme, à savoir $q \leq n^4$, on calcule tout simplement toutes les valeurs prises par le polynôme. Les cas problématiques sont donc ceux pour lesquels la taille de l'entrée, c'est-à-dire le degré, est petite devant le cardinal du corps. On choisit comme seuil $q > n^4$.

Si le cardinal du corps est grand, le théorème 1.14 nous assure qu'il suffit de tester l'absolue irréductibilité du polynôme. Ma et von zur Gathen [MvzG95] avaient montré que la complexité de ce test est polynomialement réductible à la factorisation de polynômes bivariés. La conclusion apportée par Kayal est un algorithme déterministe pour effectuer ce qu'il appelle la *factorisation uniforme*.

Définition 2.4. Soit $h(x, y) \in \mathbb{F}_q[X, Y]$ un polynôme bivarié et

$$h(X, Y) = h_1(X, Y) \cdots, h_k(X, Y)$$

sa décomposition en facteurs irréductibles. $h(X, Y)$ est dite *uniforme* si et seulement si

$$\dim_{\mathbb{F}_q}(h_i(X, Y)) = \dim_{\mathbb{F}_q}(h(X, Y)) \quad \forall 1 \leq i \leq k$$

et

$$\deg(h_i(X, Y)) = \deg(h_j(X, Y)) \quad \forall 1 \leq i, j \leq k$$

Kayal a montré que l'on peut étendre l'algorithme de factorisation en degrés distincts (DDF pour *distinct degree factorization*) donné par Gao, Kaltofen et Lauder [GKL04] afin d'obtenir une décomposition en produits de facteurs uniformes.

Théorème 2.5 (Kayal [Kay05])**.** *Il existe un algorithme polynomial déterministe prenant en entrée le polynôme $h(X, Y) \in \mathbb{F}_q[X, Y]$, et donnant la décomposition*

$$\langle (h_1(X, Y), n_1, d_1), \ldots, (h_k(X, Y), n_k, d_k) \rangle$$

telle que $h(X, Y) = h_1(X, Y) \ldots h_k(X, Y)$, où les h_i sont des polynômes uniformes constitués de polynômes irréductibles de degré n_i, dont le corps de décomposition est de degré d_i sur $\mathbb{F}_p$.

Nous ne donnerons pas ici les détails techniques qui nous porteraient trop loin de notre propos, mais il est important de savoir que ce problème est résoluble en temps polynomial de manière déterministe. Comme souvent cependant, on préférera en pratique utiliser des méthodes probabilistes plus efficaces.

2.4 Méthodes probabilistes

Nous allons voir maintenant qu'il existe par contre des algorithmes probabilistes simples de complexité polynomiale (et même essentiellement linéaire) pour tester les polynômes de permutation.

2.4.1 Versions probabilistes

Il y a essentiellement deux manières d'obtenir des algorithmes probabilistes à partir des critères simples vus précédemment.

La première consiste à remplacer des relations polynomiales par leur évaluation dans une extension de notre corps de base.

Ainsi, pour le critère de Hermite (théorème 2.1), on peut remarquer que la condition (i) est équivalente à

$$\deg_X \left((P(X)+Y)^{q-1} - P(X)^{q-1} \pmod{X^q - X}\right) < q-1.$$

En effet, en utilisant la formule du binôme, on obtient

$$(P(X)+Y)^{q-1} = Y^{q-1} + P(X)^{q-1} + \sum_{i=1}^{q-2} \binom{q-1}{i} Y^{q-1-i} P(X)^i$$

soit

$$(P(X)+Y)^{q-1} - P(X)^{q-1} = Y^{q-1} + \sum_{i=1}^{q-2} \binom{q-1}{i} Y^{q-1-i} P(X)^i.$$

Les termes en Y sont de degrés différents assurant qu'il n'y a pas d'annulations de termes. Pris modulo $(X^q - X)$, on voit bien apparaître tous les termes $P(X)^i \pmod{X^q - X}$ pour lesquels le coefficient binomial est non nul, et c'est précisément ceux pour lesquels la caractéristique p ne divise pas i en vertu du théorème de Lucas [Luc78].

Plutôt que d'évaluer cette expression dans $\mathbb{F}_q[X,Y]$, ce qui serait coûteux, on peut préférer l'évaluer en un élément y de $\mathbb{F}_{q^m}$ pris au hasard ce qui nous permet de travailler avec des polynômes à une seule indéterminée dans $\mathbb{F}_{q^m}[X]$ pour obtenir

$$(P(X)+y)^{q-1} - P(X)^{q-1} \pmod{X^q - X} \in \mathbb{F}_{q^m}[X]$$

dont le terme de degré $q-1$ est égal à

$$\sum_{i=1}^{q-2} y^{q-1-i} P(X)^i \pmod{X^q - X}.$$

Si le polynôme à deux variables $(P(X)+Y)^{q-1} - P(X)^{q-1} \pmod{X^q - X}$ est de degré $(q-1)$ en X, la probabilité d'annuler le coefficient de X^{q-1} en l'évaluant est faible.

Malheureusement, cela ne suffit pas ici à obtenir une version efficace puisque la complexité resterait en $\mathcal{O}(M(q)\log q)$. L'analyse de complexité précise se trouve dans [vzG91a]. Cependant c'est cette même méthode qui sera utilisée pour obtenir l'algorithme le plus efficace que l'on trouve dans le même article.

La deuxième méthode consiste tout simplement à tester sur un échantillon réduit une propriété qui doit être vraie pour tous les éléments du corps. Nous allons voir tout de suite un exemple.

2.4.2 Algorithmes efficaces

Revenons au lemme 2.3.

Lemme *Le polynôme $P \in \mathbb{F}_q[X]$ est de permutation si et seulement si*

$$\forall a \in \mathbb{F}_q, \deg(\mathrm{pgcd}(X^q - X, P(X) - a)) = 1.$$

Ainsi, si l'on trouve un a tel que ce degré soit différent de 1, on dispose d'un témoin assurant que P n'est pas de permutation. En testant un petit nombre de valeurs a choisies au hasard, on obtient alors un algorithme probabiliste. En ce qui concerne la complexité, chaque pgcd peut être calculé en $\mathcal{O}(M(n)\log q)$ en utilisant une exponentiation rapide pour calculer $X^q \pmod{P(X)-a}$, puis un algorithme rapide de pgcd [GG03]. La question de la probabilité de succès est quant à elle reliée au nombre de valeurs distinctes prises par notre polynôme P.

On trouve dans [MvzG95] le théorème suivant :

Théorème 2.6. *Soit $P \in \mathbb{F}_q[X]$ de degré n. Si P n'est pas une permutation, alors il existe au moins $(q-1)/n$ valeurs non prises par P.*

Ainsi, pour obtenir une probabilité d'erreur inférieure à un ε donné, il suffit de prendre $\lceil 2n\log(\varepsilon^{-1})\rceil$ valeurs a. On obtient ainsi un algorithme efficace de complexité $\mathcal{O}(n^2 \log q \log(\varepsilon^{-1}))$. De plus, il présente l'avantage d'être très facile à implémenter. L'algorithme 1 ci-dessous en est une version simple en `Magma` .

Algorithme 1 Test probabiliste simple

```
MyIsProbablyPermutationPolynomial:=function(P,eps)
 R<x> := Parent(P);
 F := BaseRing(R);
 n := Degree(P);
 r := Ceiling( 2*n*Log(eps^-1) );
 random_set := [Random(F): i in [1..r]]
 res := not exists { a :
         a in random_set |
         Degree(Gcd(p-a,Modexp(X,#F,P-a)-X)) ne 1
        };
 return res;
end function;
```

Dans [vzG91a], von zur Gathen raffine cet algorithme pour obtenir un algorithme probabiliste de complexité $\mathcal{O}(n\log(\varepsilon^{-1}))$ si $\varepsilon \leqslant q^{-1}$, dont la réponse est correcte avec probabilité supérieure à $1-\varepsilon$.

Afin de pouvoir énoncer cet algorithme, nous avons besoin de la définition suivante.

Définition 2.7. Soient $\mathbb{K}$ un corps et a_0 et a_1 dans $\mathbb{K}[X]$. L'algorithme d'Euclide définit des polynômes q_i et a_i par

$$a_{i-1} = q_i a_i + a_{i+1} \text{ et } \deg(a_{i+1}) < \deg(a_i)$$

On appellera *représentation euclidienne* de a_0 et a_1 le ℓ-uplet $(q_1, \ldots, q_{\ell-1}; a_{\ell-1})$ où a_ℓ est le premier reste nul.

Cette représentation se calcule efficacement en utilisant par exemple l'algorithme de Knuth-Schönhage. En fait, les algorithmes efficaces de calcul de pgcd calculent cette séquence en entier avec la même complexité. On pourra par exemple consulter [Pet97, GG03].

Théorème 2.8. *L'algorithme probabiliste 2 décide si $P \in \mathcal{P}oly\mathcal{P}erm$ en un nombre d'opérations sur $\mathbb{F}_q$ quasi-linéaire $(n\log q)$, où n est le degré de P.*

Algorithme 2 Test probabiliste quasi-linéaire

Entrées: $P \in \mathbb{F}_q[X]$ un polynôme unitaire de degré $2 \leqslant k < q$, et $\varepsilon > 0$.
Sorties: **vrai** si le polynôme est de permutation, sinon **faux** avec probabilité au moins $1-\varepsilon$.

1: Soit $m = 1 + \lceil \log_q(2/\varepsilon) \rceil$.
2: Choisir u aléatoirement dans $\mathbb{F}_{q^m}$.
3: Soit $a_1 = P - u$ et $a_2 = X^q - X \pmod{a_1}$.
4: Calculer la représentation euclidienne $(q_2, \ldots, q_\ell, a_\ell)$ de (a_1, a_2).
5: Soient $d_i = \deg q_i$ et γ_i le coefficient de tête de q_i.
6: **si** $\deg a_\ell \geqslant 1$ **alors**
7: **renvoyer vrai**
8: **finsi**
9: poser $n_0 = q$, $n_1 = n$ et calculer $n_i = n_{i-1} - d_i$ et $s = \sum_{i=0}^{\ell-1} n_i n_{i+1} \mod 2$.
10: Calculer

$$v = \left((-1)^s \prod_{i=2}^{\ell} \gamma^{-(n_{i-1}+n_i)} \right) - (-1)^q (u^q - u)$$

11: **renvoyer** $(v = 0)$

Le calcul de a_2 à l'étape 3 nécessite $\mathcal{O}(M(n)\log q)$ opérations en faisant une exponentiation rapide.

La complexité du calcul de la représentation euclidienne est en $\mathcal{O}(M(n)\log n)$ opérations dans $\mathbb{F}_{q^m}$.

La valeur de v à la ligne 10 se calcule en $\mathcal{O}(\ell \log n + \log q)$ opérations dans $\mathbb{F}_{q^m}$ soit $\mathcal{O}(M(m)(n\log n + \log q))$ opérations dans $\mathbb{F}_q$.

On obtient donc au total une complexité quasi-linéaire essentiellement en $\mathcal{O}(M(n)\log q)$.

On remarquera que la quasi-linéarité provient de l'utilisation de l'algorithme de Knuth-Schönhage pour le calcul de la représentation euclidienne. En pratique, il s'agit d'un algorithme assez complexe à implémenter efficacement. Si l'on utilise l'algorithme d'Euclide, la complexité obtenue est alors comparable avec celle de l'algorithme précédent, en particulier quadratique en le degré du polynôme P.

2.5 Remarque

Il existe dans `Magma` une fonction `IsProbablyPermutationPolynomial` pour tester si un polynôme est de permutation. Les détails de l'algorithme utilisé ne sont pas documentés il est juste indiqué qu'un "test probabiliste est répété 100 fois", mais d'après nos expérimentations il semble très proche de l'algorithme 1. Nous avons aussi fait une implémentation naïve de l'algorithme 2, en particulier nous n'avons pas un algorithme optimal pour le pgcd qui se révèle être déjà assez efficace. Le code correspondant est donné en annexe A.

Afin de comparer les temps d'exécution de notre algorithme avec ceux de la fonction intégrée de `Magma`, on utilise les polynômes suivants pour différentes valeurs de m.

```
F:=GF(2^(2*m+1));
R<X>:=PolynomialRing(F);

// polynome de permutation creux
P1:=X^(2^(m+1)+1)+X^3+X;
// polynome de permutation (un peu plus) dense
P2:=Evaluate(P1,X-Random(F));

// polynome creux qui n'est pas de permutation
Q1:=P1+X^7+X^3;
// polynome dense qui n'est pas de permutation
Q2:=X^(2^(m+1)+1)+&+[Random(F)*X^i: i in [0..2^(m+1)]];
```

Nous avons choisi pour l'algorithme 1 un nombre d'itérations $n = 100$ identique à `Magma` . Pour l'algorithme 2 le seuil ε est fixé à 2^{-60}. On obtient alors les temps suivants.

m	9				10			
polynôme	P1	P2	Q1	Q2	P1	P2	Q1	Q2
magma v2.15-3	0,18	0,23	0,0	0,31	1,67	6,76	0,07	6,45
algo. 1	0,15	0,21	0,0	0,32	1,51	5,43	0,05	4,29
algo. 2	0,03	0,05	0,03	1,21	0,07	0,15	0,08	4,77

TAB. 2.1 – Comparaison des algorithmes 1, 2 et de `Magma`

On peut supposer au vu des temps obtenus que l'algorithme utilisé par `Magma` est très proche de l'algorithme 1. L'algorithme 2 semble quant à lui bien plus efficace dans de nombreux cas. Il est cependant pénalisé par le fait que lorsque le polynôme n'est pas

de permutation les calculs doivent être effectués jusqu'au bout. Ainsi dans les colonnes correspondant à Q_2 on voit un temps de calcul assez long pour l'algorithme 2. Il faut cependant noter que ce temps est du même ordre quel que soit le polynôme. Pour les deux autres algorithmes, le temps obtenu varie beaucoup d'un essai sur l'autre. En effet, le calcul d'un pgcd dure environ 2 secondes et le temps total dépend donc du nombre d'itérations effectuées. Sur notre exemple, pour $m = 10$, on peut ainsi voir que `Magma` a effectué trois itérations et notre algorithme deux, mais cette différence n'est absolument pas significative. Un bon compromis serait de faire un nombre constant et faible d'itérations de l'algorithme 1 avant de passer à l'utilisation de l'algorithme 2.

Il faut cependant noter que l'utilisation de ces algorithmes peut être relativement différente suivant les connaissances a priori. Si l'on s'intéresse à des polynômes au hasard et qui ont donc très peu de chance d'être de permutation, il est important d'éliminer très rapidement les cas les plus simples. Au contraire, si l'on se base sur une étude préalable et que les polynômes testés sont "proches" en un certain sens des polynômes de permutation, les paramètres choisis seront sûrement différents.

Chapitre 3

Classes de polynômes de permutation

Un des problèmes posés par Lidl et Mullen dans [LM88] était de trouver des familles de polynômes de permutation. Nous allons essayer dans ce chapitre de dresser une liste aussi complète que possible des résultats connus.

3.1 Les polynômes de petit degré

Lorsque le degré est suffisamment petit, une utilisation directe du critère de Hermite permet de classifier tout les polynômes de permutation. On peut aussi voir cela comme une autre conséquence du théorème 1.14 sur les polynômes exceptionnels. Si le degré n du polynôme est petit, les cas particuliers ne peuvent survenir que pour les corps $\mathbb{F}_q$ tels que $q < n^4$.

Afin de simplifier la classification, nous allons définir une forme *normalisée* pour nos polynômes.

Définition 3.1. Un polynôme $P \in \mathbb{F}_q[x]$ sera dit *normalisé* si :
- P est unitaire
- $P(0) = 0$
- si son degré n n'est pas divisible par la caractéristique de $\mathbb{F}_q$, alors le coefficient de x^{n-1} est 0.

Tout polynôme peut être mis sous forme normalisée en le composant avec des polynômes de permutation.

Les polynômes de degré inférieur à cinq ont alors été totalement décrits par Dickson dans [Dic 7].

	condition sur les coefficients	condition sur q
x		
x^2		$q = 0 \pmod 2$
x^3		$q \neq 1 \pmod 3$
$x^3 - ax$	a non carré	$q = 0 \pmod 3$
$x^4 \pm 3x$		$q = 7$
$x^4 + a_1x^2 + a_2x$	0 est la seule racine	$q = 0 \pmod 2$
x^5		$q \neq 1 \pmod 5$
$x^5 - ax$	$\forall b \in \mathbb{F}_q, b^4 \neq a$	$q \neq 0 \pmod 5$
$x^5 + ax$	$a^2 = 2$	$q = 9$
$x^5 \pm 2x$		$q = 7$
$x^5 + ax^3 \pm x^2 + 3a^2x$	a non carré	$q = 7$
$x^5 + ax^3 + 5^{-1}a^2x$		$q = \pm 2 \pmod 5$
$x^5 + ax^3 + 3a^2x$	a non carré	$q = 13$
$x^5 - 2ax^3 + a^2x$	a non carré	$q = 0 \pmod 5$

TAB. 3.1 – Polynômes de permutation normalisés de degré inférieur à 5

Une telle classification devient évidemment beaucoup plus difficile à établir lorsque le degré augmente.

Le cas des polynômes de la forme $X^8 + aX^j$ avec $j < 8$ a été étudié tout d'abord par Cavior [Cav63] qui laissa de nombreuses questions ouvertes. Les réponses ont été données par la suite par Turnwald [Tur88] dans le théorème suivant.

Théorème 3.2. *Le polynôme $X^8 + aX^j$ avec $1 \leq j \leq 7$ et $a \neq 0$ est de permutation sur $\mathbb{F}_q$ si et seulement si l'une des conditions suivantes est vérifiée :*

1. $j = 1$, $q = 2^{3r}$ *et* $a^{(q-1)/7} \neq 1$;
2. $j = 1$, $q = 29$ *et* $a \in \{-4, 4, -10, 10\}$;
3. $j = 2$, $q = 2^{2r}$ *et* $a^{(q-1)/3} \neq 1$;
4. $j = 3$, $q = 11$ *et* $a \in \{-2, 2, -4, 4\}$;
5. $j = 5$, $q = 4$ *et* $a \in \{-1, 1\}$;
6. $j = 5$, $q = 7$ *et* $a \in \{-3, 3\}$.

On peut remarquer que Chou avait lui aussi enoncé de manière indépendante certains de ces cas [Cho88].

Un deuxième critère de classement possible est le degré algébrique.

Définition 3.3. Soit $P \in \mathbb{F}_{p^n}[X]$, on appelle *degré algébrique* de P le poids maximal de l'écriture en base p des degrés de ses monômes.

Les polynômes de degré algébrique zéro, c'est-à-dire constant, ne sont clairement pas des permutations. Lorsque le degré augmente, la complexité de classification augmente elle aussi très rapidement. Commençons par les polynômes de degré algébrique un pour lesquels la situation est relativement simple.

3.2 Les polynômes linéarisés

Rappelons la définition vue en introduction. On se concentre ici sur le cas où le corps de base est le corps premier.

Définition 3.4. Un polynôme $P \in \mathbb{F}_{p^n}[X]$ est dit *linéarisé* si et seulement si et seulement s'il est du type

$$P = \sum_{i=0}^{k} a_i X^{p^i}.$$

Ces polynômes correspondent aux applications linéaires de $\mathbb{F}_q$ dans lui même. Un polynôme de permutation est un polynôme induisant une application bijective. Dans le cas d'applications linéaires, il suffit d'avoir un noyau réduit à $\{0\}$, on a donc le critère suivant.

Proposition 3.5. *Un polynôme linéarisé est un polynôme de permutation si et seulement si 0 est son unique racine dans $\mathbb{F}_{p^n}$.*

On appelle aussi parfois le groupe des polynômes de permutation linéarisés groupe de Betti-Mathieu.

Les théorèmes suivants sont classiques, on peut par exemple trouver les démonstrations dans [LN97].

Théorème 3.6. *Un polynôme de permutation est de permutation sur toutes les extensions finies de $\mathbb{F}_q$ si et seulement si et seulement s'il est de la forme $aX^{p^k} + b$ avec $a \neq 0$ et $k \geqslant 0$.*

Théorème 3.7. *Si $P \in \mathbb{F}_q[X]$ n'est pas de la forme $aX^{p^k} + b$, il existe une infinité d'extensions finies $\mathbb{F}_{q^r}$ de $\mathbb{F}_q$ telles que P ne soit pas un polynôme de permutation de $\mathbb{F}_{q^r}$.*

La vérification pratique du caractère de permutation de ces polynômes est donc finalement simple puisqu'il s'agit de résoudre un système linéaire.

L'étude s'est donc plus souvent portée sur des familles plus spécifiques. On trouve par exemple dans [DGG99] un algorithme pour générer tout les orthomorphismes linéaires, c'est-à-dire les polynômes linéarisés de permutation $P(X)$ tels que $P(X) + X$ soit aussi un polynôme de permutation.

On trouve aussi un certain nombre de résultats quantitatifs. Par exemple la propriété suivante, qui est la conséquence d'un theorème que nous verrons plus tard.

Proposition 3.8. *Pour tout polynôme linéarisé $L \in \mathbb{F}_q[X]$, il existe $a \in \mathbb{F}_q$ tel que $L - aX$ soit un polynôme de permutation.*

Démonstration. Si L est un monôme, alors $a = 0$ convient. Sinon, nous verrons plus tard que les polynômes du type $L(X)/X$, où $L(X)$ est un polynôme linéarisé, ne peuvent être de permutation que si ce sont des monômes (cf. 3.7.1). Il existe donc au moins une valeur a qui n'est pas prise par ce polynôme, et le polynôme linéarisé

$$X(L(X)/X - a) = L(X) - aX$$

n'a alors pas de racine autre que 0 et est donc de permutation. □

3.3 Polynômes quadratiques

Les polynômes de degré algébrique 2, c'est-à-dire où tous les monômes sont du type $X^{q^i+q^j}$, ont une importance toute particulière en cryptographie à clé publique. En effet, de nombreux systèmes de cryptographie multivariée utilisent de tels polynômes comme nous le verrons plus en détail dans le chapitre 5.

Le principe de base est souvent de masquer une permutation centrale que l'on sait inverser, représentée par un polynôme quadratique, en la composant à droite et à gauche par des permutations linéaires. On a besoin de permutation car on a besoin de calculer l'inverse pour déchiffrer. Le caractère quadratique permet quant à lui de conserver des polynômes lacunaires et donc d'obtenir des tailles de clés suffisamment faibles.

Le problème de trouver de telles permutations se pose donc naturellement. Bien que moins important pour la validité de son algorithme, Jacques Patarin dans une version étendue de son article d'Eurocrypt 1996 introduisant HFE, a aussi introduit le problème de trouver de telles familles [Pat96a, Pat96b].

3.3.1 Premiers exemples

On peut tout d'abord rechercher des exemples parmi les classes que nous avons déjà étudiées.

On trouve bien sûr des monômes. En utilisant des résultats élémentaires de divisibilité et la propriété 1.3, on peut par exemple énoncer le résultat suivant :

Proposition 3.9. *Le monôme X^{2^k+1} est un polynôme de permutation sur $\mathbb{F}_{2^n}$ si et seulement si $n/\operatorname{pgcd}(n,k)$ est impair.*

On trouve aussi d'autres exemples plus sporadiques. Ainsi Hans Dobbertin démontre le résultat suivant :

Théorème 3.10 (Dobbertin [Dob99])**.**

$$X^{2^{m+1}+1} + X^3 + X$$

est un polynôme de permutation sur $\mathbb{F}_{2^{2m+1}}$.

On pourrait aussi citer des classes de binômes mais nous reviendrons sur ces exemples plus tard au théorème 3.28.

Dans la suite, nous allons nous concentrer sur des polynômes purement quadratiques, c'est-à-dire sans termes linéaires. On remarque alors qu'en caractéristiques différentes de 2, tous les exposants sont pairs et on a en particulier toujours égalité entre $P(X)$ et $P(-X)$. Les seuls polynômes de permutation de ce type pouvant exister sont donc obtenus en caractéristique 2.

Afin de faire une étude plus systématique, Blokhuis et al. ont introduit une classe plus réduite de polynômes. Nous allons présenter leurs résultats ainsi que la généralisation que nous avons proposée [LC07a].

3.3.2 Polynômes bilinéaires

Un moyen pour obtenir un polynôme de degré algébrique 2 est de prendre le produit de deux polynômes linéarisés. En suivant la terminologie utilisée par Blokhuis et al. [BCHO99], nous utiliserons ici la définition suivante :

Définition 3.11. Un polynôme *bilinéaire* est un polynôme de la forme :

$$P(X) = L_1(X)L_2(X) \pmod{X^q - X}$$

avec L_1 et L_2 des polynômes linéarisés.

Il faut bien faire attention au fait que cette définition n'a rien à voir avec la définition usuelle de bilinéarité.

Un simple argument de comptage permet de se convaincre que l'on ne peut pas obtenir ainsi tout les polynômes quadratiques. En effet, il existe 2^{n^2} polynômes linéarisés à coefficients dans $\mathbb{F}_{2^n}$, ce qui donne au plus 2^{2n^2} polynômes bilinéaires alors qu'il existe $2^{\frac{n^2(n-1)}{2}}$ polynômes quadratiques.

Afin de réduire encore le corpus de notre étude, nous allons considérer à équivalence linéaire près. Comme nous l'avons déjà dit en introduction, cela se justifie d'un point de vue applicatif puisque des polynômes linéairement équivalents vont avoir (le plus souvent) les mêmes propriétés cryptographiques.

Si l'on prend un polynôme de la forme $P(X) = L_1(X)L_2(X)$, en composant avec l'inverse de L_1, qui existe et est linéaire, on obtient la propriété suivante :

Proposition 3.12. *Tout polynôme bilinéaire est linéairement équivalent à un polynôme du type*

$$P(X) = XL_1(X)$$

avec L_1 un polynôme linéaire.

On remarque l'analogie avec le problème rencontré précédemment où l'on s'intéressait aux polynômes de la forme $L(X)/X$. Toutefois, nous allons voir que les permutations que l'on peut obtenir sont ici bien plus nombreuses. Tout d'abord, on peut voir que l'on trouve trois familles linéairement équivalentes entre elles.

Théorème 3.13. *Soient k et n deux entiers, et $d = \gcd(n,k)$. Si n/d est impair alors les trois classes suivantes définissent des permutations de $\mathbb{F}_{2^n}$:*

(i) X^{2^k+1}

(ii) $X^{2^k+1} + aX^{2^{n-k}+1}$ *si* $a^{\frac{2^n-1}{2^d-1}} \neq 1$

(iii) $X^{2^{2k}+1} + (aX)^{2^k+1} + aX^2$ *si* $a^{\frac{2^n-1}{2^d-1}} \neq 1$ *et* $n=3k$.

De plus, ces trois classes sont linéairement équivalentes.

Du point de vue cryptographique, cela n'est pas très satisfaisant puisque l'on reste avec des polynômes linéairement équivalents à des fonctions puissances, et que notre but est a priori de trouver des familles plus riches.

Un premier exemple de polynôme non équivalent est donné par Blokhuis et al. [BCHO99]. Il fait partie de la famille suivante.

Théorème 3.14. *Soit ℓ un entier et k un entier impair. Les polynômes suivants définissent des permutations de $\mathbb{F}_{2^{k\ell}}$:*

(iv) $X(\mathrm{Tr}_\ell^{k\ell}(X) + aX), \quad a \in \mathbb{F}_{2^\ell} \setminus \mathbb{F}_2$

Nous avons généralisé leur résultat en construisant de manière récursive des familles de polynômes.

Théorème 3.15 (Laigle-Chapuy [LC07a])**.** *Soit ℓ un entier et k un entier impair. Soit $L \in \mathbb{F}_{2^\ell}[X]$ un polynôme linéaire tel que $XL(X)$ est un polynôme de permutation bilinéaire sur $\mathbb{F}_{2^\ell}$. Alors les polynômes suivants définissent des permutations de $\mathbb{F}_{2^{k\ell}}$:*

(v) $X(L(\mathrm{Tr}_\ell^{k\ell}(X)) + a\,\mathrm{Tr}_\ell^{k\ell}(X) + aX), \quad a \in \mathbb{F}_{2^\ell}^\star$

Preuve : On remarque tout d'abord que le cas (iv) se déduit de (v) en utilisant $L(X) = X$ et en remarquant que pour $a \neq 1$

$$X(\mathrm{Tr}_\ell^{k\ell}(X) + aX) = (a+1)X\left(\mathrm{Tr}_\ell^{k\ell}(X) + \frac{a}{a+1}\,\mathrm{Tr}_\ell^{k\ell}(X) + \frac{a}{a+1}X\right)$$

Passons maintenant à la preuve du théorème. Soit $Q(X) = XL(X)$ un polynôme bilinéaire de permutation sur $\mathbb{F}_{2^\ell}$, $a \in \mathbb{F}_{2^\ell}^\star$ et $P(X) = X(L(\mathrm{Tr}_\ell^{k\ell}(X)) + a\,\mathrm{Tr}_\ell^{k\ell}(X) + aX)$.

Pour tout $x \in \mathbb{F}_{2^n}$,

$$\mathrm{Tr}_\ell^{k\ell}(P(x)) = \mathrm{Tr}_\ell^{k\ell}(x)\,\mathrm{Tr}_\ell^{k\ell}(L(\mathrm{Tr}_\ell^{k\ell}(x))).$$

De plus, comme k est impair, $L(\mathrm{Tr}_\ell^{k\ell}(x))$ qui appartient à $\mathbb{F}_{2^\ell}$ est égal à sa trace. On obtient alors

$$\mathrm{Tr}_\ell^{k\ell}(P(x)) = Q(\mathrm{Tr}_\ell^{k\ell}(x)).$$

Ainsi, si x et y sont tels que $P(x) = P(y)$, on a aussi

$$Q(\mathrm{Tr}_\ell^{k\ell}(x)) = Q(\mathrm{Tr}_\ell^{k\ell}(y)).$$

Or, Q permute $\mathbb{F}_{2^\ell}$ donc $\mathrm{Tr}_\ell^{k\ell}(x) = \mathrm{Tr}_\ell^{k\ell}(y)$. Notons t cette trace.

$$\begin{aligned} P(x) = P(y) &\Leftrightarrow x(L(t) + at + ax) = y(L(t) + at + ay) \\ &\Leftrightarrow a(x+y)(a^{-1}L(t) + t + x + y) = 0 \end{aligned}$$

Ce qui implique que $x+y = 0$ ou $x+y = a^{-1}L(t)+t$ et dans les deux cas, $x+y$ appartient à $\mathbb{F}_{2^\ell}$.

On a alors finalement $x + y = \mathrm{Tr}_\ell^{k\ell}(x + y) = t + t = 0$ et $x = y$. P induit bien une permutation. □

Exemple 3.16. Soit α un élement primitif de $\mathbb{F}_{2^9}$.

$$X^{65} + X^9 + \alpha^{73}X^2$$

et

$$X^{129} + X^{65} + X^{17} + X^9 + X^3$$

sont des polynômes de permutation de $\mathbb{F}_{2^9}$ de type (iv) et (v) respectivement.

De plus, ils ne sont linéairement équivalents ni à un monôme, ni l'un à l'autre. On remarque donc en particulier que notre théorème donne bien de nouveaux polynômes de permutation, différents de ceux connus jusqu'alors.

Exemple 3.17. En partant de la permutation de type (iv) sur $\mathbb{F}_{2^6}$

$$P_6(X) = X^{17} + X^5 + aX^2$$

avec $a \in \mathbb{F}_4 \backslash \mathbb{F}_2$ et en prenant $b \in \mathbb{F}_{2^5}^\star$, on peut construire la permutation de $\mathbb{F}_{2^{30}}$ suivante :

$$\begin{aligned} P_{30}(X) &= X\left(\frac{P_6(X)}{X} \circ \mathrm{Tr}_6^{30}(X) + b\,\mathrm{Tr}_6^{30}(X) + bX\right) \\ &= X^{2^{28}+1} + X^{2^{26}+1} + X^{2^{22}+1} + X^{2^{20}+1} + X^{2^{16}+1} \\ &\quad + X^{2^{14}+1} + X^{2^{10}+1} + X^{2^8+1} + X^{2^4+1} + X^{2^2+1} \\ &\quad + (a+b)\left(X^{2^{24}+1} + X^{2^{18}+1} + X^{2^{12}+1} + X^{2^6+1}\right) + aX^2. \end{aligned}$$

Exemple 3.18. Dans $\mathbb{F}_{2^{15}}$, les polynômes suivants sont des permutations de type (v) :

$X^{2049} + aX^{1025} + X^{65} + aX^{33} + X^3$	$a \in \mathbb{F}_{32}^\star$
$X^{4097} + aX^{1025} + X^{129} + aX^{33} + X^5$	$a \in \mathbb{F}_{32}^\star$
$X^{8193} + aX^{1025} + X^{257} + aX^{33} + X^9$	$a \in \mathbb{F}_{32}^\star$
$X^{8193} + X^{1025} + X^{129} + X^{17} + X^3 + a\,(X^{4097} + X^{513} + X^{65} + X^9)$	$a \in \mathbb{F}_8^\star$
$X^{16385} + aX^{1025} + X^{513} + aX^{33} + X^{17}$	$a \in \mathbb{F}_{32}^\star$
$X^{16385} + X^{2049} + X^{257} + X^{33} + X^5 + a\,(X^{4097} + X^{513} + X^{65} + X^9)$	$a \in \mathbb{F}_8^\star$

Le problème de savoir lesquels parmi ces polynômes sont équivalents est pour l'instant toujours ouvert.

Même si donner une formule pour l'inverse de ces polynômes semble compliqué, il est intéressant de noter que la preuve du théorème 3.15 donne un moyen algorithmique simple d'inverser ces polynômes.

En effet, pour initialiser notre construction, les seules permutations dont on dispose a priori sont des monômes, que l'on sait par conséquent inverser.

Supposons maintenant que l'on ait un polynôme $P(X)$ de la forme $X(L(\mathrm{Tr}_\ell^{k\ell}(X)) + a\,\mathrm{Tr}_\ell^{k\ell}(X) + aX)$ avec $a \in \mathbb{F}_{2^\ell}^\star$, comme défini dans le théorème 3.15. Pour calculer l'inverse d'un élément $y \in \mathbb{F}_{2^{k\ell}}$, on peut appliquer l'algorithme 3.

Algorithme 3 Inversion d'un polynôme bilinéaire de permutation

Entrées: $P(X)$ un polynôme de permutation construit comme dans le théorème 3.15 et $y \in \mathbb{F}_{2^{k\ell}}$.

Sorties: $x \in \mathbb{F}_{2^{k\ell}}$ tel que $P(x) = y$.

1: Calculer $\mathrm{Tr}_\ell^{k\ell}(y)$.

2: Trouver $t \in \mathbb{F}_{2^\ell}$ tel que $tL(t) = \mathrm{Tr}_\ell^{k\ell}(y)$.

3: Résoudre dans $\mathbb{F}_{2^{k\ell}}$ l'équation de degré 2

$$x(L(t) + at + ax) = y.$$

4: **renvoyer** la solution x vérifiant $\mathrm{Tr}_\ell^{k\ell}(x) = t$.

À la ligne 2, on doit procéder récursivement à l'inversion d'un polynôme de permutation bilinéaire sur un corps plus petit, ce que l'on sait faire par hypothèse.

Exemple 3.19. Reprenons un des exemples précédents :

$$P(X) = X^{129} + X^{65} + X^{17} + X^9 + X^3$$

sur $\mathbb{F}_{2^9}$. On a

$$P(X) = X(\mathrm{Tr}_3^9(X)^2 + \mathrm{Tr}_3^9(X) + X).$$

On a donc simplement ici $L(X) = X^2$.

Calculons $P^{-1}(\alpha^7)$ avec α un élément primitif de $\mathbb{F}_{2^9}$.

On trouve $t = \mathrm{Tr}_3^9(\alpha^7)^5 = \alpha^{438}$ puisque l'inverse pour la composition de $XL(X)$ est ici X^5.

Il nous reste donc à résoudre dans $\mathbb{F}_{2^9}$ l'équation

$$X^2 + (t^2 + t)X + \alpha^7 = 0.$$

On trouve deux solutions α^{237} et α^{281} de traces respectives t et α^{365}.

On en déduit finalement que $P^{-1}(\alpha^7) = \alpha^{237}$.

Dans ce théorème, on remarque qu'il est nécessaire pour pouvoir construire des polynômes que le degré d'extension n de notre corps par rapport à $\mathbb{F}_2$ ait un diviseur impair. Cela nous a amené à formuler les deux conjectures suivantes :

Conjecture 3.20. *Si n est une puissance de deux, X^2 est le seul polynôme bilinéaire unitaire de permutation sur $\mathbb{F}_{2^n}$.*

Conjecture 3.21. *Il n'y a pas de polynôme de permutation bilinéaire non monomial sur $\mathbb{F}_{2^p}$ pour p premier.*

De plus, nous allons voir un résultat appuyant un peu plus la conjecture 3.20.

Théorème 3.22. *Soit n_0 un entier tel que X^2 soit le seul polynôme unitaire bilinéaire de permutation sur $\mathbb{F}_{2^{2^{n_0}}}$.*
Alors pour tout $n \geq n_0$, le seul polynôme bilinéaire unitaire de permutation sur $\mathbb{F}_{2^{2^n}}$ à coefficients dans $\mathbb{F}_{2^{2^{n_0}}}$ est X^2.

Preuve : Supposons que le résultat est vrai jusqu'a $n-1$, $n > n_0$, et posons $t = 2^{n-1}$. Soit

$$P(X) = \sum_{i=0}^{2t-1} \lambda_i X^{2^i+1} \quad \in \mathbb{F}_{2^{2t}}$$

avec $\lambda_i \in \mathbb{F}_{2^{2^{n_0}}}$ un polynôme de permutation sur $\mathbb{F}_{2^{2^n}}$. P doit en particulier permuter les éléments de $\mathbb{F}_{2^t}$. Cela implique d'après l'hypothèse de récurence que $P \mod X^{2^t} + X$ doit être égal à X^2 ce qui nous donne :

$$\sum_{i=0}^{t-1} (\lambda_i + \lambda_{i+t}) X^{2^i+1} = X^2.$$

On réécrit alors

$$P(X) = XH(X) \text{ avec } H(X) = X + \mathrm{Tr}_t^{2t}\left(\sum_{i=1}^{t-1} \lambda_i X^{2^i}\right).$$

On remarque alors que H induit l'identité de $\mathbb{F}_{2^t}$.
Si $P(X) \neq X^2$, il existe $x \in \mathbb{F}_{2^{2t}} \setminus \mathbb{F}_{2^t}$ tel que $H(x) + x = \beta \neq 0$. On a alors

$$P(x+\beta) = P(x) + \beta\,(H(x) + x + \beta) = P(x)$$

ce qui prouve que P n'est pas un polynôme de permutation et on arrive à une contradiction.

Le théorème en découle alors par récurrence. □

Corollaire 3.23. *Pour $n \geq 2$, le seul polynôme bilinéaire unitaire de permutation sur $\mathbb{F}_{2^{2^n}}$ à coefficients dans $\mathbb{F}_{16}$ est X^2.*

Preuve : Pour $n_0 = 2$, on peut établir le résultat par recherche exhaustive. On applique alors le théorème précédent. □

3.4 Les binômes

Jusqu'ici nous avons classé nos polynômes en fonction de leur degré et de leur degré algébrique. Une autre voie est de considérer le nombre de termes. Le cas des monômes est simple et bien compris comme nous l'avons vu en introduction, cependant dès que l'on passe aux binômes la classification devient très compliquée et de nombreux articles ont étés consacrés à ce problème.

Rappelons qu'il s'agit du problème numéro 13 donné par Lidl et Mullen [LM93].

Citons tout d'abord des résultats d'existence liés aux propriétés évoquées en 1.2 sur les polynômes exceptionnels. Tout d'abord deux résultat de Turnwald [Tur88].

Théorème 3.24. *Soient p un nombre premier, $q = p^k$ et $n, m \in \mathbb{N}^\star$ tels que $n > m$. Si $X^n + aX^m, a \in \mathbb{F}_q^\star$ est un polynôme de permutation, alors ou bien pm divise n ou bien $q \leq (n-2)^4 + 4k - 4$.*

Théorème 3.25. *Soient p un nombre premier, $q = p^k$ et $n, m \in \mathbb{N}^\star$ tels que $n > m$. Si $X^n + aX^m, a \in \mathbb{F}_q^\star$ est un polynôme de permutation, alors $p \geq \max\{m, n-m\}$.*

Plus récemment, Ariane Masuda et Michael Zieve ont amélioré ce résultat dans le cas de corps premiers [MZar].

Théorème 3.26. *Soit $\mathbb{F}_p$ un corps premier et $n, m \in \mathbb{N}^\star$. Si $X^n + aX^m, a \in \mathbb{F}_p^\star$ est un polynôme de permutation de $\mathbb{F}_p$, alors*

$$\operatorname{pgcd}(n-m, p-1) \geq \sqrt{p - (3/4)} - 1/2 > \sqrt{p} - 1.$$

Cependant, cela ne nous donne toujours pas de familles concrètes.

En fait, nous en avons déjà vu plusieurs exemples précédemment. A l'aide du théorème 3.24, Turnwald a établi la liste des binômes de permutation de la forme $X^8 + aX$ que nous avons vu au théorème 3.2.

Un autre exemple pris parmi les polynômes linéarisés est donné par les polynômes du type

$$X^{p^i} - aX, \quad a \in \mathbb{F}_q,$$

qui sont des polynômes de permutation de $\mathbb{F}_q$, corps de caractéristique p, si et seulement si a n'est pas une racine $(p^i - 1)$-ème de l'unité.

Il est intéressant de noter que le théorème 3.24 que nous venons de voir implique en fait que si $q > n^4$ alors les seuls binômes de permutation de degré m sont des compositions de ce polynôme avec des monômes.

Une autre famille a été donnée par Luyan Wang.

Proposition 3.27 ([Wan02])**.** *Soit q,u et v tels que $3|q-1$, $\operatorname{pgcd}(v, q-1) = (q-1)/3$. Alors $X^u(X^v + 1)$ est un polynôme de permutation de $\mathbb{F}_q$ si et seulement si $\operatorname{pgcd}(u, (q-1)/3) = 1$, $u \neq v \pmod 3$ et $2^{(q-1)/3} = 1$ dans $\mathbb{F}_q$.*

Dans le même article, il donne des conditions semblables pour $5|q-1$ et $\operatorname{pgcd}(v, q-1) = (q-1)/5$ faisant intervenir des suites de Lucas. En fait, la généralisation se poursuit plus loin et c'est ce que l'on trouve dans un article de Amir Akbary et Qiang Wang [AW06, AW07, AAW08], exprimé aussi en terme de suites de Lucas. Nous n'énoncerons

pas ici ces réultats car comme le montre Michael Zieve [Zieb] il peuvent être exprimés beaucoup plus simplement en changeant légèrement de point de vue.

En effet, comme le montre le théorème 3.26, un critère important semble être la différence $(n-m)$, et plus précisement les facteurs communs avec $(q-1)$. Cela nous incite donc à considérer les binômes sous la forme $X^r(X^{s(q-1)/m}+a)$ pour des valeurs de paramètres adaptées. On se rend compte alors qu'il s'agit de résultats contenus dans l'article [LC07b] correspondant au travail effectué au début de cette thèse.

C'est donc dans la section suivante 3.5 consacrée aux polynômes s'écrivant sous la forme $X^rP(X^{(q-1)/m})$ que ces résultats trouveront leur place.

Enfin, une série de binômes a été trouvée dans un autre contexte, en s'intéressant plus directement aux applications [BCL08]. Les motivations seront expliquées dans le chapitre sur la cryptographie symétrique 5.2, pour l'instant contentons-nous du résultat.

Théorème 3.28. *Soient s et k des entiers, et $i \in \{1,2\}$. Notons*

$$g_1 = \operatorname{pgcd}(2^{3k}-1, 2^{ik}+2^{tk+s}-(2^s+1))$$

et

$$g_2 = \operatorname{pgcd}(2^{k}-1, 2^{ik}+2^{tk+s}-(2^s+1)).$$

Si $3k/\operatorname{pgcd}(s,3k)$ est impair , $g_1 \neq g_2$, et α un élément primitif de $\mathbb{F}_{2^{3k}}$, le polynôme

$$P = X^{2^{ik}+2^{(3-i)k+s}} + \alpha^{2^k-1}X^{2^s+1}$$

est de permutation sur $\mathbb{F}_{2^{3k}}$.

Preuve : La preuve utilise la technique multivariée décrite par Dobbertin dans [Dob02].

Considérons l'équation en x, avec t comme paramètre :

$$P(x)+P(x+t)=0, \quad t \in \mathbb{F}_{2^n} \tag{3.1}$$

Pour t non nul, il ne doit pas exister de solutions. Or dans ce cas on a

$$\begin{aligned} P(x)+P(x+t) &= t^{2^{(3-i)k+s}+2^{ik}}\left(\left(\frac{x}{t}\right)^{2^{ik}}+\left(\frac{x}{t}\right)^{2^{(3-i)k+s}}+1\right) \\ &\quad +\alpha^{2^k-1}t^{2^s+1}\left(\left(\frac{x}{t}\right)^{2^s}+\left(\frac{x}{t}\right)+1\right) \end{aligned}$$

Et en faisant le changement de variable $x=ty$ on obtient l'équation

$$a(y^{2^{ik}}+y^{2^{(3-i)k+s}}+1)+(y^{2^s}+y+1)=0 \tag{e_1}$$

où

$$a = t^{2^{ik}+2^{(3-i)k+s}-2^s-1}/\alpha^{2^k-1} = \left(t^{2^{k+s}+2^s+1-2^k(2^s-1)(i-1)}/\alpha\right)^{2^k-1} \tag{3.2}$$

Nous allons essayer de nous ramener à une équation de plus bas degré. Pour ce faire, posons $z=y^{2^k}$, $t=z^{2^k}$, $b=a^{2^k}$ et $c=b^{2^k}$. a étant une puissance (2^k-1)-ième on a en particulier $abc=1$.

On voit donc qu'encore une fois cette méthode utilise d'une certaine manière les divisibilités de $2^{3k}-1$. Après avoir introduit ces variables formelles, nous allons procéder à une élimination de Gauss.

cas i=1

On obtient alors

$$\begin{array}{lll} a(z+t^{2^s}+1)+(y^{2^s}+y+1) &= 0 & (e_1) \\ b(t+y^{2^s}+1)+(z^{2^s}+z+1) &= 0 & (e_2) = (e_1)^{2^k} \\ c(y+z^{2^s}+1)+(y^{2^s}+y+1) &= 0 & (e_3) = (e_2)^{2^k} \end{array}$$

En éliminant z et t, on peut alors obtenir l'équation suivante :

$$\left(Q(a)(y^{2^s}+y+1)\right)^{2^s}=0$$

$$\text{où } \; Q(a)=(1+a)\frac{(ab)^{2^s+1}+(ab)^{2^s}+b^{2^s}a+ab+a+b^{2^s}}{a^2\left((ab)^{2^s}+1\right)}$$

cas i=2

Les équations sont dans ce cas

$$\begin{array}{lll} e_1 & = & a(t+z^{2^s}+1)+(y^{2^s}+y+1) = 0 \\ e_2=e_1^{2^k} & = & b(y+t^{2^s}+1)+(z^{2^s}+z+1) = 0 \\ e_3=e_2^{2^k} & = & c(z+y^{2^s}+1)+(t^{2^s}+t+1) = 0 \end{array}$$

Une fois z et t éliminés, il reste :

$$\left(R(a)(y^{2^s}+y+1)\right)^{2^s}=0$$

$$\text{où } R(a)=(b+1)\frac{(ab)^{2^s+1}+(ab)^{2^s}+a^{2^s}+a^{2^s}b+a^{2^s}+ab+b}{a\left(a^{2^s}\left(b^{2^s+1}+b^{2^s}+b^2+1\right)+b^2+b\right)}$$

$P(x)+P(x+t)=0$ entraîne donc $Q(a)(y^{2^s}+y+1)=0$ ou $R(a)(y^{2^s}+y+1)=0$ suivant la valeur de i.

Le polynôme $y^{2^s}+y+1$ n'a pas de racine dans $\mathbf{F}_{3k}$ car $3k/\operatorname{pgcd}(s,3k)$ est impair (voir [CH04]).

Il nous reste donc à montrer que $Q(a)$ et $R(a)$ sont non nuls.

$$\begin{aligned} Q(a)=0 \;&\Leftrightarrow\; a=1 \text{ ou } (ab)^{2^s+1}+(ab)^{2^s}+b^{2^s}a+ab+a+b^{2^s}=0 \\ &\Leftrightarrow\; a=1 \text{ ou } a=\left(\left(\frac{a+1}{ab+1}\right)^{2^{n-s}}\right)^{2^{k+s}+2^s+1} \end{aligned}$$

De même,

$$R(a)=0 \;\Leftrightarrow\; b=1 \text{ ou } a=\left(\frac{a+1}{ab+1}\right)^{2^{k+s}+2^s+1}$$

Or si a est une puissance $(2^{k+s}+2^s+1)$, l'équation (3.2) implique que α est une puissance (g_1/g_2) ce qui n'est pas possible α étant un élément primitif.

□

3.5 Polynômes du type $X^r P\big(X^{(q-1)/m}\big)$

Il ne s'agit pas à proprement parler d'une classe de polynômes puisque tout polynôme peut être mis sous cette forme. Cependant, nous allons développer ici des outils spécifiques pour étudier les polynômes en utilisant cette écriture, et nous l'utiliserons ensuite pour donner quelques familles explicites.

Nous verrons aussi par la suite au chapitre 4 comment utiliser ces résultats pour estimer le nombre de polynômes de permutation avec une forme préscrite.

Le théorème central a été énoncé par Lidl et Wan.

Théorème 3.29 (Lidl, Wan [WL91])**.**
Soient m et r des entiers naturels tels que $m|q-1$, α un élément primitif de $\mathbb{F}_q$, et $P \in \mathbb{F}_q[X]$.
Alors $Q = X^r P(X^{(q-1)/m})$ est un polynôme de permutation de $\mathbb{F}_q$ si et seulement si les conditions suivantes sont vérifiées :

(i) $\operatorname{pgcd}(r, (q-1)/m) = 1$.

(ii) Pour tout $0 \leqslant i < m$, $P(\alpha^{i\frac{q-1}{m}}) \neq 0$.

(iii) Pour tout $0 \leqslant i < j < m$,

$$Q(\alpha^i)^{(q-1)/m} \neq Q(\alpha^j)^{(q-1)/m}.$$

Preuve : Si $X^r P(X^{(q-1)/m})$ est un polynôme de permutation, il n'a qu'une racine, 0, la condition (ii) est donc nécessaire. Si (ii) est vérifiée, montrons que (i) et (iii) sont nécessaires et suffisantes. Soit α une racine primitive de $\mathbb{F}_q$ et $\omega = \alpha^{\frac{q-1}{m}}$.
Prenons $k = i + mj$ avec $0 \leqslant i < m$ et $0 \leqslant j < (q-1)/m$.
Cela nous donne alors

$$Q(\alpha^k) = \alpha^{kr} P(\alpha^{k\frac{q-1}{m}}) = \alpha^{mrj}\alpha^{ri} P(\omega^i).$$

On a donc un polynôme de permutation si et seulement si :

- Pour $0 \leqslant j_1 < j_2 < (q-1)/m$, $rj_1 \neq rj_2 \mod (q-1)/m$. Cela équivaut à la condition (i).
- Pour $0 \leqslant i_1 < i_2 < m$,

$$(\alpha^{ri_1} P(\omega^{i_1}))^{(q-1)/m} \neq (\alpha^{ri_2} P(\omega^{i_2}))^{(q-1)/m} \mod m$$

En effet, supposons le contraire et prenons i_1 et i_2 tels que

$$\left(\frac{\alpha^{ri_1} P(\omega^{i_1})}{\alpha^{ri_2} P(\omega^{i_2})}\right)^{(q-1)/m} = 1$$

Comme α est un élément primitif et r est premier avec $\frac{q-1}{m}$, α^{mr} engendre toute les racines $\left(\frac{q-1}{m}\right)$-ièmes de l'unité. Il existe donc ℓ tel que

$$\alpha^{ri_1} P(\omega^{i_1}) = \alpha^{mr\ell}\alpha^{ri_2} P(\omega^{i_2}).$$

En prenant j_1 et j_2 tels que $j_2 - j_1 = \ell$, on obtient alors $Q(\alpha^{i_1+mj_1}) = Q(\alpha^{i_2+mj_2})$ ce qui est exclu.
Il s'agit bien de la condition (iii) et le théorème est démontré.

□

Il s'agit d'un résultat simple mais qui permet de bien comprendre la structure de ces polynômes. On peut ensuite par exemple construire des polynômes de permutation lacunaires en utilisant ce résultat et en résolvant simplement un système d'équations linéaires.

Proposition 3.30 (Lidl, Wan [WL91]). *Soient m un diviseur de $q-1$, et $r \in \mathbb{N}$ tel que* $\gcd(r,(q-1)/m)=1$.
Soient $\pi \in \mathcal{S}_m$ une permutation et $t : [\![0, m-1]\!] \to [\![0, \frac{q-1}{m}-1]\!]$. Notons $\omega = \alpha^{(q-1)/m}$.
Alors le système

$$\text{Pour } 0 \leqslant i < m, \qquad \sum_{j=0}^{m-1} a_j \omega^{ij} = \alpha^{\pi(i)-jr+mt(i)}$$

a une unique solution $(a_j)_{0\leq j<m} \in \mathbb{F}_q^m$ et

$$P(X) = \sum_{j=0}^{m-1} a_j X^j$$

est un polynôme de permutation de $\mathbb{F}_q$.

Preuve : Il s'agit d'une traduction du théorème précédent. □

Remarque 3.31. On peut remarquer que les polynômes de permutation du type s'écrivant sous la forme $X^r P(X^{(q-1)/m})$ forment en fait un sous groupe du groupe des polynômes de permutation. De plus, les polynômes de ce type avec $r=1$ en sont un sous groupe. La structure de ces groupes est étudiée dans [WL91].

A l'aide de ce théorème, en utilisant des résultats sur le nombre de solutions d'un système d'équations sur un corps fini, on peut montrer le résultat suivant :

Corollaire 3.32 (Lidl, Wan [WL91]). *Soit m un diviseur de $q-1$ et $P \in \mathbb{F}_q[X]$, alors pour q suffisamment grand, il existe $a \in \mathbb{F}_q$ tel que $X^r(P(X^{(q-1)/m})+a)^k$ soit un polynôme de permutation de $\mathbb{F}_q$ pour tout $k \in \mathbb{N}^\star$ et pour tout $r \in \mathbb{N}$ tel que* $\gcd(r, q-1)=1$.

Nous verrons des versions plus précises de ce théorème dans le chapitre 4, en particulier un résultat quantitatif donné dans [LC07b]. Cela assure l'existence de polynômes de permutation, mais ne donne aucune classe infinie explicitement.

Revenons au théorème 3.29. En choisissant soigneusement les coefficients de notre polynôme dans un sous corps, et en prenant une extension bien choisie, on peut aussi utiliser ce résultat pour donner une famille explicite de binômes de permutation. Plus précisement, on peut énoncer le théorème suivant, qui a été publié dans [LC07b] :

Théorème 3.33. *Soient $k \in \mathbb{N}^\star$, p un nombre premier et m un diviseur de p^k-1. On pose $q=p^{km}$.*
Soit r premier avec $(q-1)/m$ et $P \in \mathbb{F}_{p^k}[X]$ ne s'annulant pas sur l'ensemble des racines m-ièmes de l'unité.
Alors le polynôme

$$X^r P\left(X^{\frac{q-1}{m}}\right)$$

est un polynôme de permutation sur $\mathbb{F}_q$.

Preuve : On utilise le théorème 3.29 ; ω désignera donc une racine m-ième primitive de l'unité de $\mathbb{F}_q$.

Les hypothèses (i) et (ii) du théorème 3.29 sont en fait reprises dans notre énoncé. Il ne reste donc que (iii) à étudier.

Il est clair que

$$\frac{q-1}{m} = \frac{p^k-1}{m}\sum_{j=0}^{m-1} p^{kj}$$

On a donc

$$\omega^{ri} P(\omega^i)^{\frac{q-1}{m}} = \omega^{ri}\left(\prod_{j=0}^{m-1} P(\omega^i)^{p^{kj}}\right)^{\frac{p^k-1}{m}}$$

Or ω et les coefficients de P sont dans $\mathbb{F}_{p^k}$, on a donc

$$\begin{aligned}\omega^{ri} P(\omega^i)^{\frac{q-1}{m}} &= \omega^{ri}\left(\prod_{j=0}^{m-1} P(\omega^i)\right)^{\frac{p^k-1}{m}} \\ &= \omega^{ri} P(\omega^i)^{p^k-1} \\ &= \omega^{ri}\end{aligned}$$

Enfin, r étant premier à $q-1$ il est premier avec p^k-1 qui est un diviseur de $q-1$. Les valeurs ω^{ri} sont donc bien distinctes. Cela prouve le point (iii) et termine la preuve. □

Remarque 3.34. Si l'on choisit k comme un produit de deux nombres $k = k_1k_2$, et $m = p^{k_1}-1$, les conditions sont encore plus simples à vérifier puisque les racines m-ièmes de l'unité constituent en fait le sous-corps $\mathbb{F}_{p^{k_1}}$. On peut par exemple choisir un polynôme irréductible pour s'assurer qu'il n'a pas de racine.

Exemple 3.35. Avec $p = 2$, $k = 6$ et $m = 3$, on obtient $q = 2^{18}$. En prenant $P = X^2 + aX + b$ un polynôme irréductible de $\mathbb{F}_{2^6}[X]$, on est assuré que

$$X^{29}(P(X^{\frac{q-1}{3}})) = X^{174791} + aX^{87410} + bX^{29}$$

est un polynôme de permutation de $\mathbb{F}_{2^{18}}$.

3.5.1 Les binômes

Comme nous l'avions annoncé dans la section précédente, ces résultats donnent en particulier un certain nombre d'exemples de familles de binômes de permutation.

Si l'étude des polynômes de permutation a attiré l'attention de nombreux auteurs [Car62, NR82, Tur88, Sma90, Wan94, Wan02, MZar, LC07b], il semble qu'au fil du temps tous les résultats obtenus se sont trouvés être des cas particuliers de familles plus larges, et en particulier du type précédent.

Si l'on spécialise le théorème 3.33 en prenant $P(X)$ de la forme $(X^e + a)$, on obtient clairement des binômes. Regardons plus précisement le corollaire [LC07b].

Corollaire 3.36. *Soient $k \in \mathbb{N}^\star$, p un nombre premier et m un diviseur de $p^k - 1$. On pose $q = p^{km}$.*
Soit r premier avec $(q-1)/m$ et $a \in \mathbb{F}^\star_{p^{km}}$ tel que $a^m \neq 1$.
Alors le polynôme

$$X^r(X^{\frac{q-1}{m}} - a)$$

est un polynôme de permutation sur $\mathbb{F}_q$.

Exemple 3.37.
Prenons $p = k = \ell = 2$. Le corps ambiant est donc $\mathbb{F}_{2^{2\cdot 2(2^2-1)}} = \mathbb{F}_{2^{12}}$, et on regarde les polynômes du type $XP(X^{\frac{2^{12}-1}{3}})$ c'est-à-dire $XP(X^{1365})$.
Soit α un élément primitif de $\mathbb{F}_{2^{12}}$. Alors $u^{11\frac{2^{12}-1}{2^4-1}} = u^{3003} \in \mathbb{F}_{2^4} \setminus \mathbb{F}_{2^2}$. Le polynôme

$$P(X) = X(X^{1365} + u^{3003})$$

est donc un polynôme de permutation, et le calcul avec `Magma` du polynôme représentant la permutation réciproque nous donne :

$$Q(X) = X(u^{273}X^{2730} + u^{3276}X^{1365} + u^{2184})$$

Avec $k = 1$, on obtient en fait exactement la famille décrite dans [AW06], mais avec une caractérisation beaucoup plus simple, sans recourir à la période de la suite

$$a_n = \sum_{t=1}^{(\ell-1)/2} \left(2\cos\frac{\pi(2t+1)}{\ell}\right)^n.$$

L'équivalence des deux formulations est détaillée dans [Zieb].

De la même manière, Pascale Charpin et Gohar Kyureghyan [CK08] ont caractérisé une famille de binômes de permutation impliquée dans la preuve d'un résultat concernant les fonctions courbes que nous reverrons plus tard. Leur résultat est le suivant.

Théorème 3.38. *Soient $0 \leq \ell < t$ et $\nu \neq 0$. Le polynôme*

$$X^{2^\ell+2} + \nu X \text{ dans } \mathbb{F}_{2^t}[X]$$

est un polynôme de permutation de $\mathbb{F}_{2^t}$ si et seulement si t est pair et

- *soit $\ell = 1$ et ν n'est pas une puissance troisième dans $\mathbb{F}_{2^t}$,*
- *soit $t = 2\ell$, $\ell \geq 3$ avec ℓ impair et $\nu \in \omega\mathbb{F}_{2^\ell}$ où $\omega \in \mathbb{F}_4 \setminus \mathbb{F}_2$.*

Une fois encore, on remarque que la famille obtenue est en fait

$$X(X^{\frac{2^{2\ell}-1}{2^\ell-1}} + \nu),$$

et découle donc du corollaire, avec $p = 2$ et $k = 1$. L'intérêt du résultat est donc de démontrer que c'est une condition nécessaire.

Au final, il semble donc que d'un point de vue théorique, le nombre de termes dans le polynôme ne soit pas le bon critère de classification, même s'il présente un intérêt certain pour les applications pratiques. Le chapitre suivant sur la répartition des polynômes de permutation va apporter encore quelques évidences de ce fait.

3.6 Les polynômes de Dickson

Il s'agit d'une classe de polynômes très étudiée et possédant de nombreuses propriétés. Nous nous contenterons ici des résultats en lien direct avec les polynômes de permutation.

La littérature les concernant est importante [Fri70, Wil71] mais nous ne donnerons ici que les résultats les plus importants.

Pour le reste, on pourra consulter sur le sujet le livre référence de Lidl, Mullen et Turnwald [LMT93].

Définition 3.39. Le n-ième polynôme de Dickson de paramètre a est donné par :

$$D_n(X,a) = \sum_{i=0}^{\lfloor n/2 \rfloor} \frac{n}{n-i}\binom{n-i}{n}(-a)^i X^{n-2i}$$

Lemme 3.40.

$$D_n(u+a/u,a) = u^n + (a/u)^n$$

Proposition 3.41. *Si $a \in \mathbb{F}_q^\star$, le polynôme $D_n(X,a)$ est de permutation sur $\mathbb{F}_q$ si et seulement si* $\operatorname{pgcd}(n, q^2-1) = 1$.

Remarque 3.42. Le cas $a = 0$ est traité au théorème 1.3 puisque $D_n(X,0) = X^n$.

On peut montrer que l'ensemble $\{D_n(X,a) | \operatorname{pgcd}(n,q^2-1) = 1\}$ est stable par composition si et seulement si $a \in \{0,1,-1\}$.

Ces polynômes ont été proposés pour créer un système cryptographique à clé publique, on pourra à ce sujet consulter [MN81, LM84b, Nöb88, Pie93, BBL95, XLP99, SZP02].

Concernant la structure des cycles des permutations obtenues on pourra consulter [LM91].

On pourrait aussi regarder si les polynômes de Dickson ne donnent pas d'exemples supplémentaires de polynômes de permutation quadratiques. Malheureusement, le degré algébrique de ceux-ci augmente rapidement. On démontre facilement le résultat suivant :

Proposition 3.43. *Les seuls polynômes de Dickson de première espèce $D_k(X,a)$ quadratiques sur $\mathbb{F}_{2^n}$ sont obtenus pour k valant 3.2^ℓ ou 5.2^ℓ.*

Il existe aussi des polynômes de Dickson dits de deuxième espèce dont l'étude est plus compliquée. On pourra consulter [Coh94, HM95, HM98, Mat82].

3.7 o-polynôme et arcs dans $PG(2,2^n)$

Nous allons présenter brièvement les définitions nécessaires pour introduire des classes de polynômes provenant de la géométrie.

Définition 3.44. Un n-arc de $PG(N,q)$, l'espace projectif de dimension N sur $\mathbb{F}_q$, est un ensemble de n points tels que pour tout $(N+1)$-uplet de cet ensemble les points sont linéairement indépendants.

Notons $m(N, q)$ le cardinal maximal d'un n-arc.

On montre que $m(2, q)$ vaut $(q + 1)$ si q est impair et vaut $(q + 2)$ si q est pair. On appelle *ovoïde* un $(q + 1)$-arc de $PG(2, q)$ pour q impair et *hyperoval* un $(q + 2)$-arc de $PG(2, q)$ pour q pair.

Dans le cas de $q = 2^n$, les hyperovales sont liés aux polynômes de permutation par le théorème suivant.

Définition 3.45. Un polynôme de permutation P tel que pour tout $a \in \mathbb{F}_q$ le polynôme $(P(X + a) + P(a))/X$ est un polynôme de permutation de $\mathbb{F}_q$ est appelé un *o-polynôme*.

Théorème 3.46. *Tout hyperoval de $PG(2, 2^h)$ avec $h > 1$ est projectivement équivalent à un hyperovale du type $\mathcal{D}(P) = \{(1, x, P(x)); x \in \mathbb{F}_q\} \cup \{e_1, e_2\}$ où P est un o-polynôme.*

3.7.1 À propos d'un résultat de Payne

Le premier exemple d'o-polynômes est tout simplement les monômes X^{2^k} dans le cas où $\text{pgcd}(k, n) = 1$.

Il est clair que ce sont des o-polynômes linéaires et on peut se demander s'il en existe d'autres.

On remarque que si $P(X)$ est linéaire, alors pour tout $a \in \mathbb{F}_q$,

$$(P(X + a) + P(a))/X = P(X)/X.$$

Trouver un o-polynôme linéaire revient donc à trouver une permutation linéaire $L(X)$ telle que $L(X)/X$ soit aussi une permutation.

La caractérisation des ovals de ce type est ancienne. En fait, on peut prouver qu'il n'y en a pas d'autres.

Théorème 3.47 (Payne [Pay71])**.** *Il n'existe pas d'ovoïdes de translations de $PG(2, n)$ non triviaux.*

On peut réécrire cela en terme de polynômes de permutation de la manière suivante :

Théorème 3.48. *Soit*

$$Q(X) = \sum_{i=1}^{n} c_i X^{2^i - 1}, c_i \in \mathbb{F}_{2^n}$$

un polynôme de permutation. Alors $Q(X) = cX^{2^k - 1}$ avec $\text{pgcd}(k, n) = 1$ et $c \in \mathbb{F}_{2^n}^\star$.

Nous allons donner ici une preuve de ce résultat, formulée uniquement en termes de polynômes.

Pour cela, nous aurons besoin des deux lemmes suivants.

Lemme 3.49. *Soit H un polynôme de $\mathbb{F}_{2^n}$ tel que $H(0) = 0$ et vérifiant*

$$\forall a, \forall b, a \neq b \neq 0, H(a) \neq H(b),$$

et $H(e) = 0$ pour un unique e non nul. Alors le degré de H est exactement $2^n - 1$.

Preuve : Puisque $H(e) = H(0) = 0$, H n'est pas une permutation. L'image de H contient exactement $2^n - 1$ éléments dont 0 qui apparaît deux fois. Il existe donc un élément non nul β qui n'est pas dans $H(\mathbb{F}_{2^n})$.

Soit P le polynôme de $\mathbb{F}_{2^n}$ défini par

$$P(X) = \begin{cases} H(X) \text{ si } x \neq e \\ \beta \text{ si } x = e. \end{cases}$$

P est un polynôme de permutation.

De plus, on vérifie que

$$P(X) = H(X) + \beta \left((x+e)^{2^n-1} + 1 \right).$$

Or d'après le critère de Hermite (théorème 2.1), P est de degré au plus $2^n - 2$. Le terme βx^{2^n-1} doit donc appartenir à H qui est donc de degré $2^n - 1$. □

Lemme 3.50. *Soient n et μ deux entiers avec $0 \leq \mu < n-1$ et soient $(u_0, \ldots, u_\mu)$ et $(a_0, \ldots, a_\mu)$ deux $(\mu+1)$-uplets d'éléments de l'intervalle $[1, n-1]$, avec de plus les u_i deux à deux distincts.*

Soient u et v deux entiers définis par

$$u = \sum_{i=0}^{\mu} 2^{u_i} \text{ et } v = \sum_{i=0}^{\mu} 2^{a_i+u_i}.$$

Alors,

$$x^{v-u} = x^{2^n-1} \pmod{x^{2^n} + x} \text{ dans } \mathbb{F}_{2^n}[x]$$

si et seulement si

$$\{a_i + u_i \pmod{n}, i \in [0, \mu]\} = \{u_i, i \in [0, \mu]\}.$$

Preuve : Puisque $v > u$, la congruence $x^{v-u} = x^{2^n-1} \pmod{x^{2^n} + x}$ est équivalente à

$$x^v = x^u \pmod{x^{2^n} + x}.$$

Mais on sait que

$$x^v = \prod_{i-0}^{\mu} x^{2^{a_i+u_i} \pmod{n}} = x^{v'} \pmod{x^{2^n} + x}$$

avec $v' = \sum_{i=0}^{\mu} 2^{a_i+u_i \pmod{n}}$. Ainsi on a bien l'égalité $x^v = x^u \pmod{x^{2^n} + x}$ si et seulement si

$$\{a_i + u_i \pmod{n}, i \in [0, \mu]\} \text{ et } \{u_i, i \in [0, \mu]\}$$

sont égaux. En effet, ces deux ensembles doivent avoir le même cardinal pour que la congruence soit vérifiée. □

Nous voilà prêts pour la démonstration annoncée.

Preuve : [du théorème 3.48]

Soit $k \in [1, n-1]$ tel que $c_k \neq 0$. Soit $r_k = n-1$ si $\text{pgcd}(k,n) = 1$ et $n/\text{pgcd}(k,n)$ sinon.

Nous allons prouver par récurrence sur μ que

$$c_{-\mu k} = 0 \text{ pour tout } \mu \in [0, r_k - 1], \tag{3.3}$$

où tout les indices sont pris modulo n.

Pour $\mu = 0 \pmod n$, c'est-à-dire $\mu = n$, il suffit d'appliquer la propriété 3.49. Le degré de Q doit être inférieur strictement à $2^n - 1$, donc $c_n = 0$ et la propriété est vérifiée.

Soit $\mu < r_k - 1$. Supposons que $c_{-\mu k} = 0$ pout tout μ' avec $0 \leq \mu' \leq \mu - 1$.

Soit $u = \sum_{j=0}^{\mu} 2^{kj} \pmod n$. On a alors

$$\begin{aligned}
Q^u(X) &= \left(\sum_{i=1}^{n-1} c_i x^{2^i-1}\right)^{\sum_{j=0}^{\mu} 2^{kj}} \\
&= \prod_{j=0}^{\mu} \left(\sum_{i=1}^{n-1} c_i x^{2^i-1}\right)^{2^{kj}} \\
&= \prod_{j=0}^{\mu} \left(\sum_{i=1}^{n-1} c_i^{2^{kj}} x^{2^{kj+i}-2^{kj}}\right) \\
&= \sum_{a \in [1,n-1]^{\mu+1}} \left(\prod_{j=0}^{\mu} c_{a_j}^{2^{k_j}}\right) m_a(x)
\end{aligned}$$

avec

$$m_a(x) = x^{\sum_{j=0}^{\mu} 2^{k_j + a_j} - 2^{kj}}$$

et $a = (a_0, \ldots, a_\mu), a_j \in [1, n-1]$.

En utilisant le lemme 3.50, on en déduit que les valeurs de a pour lesquelles m_a est de degré $2^n - 1$ sont celles telles que

$$\{a_j + k_j \pmod n, j \in [0, \mu]\} = \{kj \pmod n, j \in [0, \mu]\}$$

ou de manière équivalente telles qu'il existe une permutation σ de $[0, \mu]$ telle que

$$a_j = k(\sigma(j) - j) \pmod n, \text{ pour tout } j \in [0\mu].$$

Puisque le monôme m_a a pour coefficient

$$b_\sigma = \prod_{j=0}^{\mu} c_{k(\sigma(j)-j)}^{2^{kj}},$$

on en déduit que le coefficient du terme de degré $2^n - 1$ de Q^u est

$$\sum_{\sigma \in \mathcal{S}'_\mu} b_\sigma = \sum_{\sigma \in \mathcal{S}'_\mu} \prod_{j=0}^{\mu} c_{k(\sigma(j)-j)}^{2^{kj}},$$

où $\mathcal{S}'_\mu$ désigne l'ensemble des permutations de $[0, \mu]$ telles que $k(\sigma(j) - j) \neq 0 \pmod n$ pour tout j. Clairement, σ étant une permutation on a $\sum_{j=0}^{\mu}(\sigma(j) - j) = 0$. Comme de plus $\sigma(j) \neq j$ pour tout $j \in [0, \mu]$, il existe au moins une valeur $j \in [0, \mu]$ telle que $\sigma(j) - j < 0$.

S'il existe j tel que $-(\mu - 1) \leq \sigma(j) - j < 0$, le coefficient b_σ est un multiple de $c_{-k(j-\sigma(j))}$, qui est nul d'après l'hypothèse de récurrence.

Sinon, la seule valeur j pour laquelle $\sigma(j) - j < 0$ satisfait $\sigma(j) - j \leq -\mu$. La permutation σ est alors nécessairement définie par

$$\begin{array}{cccccccc} j & = & 0 & 1 & 2 & \dots & \mu - 1 & \mu \\ \sigma(j) & = & 1 & 2 & 3 & \dots & \mu & 0 \end{array}$$

De plus, cette permutation σ appartient bien à $\mathcal{S}'_\mu$ car $k(\sigma(j) - j) \neq 0 \pmod n$ comme $\mu < n/\operatorname{pgcd}(k, n)$.

Le coefficient du terme de degré $2^n - 1$ de Q^u est donc

$$\prod_{j=0}^{\mu-1} c_k^{2^{kj}} c_{-\mu k}^{2^{ku}}.$$

Puisque u est pair, le critère de Hermite 2.1 implique que $c_{-\mu k} = 0$ si Q est un polynôme de permutation.

Par récurrence, la propriété est donc vraie pour tout $\mu \in [1, r_k - 1]$.

Enfin,

– si $\operatorname{pgcd}(n, k) = 1$, on a

$$\{-\mu k \pmod n, \mu \in [1, n-2]\} = \{1, \dots, n-1\} \setminus \{k\}.$$

On a donc $Q(x) = c_k x^{2^k - 1}$;

– si $\operatorname{pgcd}(n, k) = d > 1$, nous venons de prouver que $c_{-(r_k-1)k} = 0$ où $n = r_k \operatorname{pgcd}(k, n)$. Ainsi, $-k(r_k - 1) = k \pmod n$, et par conséquent $c_k = 0$ ce qui est une contradiction.

□

3.7.2 Les autres familles

Les familles infinies connues d'o-polynômes sont résumées dans le tableau 3.2.

Avec

$$S(X) = \frac{\delta^2(X^4 + X) + \delta^2(1 + \delta + \delta^2)(X^3 + X^2)}{X^4 + \delta^2 X^2 + 1} + X^{1/2}$$

et en notant $K = \mathbb{F}_{q^2}$, $F = \mathbb{F}_q$ et $tr(x) = \operatorname{Tr}_q^{q^2}(x) = x + x^q$,

$$A(X) = \frac{1}{tr(b)}(tr(b^m)(X + 1) + tr((bX + b^q)^m)(X + tr(b)X^{1/2} + 1)^{1-m}) + X^{1/2}$$

Les polynômes de ce type sont très rares. Les polynômes $S(X)$ appelés polynômes de Subiaco ont été mis en évidence en 1995.Les polynômes $A(X)$ sont appelés polynômes d'Adelaïde. Ce sont les derniers en date à avoir été exhibés, la preuve datant de 1999. Une référence régulièrement mise à jour à propos des hyperovals se trouve sur la page de Cherowitzo [Che].

$\mathbf{q = 2^h}$	$P(X)$	**Conditions**
	X^{2^k}	$\text{pgcd}(k, n) = 1$
h impair	X^6	
h impair	$X^{3\sigma+4}$	$\sigma = 2^{(h+1)/2}$
h impair	$X^{\sigma+\lambda}$	$\sigma = 2^{(h+1)/2}$ $\lambda = 2^m$ si et seulement si $h = 4m - 1$ et $\lambda = 2^{3m+1}$ si et seulement si $h = 4m + 1$.
h impair	$X^{1/6} + X^{3/6} + X^{5/6}$	
h impair	$X^\sigma + X^{\sigma+2} + X^{3\sigma+4}$	$\sigma = 2^{(h+1)/2}$
	$S(X)$	$\text{Tr}_1^h(1/\delta) = 1$ et de plus $\delta \notin \mathbb{F}_4$ si et seulement si $h = 4m + 2$
h pair	$A(X)$	$m = \pm\frac{q-1}{3}$ et $b \in \mathbb{F}_{q^2}$ tel que $b^{q+1} = 1$ et $b \neq 1$.

TAB. 3.2 – Familles d'o-polynômes

3.8 Les polynômes exceptionnels

Une classe très importante sur le plan théorique est celle des polynômes exceptionnels, c'est-à-dire comme nous l'avons vu des polynômes qui sont de permutation sur une infinité d'extensions d'un corps.

Les premiers exemples nous sont biens connus puisqu'il s'agit
- des permutations puissances X^k (voir 1.3) ;
- des permutations linéarisées $L(X) = \sum a_i X^{q^i}$ (voir 1.6) ;
- des permutations de Dickson de première espèce $D_k(X, a)$ (voir 3.39) ;
- les compositions de ces polynômes, par exemple les polynômes sous-linéarisés

$$S(X) = X^k \circ L(X) \circ X^{-k}.$$

Ces résultats ont été établis par Dickson, puis durant un siècle aucun autre exemple n'est apparu.

Les travaux de Fried, Guralnick et Saxl [FGS93] ont ensuite grandement relancé l'intérêt pour ce sujet en énumérant les groupes de monodromie possibles pour ces polynômes. On a alors cherché à établir une classifiation complète de ces polynômes, et ce travail est aujourd'hui très avancé.

Comme la composition de deux polynômes exceptionnels est aussi un polynôme exceptionnels, les efforts de classification se sont concentrés sur les polynômes *indécomposables*, c'est-à-dire ne pouvant pas s'écrire $P = A \circ B$ avec A et B deux polynômes de degré inférieur à P.

À l'heure actuelle les, exemples de polynômes séparables, indécomposables et exceptionnels connus sont, en plus des exemples cités ci-dessus :
- soient $q = 2^e > 2$ avec e impair et k tel que $\mathbb{F}_{2^k} \cap \mathbb{F}_q = \mathbb{F}_2$,

$$\left(\frac{\text{Tr}_1^e(X) + \alpha}{X}\right)^q \left(\text{Tr}_1^e(X) + \frac{\text{Tr}_1^e(X) + \alpha}{\alpha + 1} \cdot \text{Tr}_1^e\left(\frac{X(\alpha^2 + \alpha)}{(\text{Tr}_1^e(X) + \alpha)^2}\right)\right)$$

avec $\text{Tr}_1^e(X) = X + X^2 + \cdots + X^{2^{e-1}}$.

– soient $q = 2^e > 2$ avec e impair et k tel que $\mathbb{F}_{2^k} \cap \mathbb{F}_q = \mathbb{F}_2$,

$$X \left(\sum_{i=0}^{e-1} (\alpha X^n)^{2^i - 1} \right)^{(q+1)/n}$$

où n divise $2^e + 1$ et $\alpha \in \mathbb{F}_q^\star$.

– soient $q = 3^e > 3$ avec e impair et k tel que $\mathbb{F}_{3^k} \cap \mathbb{F}_q = \mathbb{F}_3$,

$$X \left(X^{2n} - \alpha\right)^{(q+1)/(4n)} \left(\frac{\left(X^{2n} - \alpha\right)^{(q-1)/2} + \alpha^{(q-1)/2}}{X^{2n}} \right)^{(q+1)/(2n)}$$

où n divise $(q+1)/4$ et $\alpha \in \mathbb{F}_q^\star$.

On pourra en particulier consulter les articles de Michael Zieve [GZ, GRZ, LJZ96].

Guralnick, Zieve et Müller ont aussi un article en préparation [GMZ] présentant des avancées dans un des derniers cas manquant, mais il n'est pas encore disponible à l'heure de la rédaction de ce document.

Chapitre 4

Répartition des polynômes de permutation

Pour conclure la partie théorique de cette première partie, nous allons voir comment les polynômes de permutation sont répartis selon différents critères, de degré ou de nombre de coefficients par exemple.

4.1 Polynômes de formes prescrites

Dans [LC07b], nous avons proposé une méthode pour estimer le nombre de binômes de permutation de la forme

$$X^r \left(X^{\frac{q-1}{\ell}} + a \right).$$

Pour ce faire, nous utilisons un théorème sur les sommes de caractères (énoncé plus tard 4.5).

On obtient alors le théorème suivant :

Théorème 4.1. *Soient q une puissance d'un nombre premier, ℓ un diviseur de $q-1$ et r un nombre premier avec $q-1$. On définit $N_r(\ell, q)$ le nombre de polynômes de permutation de la forme $X^r \left(X^{\frac{q-1}{\ell}} + a \right)$ avec $a \in \mathbb{F}_q^\star$. On a alors l'inégalité suivante :*

$$\left| N_r(\ell, q) - \frac{\ell!}{\ell^\ell} q \right| \leqslant \ell! \left(\frac{1}{\ell^\ell} + (\ell - 2) \right) \sqrt{q} + (\ell + 1)!$$

Il est à noter que cet encadrement n'est pas optimal et on pourrait améliorer légèrement les constantes.

Ainsi, la proportion de binômes obtenue est d'autant plus grande que le diviseur ℓ considéré est petit. Cela peut se voir sur les graphiques suivant, représentant le nombre de valeurs a telles que $X^i + aX^j = X^j(X^{i-j} + a)$ soit un polynôme de permutation (un nombre élevé se traduisant par un point clair), respectivement sur $\mathbb{F}_{81}$ et $\mathbb{F}_{64}$. On remarque en effet des lignes diagonales correspondant pour $\mathbb{F}_{64}$ à $i - j = 21$ et donc $\ell = 3$, et pour $\mathbb{F}_{81}$ à $i - j = 40$ et donc $\ell = 2$.

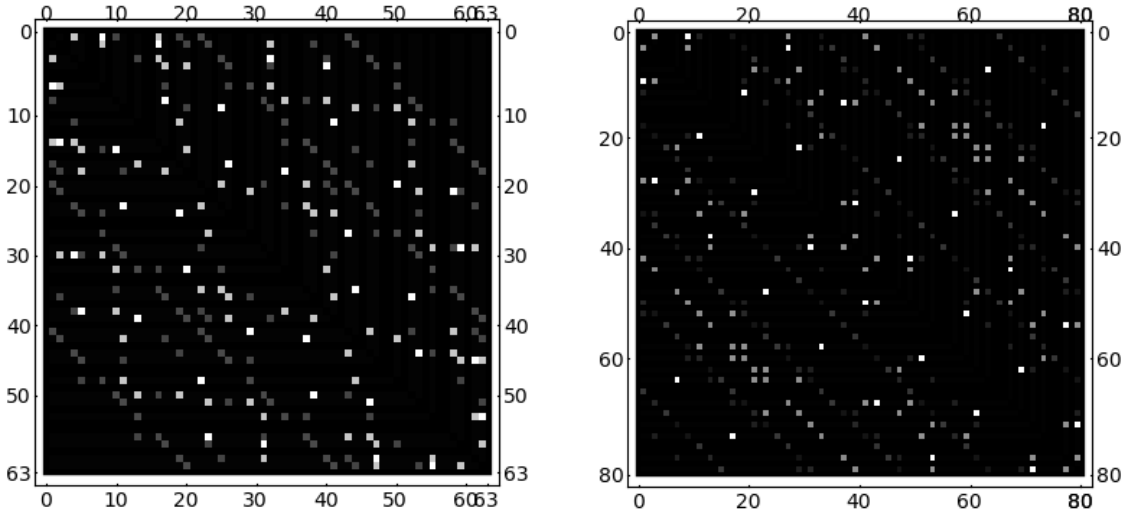

FIG. 4.1 – Répartition des binômes de permutation dans $\mathbb{F}_{64}$ et $\mathbb{F}_{81}$

Cette approche a été généralisée dans [AGW08] pour obtenir le résultat suivant.

Notation : Soient q une puissance d'un nombre premier, $\ell > 1$ un diviseur de $q-1$ et $s = (q-1)/\ell$. Soient aussi m, r deux entiers et $e = (e_1, \ldots, e_m)$ tels que

$$0 < e_1 < \cdots < e_m < \ell, \ \text{pgcd}(e_1, \ldots, e_m, \ell) = 1 \text{ et } r + e_m s < q.$$

Alors $\boldsymbol{N^m_{r,e}(\ell, q)}$ est le nombre de m-uplets $a \in (\mathbb{F}_q^\star)^m$ tels que

$$x^r \left(x^{e_m s} + a_1 x^{e_{m-1} s} + \cdots + a_{m-1} x^{e_1 s} + a_m\right)$$

est un polynôme de permutation.

Théorème 4.2. *Avec les notations précédentes,*

$$\left| N^m_{r,e}(\ell, q) - \frac{\ell!}{\ell^\ell} q^m \right| < \ell! \ell q^{m - \frac{1}{2}}$$

On remarque que le théorème précédent correspond au cas $N^1_{r,(1)}(\ell, q)$

La preuve suit la méthode que nous avons proposée dans [LC07b] et est assez technique. Nous allons avoir besoin des définitions supplémentaires suivantes.

Soient $\mathcal{S}_\ell$ l'ensemble des permutation de $\{1, \ldots, \ell\}$ et μ_ℓ l'ensemble des racines ℓ-ièmes de l'unité de $\mathbb{F}_q$.

Soient α un générateur du groupe cyclique $\mathbb{F}_q^\star$ et ω une racine ℓ-ième de l'unité primitive dans $\mathbb{C}$.

Soit ψ le caractère multiplicatif d'ordre ℓ de μ_ℓ défini par $\psi(\alpha^s) = \omega$, que l'on étend par $\psi(0) = 0$. Enfin soit Ψ le caractère multiplicatif de $\mathbb{F}_q$ défini par $\Psi(\alpha) = \psi(\alpha^s)$ et étendu par $\Psi(0) = 0$.

Pour toute permutation $\sigma \in \mathcal{S}_\ell$, on définit

$$P_\sigma(\beta_1, \ldots, \beta_\ell) = \prod_{i=1}^{\ell} \left(\sum_{j=0}^{\ell-1} \left(\psi(\beta_i) \psi(\alpha^s)^{-\sigma(i)} \right)^j \right).$$

On vérifie que

$$\{\beta_1, \ldots, \beta_\ell\} = \mu_\ell \Leftrightarrow \exists \sigma \in \mathcal{S}_\ell ; P_\sigma(\beta_1, \ldots, \beta_\ell) = \ell^\ell. \tag{4.1}$$

De plus, cette permutation σ est alors unique.

Lemme 4.3. *Soient $\beta_1, \ldots, \beta_\ell \in \mu_\ell$. Alors,*

$$\frac{1}{\ell^\ell} \sum_{\sigma \in \mathcal{S}_\ell} P_\sigma(\beta_1, \ldots, \beta_\ell) = \begin{cases} 1 \text{ si } \{\beta_1, \ldots, \beta_\ell\} = \mu_\ell \\ 0 \text{ sinon.} \end{cases}$$

Lemme 4.4. *Soient $\beta_1, \ldots, \beta_\ell \in \mu_\ell \cup \{0\}$ avec au moins un des β_i nul. Alors,*

$$0 \leq \frac{1}{\ell^\ell} \sum_{\sigma \in \mathcal{S}_\ell} P_\sigma(\beta_1, \ldots, \beta_\ell) \leq \frac{1}{\ell}$$

Enfin, nous aurons besoin de ce résultat classique sur les sommes de caractères (voir [LN97] Th.5.41).

Théorème 4.5. *Soit Ψ un caractère multiplicatif de $\mathbb{F}_q$ d'ordre $m > 1$, et $P \in \mathbb{F}_q[X]$ un polynôme qui n'est pas une puissance m-ième. Soit d le nombre de racines distinctes de P dans son corps de décomposition. Alors pour tout x de $\mathbb{F}_q$,*

$$\left| \sum_{a \in \mathbb{F}_q} \Psi(xP(a)) \right| \leqslant (d-1)\sqrt{q}.$$

Armés de tout cela, nous pouvons maintenant détailler la preuve de notre théorème.

Preuve : [du théorème 4.2] Notons

$$g_a = x^r \left(x^{e_m s} + a_1 x^{e_{m-1} s} + \cdots + a_{m-1} x^{e_1 s} + a_m \right).$$

D'après le théorème 3.29, g_a est un polynôme de permutation si et seulement si les deux conditions suivantes sont satisfaites.

(i) Pour tout $i \in \{1, \ldots, \ell\}$, $\alpha^{e_m s} + a_1 \alpha^{e_{m-1} s} + \cdots + a_{m-1} \alpha^{e_1 s} + a_m \neq 0$.

(ii) $g_a(\alpha^i)^s \neq g_a(\alpha^j)^s$, pour $1 \leq i < j \leq \ell$.

En utilisant le lemme 4.3, on obtient alors

$$N_{r,e}^m(\ell, q) = \frac{1}{\ell^\ell} \sum_{\substack{a \in (\mathbb{F}_q^\star)^m \\ a \text{ vérifie } (i)}} \sum_{\sigma \in \mathcal{S}_\ell} P_\sigma \left(g_a(\alpha^1)^s, \ldots, g_a(\alpha^\ell)^s \right). \tag{4.2}$$

Afin d'alléger les notations, nous noterons

$$P_\sigma^a = P_\sigma \left(g_a(\alpha^1)^s, \ldots, g_a(\alpha^\ell)^s \right), \text{ et } S(\mathcal{E}) = \frac{1}{\ell^\ell} \sum_{a \in \mathcal{E}} \sum_{\sigma \in \mathcal{S}_\ell} P_\sigma^a$$

En utilisant le principe d'inclusion-exclusion, on remarque que le nombre de m-uplets $(a_1, \ldots, a_m) \in (\mathbb{F}_q^\star)^m$ tels que pour au moins un i,

$$\alpha^{e_m i s} + a_1 \alpha^{e_{m-1} i s} + \cdots + a_{m-1} \alpha^{e_1 i s} + a_m = 0,$$

est au plus

$$\ell \sum_{j=0}^{m-1} (-1)^j \binom{m}{j} q^{m-j-1} = \ell \frac{(q-1)^m - (-1)^m}{q}.$$

De plus, il y a $q^m - (q-1)^m$ m-uplets $(a_1, \ldots, a_m) \in (\mathbb{F}_q^\star)^m$ tels que l'un des a_i au moins vaut 0.

En utilisant dans l'équation (4.2) le lemme 4.3 pour les m-uplets ne vérifiant pas la condition (i) et le lemme 4.4 pour ceux pour lesquels l'un des a_i est nul, on obtient alors

$$-\frac{q^{m+1} - (q-1)^{m+1} - (-1)^m}{q} \leq N_{r,e}^m(\ell, q) - S(\mathbb{F}_q^m) \leq 0. \tag{4.3}$$

Nous définissons alors

$$K(q, m) = \frac{q^{m+1} - (q-1)^{m+1} - (-1)^m}{q}$$

Nous allons maintenant évaluer la somme $S(\mathbb{F}_q^m)$. Soit $a = (a_1, \dots, a_m) \in \mathbb{F}_q^m$ et $k = (k_1, \dots, k_m) \in \{0, \ell - 1\}^\ell$, définissons

$$M_{a,\sigma,k} = \prod_{i=1}^{\ell} \left(\psi(g_a(\alpha^i)^s)\psi(\alpha^s)^{-\sigma(i)}\right)^{k_i}.$$

On a alors

$$P_\sigma^a = \sum_{k \in \{0,\ell-1\}^\ell} M_{a,\sigma,k}. \tag{4.4}$$

Soient $(a_1, \dots, a_{m-1})$ fixés et $\beta = \alpha^s$ un générateur de μ_ℓ. Notons

$$C(a, i) = \beta^{e_m i} + a_1\beta^{e_{m-1} i} + \cdots + a_{m-1}\beta^{e_1 i}.$$

En utilisant la multiplicativité de ψ, le produit $M_{a,\sigma,k}$ vaut

$$\psi\left(t^s \prod_{i=1}^{\ell} (C(a, i) + a_m)^{k_i s}\right), \tag{4.5}$$

avec $t(\sigma) = \alpha^{\sum_{i=1}^{\ell} k_i(ri - \sigma(i))}$.

En utilisant la définition de Ψ, ainsi que les équations (4.4) et (4.5), on peut exprimer P_σ^a par la somme de caractères suivante :

$$\sum_{k \in \{0,\ell-1\}^\ell} \Psi\left(t \prod_{i=1}^{\ell} (C(a, i) + a_m)^{k_i}\right). \tag{4.6}$$

Soit E l'ensemble des m-uplets $(a_1, \dots, a_{m-1}) \in \mathbb{F}_q^{m-1}$ tels que pour tout $i \neq j$, $C(a, i) \neq C(a, j)$.

Soit $I(k)$ le nombre de k_i non nuls. Si $I(k) > 0$ et si $(a_1, \dots, a_{m-1})$ est dans E, alors

$$a_m \mapsto t \prod_{i=1}^{\ell} (C(a, i) + a_m)^{k_i}$$

n'est pas une puissance ℓ-ième puisque ses racines sont exactement les $-C(a, i)$ avec multiplicité $k_i < \ell$. Ainsi, lorsque $(a_1, \dots, a_{m-1})$ est dans E nous pourrons appliquer le théorème de Weil bornant les sommes de caractères 4.5.

Commençons par voir que E constitue la majorité des m-uplets. En effet, Si $m = 1$, alors $\#E = 1$.

Si $m > 1$,

$$q^{m-2} \leq q^{m-1} - \#E \leq \binom{\ell}{2} q^{m-2}.$$

et donc

$$q^{m-2} \geq \#E \geq q^{m-1} - \binom{\ell}{2} q^{m-2}.$$

Il est clair que

$$S(\mathbb{F}_q^m) = S(\mathbb{F}_q^m \setminus E \times \mathbb{F}_q) + S(E \times \mathbb{F}_q).$$

En appliquant le lemme 4.3, on peut majorer

$$\left|S(\mathbb{F}_q^m \setminus E \times \mathbb{F}_q)\right| \leq q^m - q\#E \leq \binom{\ell}{2} q^{m-1} \tag{4.7}$$

Pour l'autre somme,

$$\begin{aligned} & S(E \times \mathbb{F}_q) \\ = & \frac{1}{\ell!} \sum_{\sigma \in \mathcal{S}_\ell} \sum_{(a_1,\ldots,a_{m-1}) \in E} \sum_{i=0}^{\ell} \sum_{\substack{k \in \{0,\ell-1\}^\ell \\ I(k)=i}} \sum_{a_m \in \mathbb{F}_q} \Psi\left(t \prod_{j=1}^{\ell} (C(a,j)+a_m)^{k_j}\right) \\ = & \frac{\ell! q\#E}{\ell^\ell} + \frac{1}{\ell!} \sum_{\sigma \in \mathcal{S}_\ell} \sum_{(a_1,\ldots,a_{m-1}) \in E} \sum_{i=1}^{\ell} \sum_{\substack{k \in \{0,\ell-1\}^\ell \\ I(k)=i}} \sum_{a_m \in \mathbb{F}_q} \Psi\left(t \prod_{j=1}^{\ell} (C(a,j)+a_m)^{k_j}\right), \end{aligned}$$

Le terme $\frac{\ell! q\#E}{\ell^\ell}$ correspondant à $I(k) = 0$. On peut alors appliquer le théorème 4.5 et obtenir finalement

$$\left|S(E \times \mathbb{F}_q) - \frac{\ell! q\#E}{\ell^\ell}\right| \leq \frac{\ell!}{\ell^\ell} \#E \left(\sum_{i=1}^{\ell} \sum_{\substack{k \in \{0,\ell-1\}^\ell \\ I(k)=i}} (i-1) q^{1/2} \right)$$

ce qui en calculant la somme donne

$$\left|S(E \times \mathbb{F}_q) - \frac{\ell! q\#E}{\ell^\ell}\right| \leq \frac{\ell!}{\ell^\ell} \#E \left(1 + \ell^\ell(\ell-2)\right) q^{1/2} \tag{4.8}$$

Puis en utilisant

$$\left|S(E \times \mathbb{F}_q) - \frac{\ell! q^m}{\ell^\ell}\right| \leq \frac{\ell!}{\ell^\ell} \Big(\left(q^{m-1} - \#E\right) + \#E \left(1 + \ell^\ell(\ell-2)\right) q^{1/2} \Big) \tag{4.9}$$

En regroupant les équations 4.9 et 4.7, on obtient

$$-\binom{\ell}{2} q^{m-2} \leq S(\mathbb{F}_q^m) - \frac{\ell!}{\ell^\ell} q^m \leq \binom{\ell}{2} q^{m-2}$$

En revenant à l'équation (4.3) on peut finalement écrire que

$$-K(q,m) \leq N_{r,e}^m(\ell,q) - S(\mathbb{F}_q^m \setminus E \times \mathbb{F}_q) - S(E \times \mathbb{F}_q) \leq 0,$$

ce qui donne en regroupant les résultats précédents,

$$-K(q,m) - \binom{\ell}{2} q^{m-2} - \frac{\ell!}{\ell^\ell} \#E \left(1 + \ell^\ell(\ell-2)\right) q^{1/2} \leq N_{r,e}^m(\ell,q) - \frac{\ell!}{\ell^\ell} q^m \leq \binom{\ell}{2} q^{m-2}.$$

Enfin, en revenant à la définition de K et à l'équation (4.7) on peut terminer notre encadrement. □

Dans le cas particulier des binômes, notre résultat a aussi été légérement amélioré par Ariane Masuda et Michael Zieve. Dans [MZar] ils donnent le résultat suivant.

Théorème 4.6. *Soit $n > m > 0$ deux entiers et $q = p^k \geq 4$ tel que $\gcd(m, n, q-1) = 1$. Si $\gcd(m-n, q-1) > 2q(\log\log q)/(\log q)$, alors il existe $a \in \mathbb{F}_q^\star$ tel que $X^n + aX^m$ soit un polynôme de permutation de $\mathbb{F}_q$.*

Ariane Masuda, dans une série d'article avec Daniel Panario, Qiang Wang et Michael Zieve a aussi dénombré les binômes de permutation de $\mathbb{F}_q$ lorsque $q = 4p + 1$ avec p et q premiers [MPW06, MZ07].

4.2 Polynômes évitant certains exposants

Le critère de Hermite 2.1 assure que le degré d'un polynôme de permutation ne peut pas être un diviseur de $q - 1$. Ainsi, le degré maximum possible est $q - 2$.

Konyagin et Pappalardi se sont intéressés au nombre de polynômes de permutation de degré donné.

Théorème 4.7 ([KP06])**.** *Soit $N_d(q)$ le nombre de polynômes de permutation de $\mathbb{F}_q$ de degré d.*

$$\left| N_{q-2}(q) - \frac{\phi(q)}{q} q! \right| \leqslant q^{q/2} \sqrt{\frac{2e}{\pi}}$$

Ils ont aussi montré le théorème suivant.

Théorème 4.8 ([KP02])**.** *Soit $N(k_1, \ldots, k_j; q)$ le nombres de polynômes de permutation dont les coefficients de x^{k_i} sont nuls pour $i = 1, \ldots, j$.*

$$\left| N(k_1, \ldots, k_j; q) - \frac{q!}{q^j} \right| < \left[\left(1 + \sqrt{\frac{1}{e}} \right) \sqrt{(q - k_1 - 1)q} \right]^q$$

Contrairement au théorème que nous avons démontré dans la section précédente, la forme du polynôme n'est ici pas vraiment spécifié puisque nous n'avons aucune information sur les coefficients ne correspondant pas à un des degrés interdits k_i ; en particuliers, ils peuvent être nuls.

4.3 Permutations complètes et différences de polynômes de permutation

Enfin, nous concluerons ce chapitre avec divers résultats montrant en quelque sorte "l'isolement" des polynômes de permutation. En effet, le problème de la différence entre deux polynômes de permutation a été posé dès 1968 par Chowla et Zassenhaus [ZC68]. Voici la version plus précise prouvée par Cohen [Coh90].

Théorème 4.9. *Soit $P(X)$ un polynôme à coefficients entiers de degré $n \geq 2$. Soit p un nombre premier, $p > (n^2 - 3n + 4)$, tel que $P(X) \in \mathbb{F}_p[X]$. Alors pour tout $a \in \mathbb{F}_p^\star$ le polynôme $P(X) + aX$ n'est pas de permutation.*

Les polynômes de permutation $P(X)$ tels que $P(X) + X$ soit aussi un polynôme de permutation ont une importance toute particulière et sont appelés des polynômes de permutation complets, ou orthomorphismes. Ils apparaissent dans divers problèmes combinatoires ou de codage. On peut citer par exemple l'article de Dai, Golomb et Gong s'intéressant à la génération de tous les polynômes linéaires de permutation complets [DGG99].

Chapitre 5

Applications en cryptographie

Considérer les permutations sous l'angle de leur représentation polynomiale permet de faire un lien entre la nature combinatoire des permutations et la nature algébrique du corps sous-jacent. D'une part, cela permet de représenter d'une façon simple et surtout concise des permutations complexes des éléments d'un corps fini. Cet aspect sera utilisé pour essayer de définir des systèmes de chiffrement, le plus souvent dans le domaine de la cryptographie à clé publique comme nous allons le détailler dans la section 5.1. D'autre part, cela permet d'établir des propriétés combinatoires à l'aide des techniques de calculs algébriques habituelles. Ainsi, lors de l'étude des fonctions booléennes ou vectorielles, si l'on cherche à obtenir des caractéristiques extrémales, il est courant de voir intervenir des familles de polynômes de permutation. Nous verrons des exemples dans la section 5.2.1.

5.1 Applications en cryptographie à clé publique

La première application, et celle qui a relancé l'intérêt pour le sujet, se situe en cryptographie à clé publique. Levine et Brawley [BL77] ont grandement contribué à relancer la recherche concernant les polynômes de permutation définis sur les corps finis en proposant des applications cryptographiques. D'autres constructions générales ont aussi été proposées par Schwenk et Huber [SH98].

5.1.1 Généralisations du RSA

Une autre motivation a été l'apparition dès 1978 de l'algorithme RSA pour la signature et le chiffrement à clé publique [RSA78]. En effet, il s'agit d'un exemple particulièrement simple d'utilisation de polynômes de permutation.

Rappelons brièvement son principe. Soient p et q deux nombres premiers. Notons $N = pq$ leur produit et choisissons e un entier premier avec $\varphi(N) = (p-1)(q-1)$, et $d = e^{-1} \pmod{\varphi)(N)}$. Le couple (N, e) va constituer la clé publique, et d est la clé privée.

Le chiffrement est alors simplement défini par

$$c = m^e \pmod{N};$$

et le déchiffrement par

$$m = c^d \pmod{N}.$$

Le polynôme de permutation intervenant est donc le polynôme $X^e \in (Z/N\mathbb{Z})[X]$. Il possède les propriétés suivantes :

- spécification efficace. Étant un monôme unitaire il suffit de définir l'exposant, c'est ce qui permet à la clé publique d'être de taille faible ;
- évaluation efficace puisqu'il s'agit simplement d'une exponentiation modulaire ;
- inversion difficile *si l'on ne connaît pas la trappe*, c'est-à-dire ici la décomposition $N = pq$.

De nombreux travaux ont été consacrés à la recherche d'autres familles de polynômes pouvant être utilisés suivant ce schéma.

On peut en particulier citer le système basé sur les polynômes de Dickson et baptisé LUC par Smith et Lennon [SL93] (la définition originale utilisait des suites de Lucas). On remarquera que ce cryptosystème est très proche de ceux proposés par Müller et Nobauer quelques années auparavant [MN81, MN85, Nöb88]. Les polynômes de Dickson

possèdent bien les trois propriétés ci-dessus. Pour les définir, il suffit de donner leur degré et un élément de l'anneau (voir la définition 3.39). L'évaluation peut se faire de manière efficace grâce à la définition récursive [JQ96]. Enfin comme pour le RSA, la "trappe", c'est-à-dire l'information supplémentaire qui permet une inversion efficace, se trouve dans la construction de l'anneau sous-jacent. À l'heure actuelle ce système dispose encore sensiblement des mêmes garanties de sécurité que le RSA.

De tels polynômes de permutation à trappes peuvent être à la base de nombreuses constructions cryptographiques. On pourra consulter en particulier l'article de Castagnos et Vergnaud [CV07].

Une autre voie de recherche consiste à considérer des polynômes sur des corps finis et à insérer la trappe dans la définition de ces polynômes.

5.1.2 Cryptographie multivariée

Un autre domaine d'application possible en cryptographie à clé publique est celui des schémas multivariés. Ici, la trappe va se trouver dans la construction du polynôme.

Commençons par exposer un des premiers d'entre eux, proposé par Matsumoto et Imai [MI83, MIHM85, MI88] et couramment appelé $C^\star$.

Dans ce système, on dispose d'un polynôme central X^{q^n+1} qui va être masqué par deux applications linéaires bijectives S et T. Le chiffrement est alors simplement l'évaluation de $S \circ X^{q^n+1} \circ T$, et ce polynôme constitue donc la clé publique.

Le choix de ce polynôme central répond à deux attentes : pouvoir assurer facilement la bijectivité (nous avons vu que les monômes sont la famille la plus simple de ce point de vue), et assurer une représentation relativement compacte de la clé publique. En effet, en prenant un polynôme de degré algébrique 2, on est assuré que la clé publique obtenue est de la forme

$$\sum_{0 \geq i,j < n} a_{i,j} X^{q^i+q^j} + \sum_{0 \geq k < n} b_k X^{q^k}.$$

Il suffit donc de $n(n+1)/2$ éléments de $\mathbb{F}_q$ pour définir ce polynôme, à comparer aux q^{n-2} coefficients pour un polynôme de permutation en général.

Malheureusement, ce système n'est pas sûr. Une attaque très efficace a été proposée par Jacques Patarin [Pat95], et une variante montre qu'autoriser S et T à être des applications affines n'apporte pas plus de sécurité.

Cependant de nombreuses variantes ont été imaginées pour tenter d'obtenir un système sûr. Pour cela, on peut enlever des équations, ajouter de l'aléa, projeter sur un sous-espace, etc. Toute la taxinomie des différentes approches possibles a été recensée par Christopher Wolf dans sa thèse intitulée *Multivariate Quadratic Polynomials in Public Key Cryptography* [Wol05].

Un des derniers systèmes pouvant posséder des paramètres de sécurité attractifs est celui proposé par Jacques Patarin, HFE [Pat96a, Pat96b]. Ici, le polynôme central n'est cependant pas une permutation et il faut adapter le chiffrement afin d'obtenir une bijection. Dans l'article introduisant HFE, le problème de trouver une famille de polynômes de permutation quadratiques est soulevé. Cela permettrait en effet de définir une nouvelle variante de l'algorithme originel de Matsumoto-Imai.

L'attaque de $C^\star$ proposée par Jacques Patarin procède par linéarisation et utilise le

fait que si l'on pose $X^{q^n+1} = Y$, alors on a aussi

$$X^{q^{2n}}Y = Y^{q^n}X;$$

et les exposants de cette équation sont tous de poids 1. On peut alors se demander ce qui se passerait si l'on utilisait un autre polynôme à la place du monôme original.

Nous avons au cours de cette thèse défini une famille de tels polynômes de permutations quadratiques de la forme

$$\sum_{i=0}^{n} a_i X^{2^i+1}$$

définis de manière récursive (voir 3.3). Nous avons de plus vu qu'ils ne sont pas tous équivalents au monôme X^{q+1}. Enfin nous avons vu que l'on dispose d'un algorithme simple pour effectuer la permutation inverse (voir l'algorithme 3).

On peut donc imaginer utiliser ces familles pour obtenir une nouvelle variante du système de Matsumoto-Imai. L'intérêt cryptographique de cette construction reste encore à étudier.

5.2 Applications en cryptographie à clé secrète

En cryptographie à clé secrète, l'utilisation faite des polynômes de permutation est souvent différente. En effet, pour des raisons d'efficacité, on ne peut pas se permettre de travailler sur de trop gros corps ou avec des expressions polynomiales compliquées. Le caractère bijectif du chiffrement est alors garanti par des constructions simples du type schéma de Feistel, tandis que les différents composants ne sont pas nécessairement injectifs. Cependant, la résistance aux différentes familles d'attaques classiques, telles que la cryptanalyse linéaire ou la cryptanalyse différentielle, repose souvent sur les propriétés des fonctions booléennes ou vectorielles utilisées. Pour concevoir de nouveaux systèmes, on a donc besoin de construire des classes de fonctions extrémales pour divers critères. C'est ici que les polynômes de permutation apparaissent, à travers leur nature combinatoire.

Nous allons détailler le lien entre deux critères importants et les polynômes de permutation, et montrer comment les résultats vus au chapitre 3 peuvent s'appliquer.

5.2.1 Fonctions APN

En chiffrement itératif par bloc, la cryptanalyse différentielle — puis par la suite ses diverses variantes — est devenue un outil majeur depuis ses premiers succès contre le DES [Mat93], et chaque nouveau système se doit de donner des arguments de résistance à cette classe d'attaques.

Le principe de base de cette attaque est d'identifier des couples de textes clairs dont la différence est fixée, et dont la différence des chiffrés correspondants est fortement biaisée.

Plus formellement, définissons la dérivée d'une fonction discrète.

Définition 5.1. Soient n un entier, p un nombre permier et F une fonction de $\mathbb{F}_{p^n}$ dans lui même. Pour tout $a \in \mathbb{F}_{p^n}$, la dérivée de F au point a est la fonction D_aF de $\mathbb{F}_{p^n}$ dans lui-même

$$D_aF(x) = F(x+a) + F(x), \forall x \in \mathbb{F}_{p^n}.$$

On s'intéresse alors au biais maximum que l'on peut observer.

Définition 5.2. Soit F une fonction de $\mathbb{F}_{p^n}$ dans lui-même. Pour tout a et b dans $\mathbb{F}_{p^n}$, on défini

$$\delta_F(a,b) = \#\{x \in \mathbb{F}_{p^n} | D_a F(x) = b\}$$

ainsi que

$$\delta(F) = \max_{a \neq 0, b \in \mathbb{F}_{p^n}} \delta_F(a,b).$$

En caractéristique supérieure à 2, il est possible d'avoir $\delta(F)$ égal à 1. Les fonctions atteignant cette valeur sont appelées fonctions *parfaitement non-linéaires* ou PN.

En caractéristique 2, on a

$$D_a F(x) = F(x+a) + F(x) = D_a F(x+a)$$

ce qui implique que $\delta(F)$ est supérieur ou égal à 2. Les fonctions pour lesquelles cette borne est atteinte sont appelées fonctions *presque parfaitement non linéaires* ou APN (pour *almost perfect nonlinear*). C'est à celles-ci que nous allons maintenant nous consacrer.

L'existence de permutations APN dans le cas d'un nombre pair de variables est un problème ouvert majeur. Il revêt une importance toute particulière puisqu'en pratique les chiffrements par bloc travaillent généralement sur un nombre de bits pairs pour des raisons d'implémentation.

Théorème 5.3. *Soit F une permutation de $\mathbb{F}_{2^n}$ avec $n = 2t$.*

(i) Si $n = 4$ alors F n'est pas APN.

(ii) Si $F \in \mathbb{F}_4[X]$ alors F n'est pas APN.

(iii) Si $F \in \mathbb{F}_{2^t}[X]$ alors F n'est pas APN.

(iv) Si F est telle que toutes ses composantes $f_\lambda, \lambda \in \mathbb{F}_{2^n}^\star$, sont des fonctions plateau, alors F n'est pas APN.

Remarque 5.4. (iv) implique qu'il n'existe pas de permutation quadratique APN avec un nombre pair de variables.

Nous allons voir un exemple de caractérisation de fonctions APN dans lequel intervient une famille de polynômes de permutation.

Théorème 5.5 ([BCCLC06])**.** *Les seules fonctions APN quadratiques de $\mathbb{F}_{2^n}$ de la forme*

$$F(X) = \sum_{i=1}^{n-1} c_i x^{2^i+1}, c_i \in \mathbb{F}_{2^n}, \tag{5.1}$$

sont les fonctions puissance X^{2^k+1} où k est premier avec n.

Proposition 5.6. *Soit F définie par (5.1). Alors F est APN si et seulement si le polynôme*

$$Q(X) = F(X)/X^2 = \sum_{i=1}^{n-1} c_i x^{2^i-1}, c_i \in \mathbb{F}_{2^n} \tag{5.2}$$

est un polynôme de permutation.

Preuve : Pour tout $a \in \mathbb{F}_{2^n}^\star$, on a

$$D_aF(x) = \sum_{i=1}^{n-1} c_i \left(x^{2^i} a + x a^{2^i} \right) + F(a).$$

l'ensemble $\{D_aF(x), x \in \mathbb{F}_{2^n}\}$ est donc de cardinal 2^{n-1} si et seulement si le polynôme affine D_aF a un noyau de dimension 1, c'est-à-dire si

$$\sum_{i=1}^{n-1} c_i \left(x^{2^i} a + x a^{2^i} \right) \neq 0, \forall x \notin \{0, a\},$$

ou encore de manière équivalente (en divisant par xa) si

$$\sum_{i=1}^{n-1} c_i x^{2^i-1} \neq \sum_{i=1}^{n-1} a^{2^i-1}, \forall x \notin \{0, a\}.$$

On doit donc avoir pour tout a et b distincts non nuls $Q(a) \neq Q(b)$. De plus s'il existe a non nul tel que $Q(a) = Q(0) = 0$, il est unique. Mais dans ce cas, Q satisfait les hypothèses du lemme 3.49 et doit donc être de degré $2^n - 1$ ce qui est impossible. Donc pour tout a non nul, $Q(a) \neq Q(0)$ ce qui ajouté à la condition précédente assure que Q est un polynôme de permutation. □

Afin de prouver le théorème, il ne nous reste plus qu'à étudier les polynômes de la forme (5.2). En fait, nous avons déjà démontré ce résultat, il s'agit du théorème 3.48.

Le lien étroit entre les fonctions APN et les permutations ne s'arrête pas là.

En effet, comme nous l'avons vu lors de l'introduction, étudier le caractère APN d'une fonction F revient à étudier le nombre de solutions de l'équation $F(x+a)+F(x) = b$ pour toutes les valeurs des paramètres a non nul et b. Mais on peut aussi dire que la fonction F est une permutation si et seulement si pour tout a non nul, l'équation $F(x+a)+F(x) = 0$ n'a pas de solution.

C'est ainsi que la famille de binômes de permutation décrite dans le chapitre précédent au théorème 3.28 a été obtenue comme sous produit d'une recherche de binômes correspondant à une fonction APN, en utilisant la même technique de preuve que celle que nous avons détaillée, à savoir la méthode multivariée décrite par Hans Dobbertin [Dob02].

5.2.2 Fonctions courbes

Une autre famille de fonctions booléennes importante en cryptographie est celle des fonctions dites courbes. Afin de définir précisément cette notion, nous avons besoin de rappeler quelques notions liées à l'étude spectrale des fonctions booléennes.

Définition 5.7. La *transformée de Walsh* d'une fonction booléenne $F : \mathbb{F}_{2^n} \to \mathbb{F}_2$ est définie par

$$\begin{array}{rccc} \mathcal{F}[F] : & \mathbb{F}_{2^n} & \to & \mathbb{Z} \\ & a & \mapsto & \sum_{x \in \mathbb{F}_{2^n}} (-1)^{F(x)+a\cdot x} \end{array} \tag{5.3}$$

La valeur de $\mathcal{F}[F]$ au point a est appelé *coefficient de Walsh* en a et le multi-ensemble

$$\{\mathcal{F}[F](a) : a \in \mathbb{F}_{2^n}\}$$

est le *spectre de Walsh* de la fonction F.

La transformée de Walsh est donc la transformée de Fourier de la fonction signe associée à notre fonction booléenne. Comme pour la transformée de Fourier classique, on peut définir cette transformation de manière récursive ce qui nous permettra d'obtenir un algorithme efficace de calcul en $\mathcal{O}(n2^n)$.

Chaque coefficient de Fourier correspond à la corrélation entre notre fonction et une fonction affine. On s'intéresse souvent à la valeur maximale de cette corrélation

$$\mathcal{L}(F) = \max_{a \in \mathbb{F}_{2^n}} |\mathcal{F}[F](a)| .$$

Cette valeur est naturellement liée à ce que l'on appelle la non-linéarité de la fonction F.

Définition 5.8. La *non-linéarité* d'une fonction booléenne F, notée $\mathcal{N}(F)$, est la distance de F à l'ensemble des fonctions affines :

$$\mathcal{N}(F) = \min_{a \in \mathbb{F}_{2^n}} wt(F + \varphi_a).$$

On vérifie de plus l'égalité suivante :

$$\mathcal{N}(F) = \frac{1}{2}\left(2^n - \mathcal{L}(F)\right).$$

On voit en particulier que $\mathcal{N}(F)$ est compris entre 0 et 2^{n-1}. D'un point de vue cryptographique, il est le plus souvent intéressant d'être le plus loin possible des applications linéaires, ce qui nous amène à la propriété suivante.

Proposition 5.9. *Les fonctions booléennes telles que $\mathcal{L}(F) = 2^{n/2}$ n'existent que pour n pair. On appelera ces fonctions des* fonctions courbes.

Il existe de nombreux liens entre les fonctions APN et les fonctions courbes, on pourra par exemple consulter [BCCLC06].

Dans [CK08], Pascale Charpin et Gohar Kyureghyan établissent le lien entre le caractère courbe de fonctions quadratiques et une famille de binômes de permutation. Elles démontrent le résultat suivant.

Théorème 5.10. *Soient $\omega \in \mathbb{F}_4 \setminus \mathbb{F}_2$ et $\mu \in \mathbb{F}_{2^{4r}}$. La fonction booléenne*

$$\mathrm{Tr}_1^{4r}(\mu x^{2^{2r}+2^{r+1}+1})$$

est courbe si et seulement si r est impair et il existe $a \in \mathbb{F}_{2^{4r}}$ et $\lambda \in \omega\mathbb{F}_{2^r}$ tels que $\mu = \lambda a^{2^{2r}+2^{r+1}+1}$.

La preuve, que nous ne détaillerons pas ici, repose sur le résultat suivant.

Théorème 5.11. *Soit $W \subset \mathbb{F}_{2^{2t}}$ tel que $\mathbb{F}_{2^{2t}} = \mathbb{F}_{2^t} \oplus W$. Soit une fonction booléenne $f : \mathbb{F}_{2^{2t}} \to \mathbb{F}_2$ pouvant s'exprimer sous la forme*

$$f(x) = f(y + a) = \mathrm{Tr}_1^t(y\pi(a) + h(a)),$$

où $x = y + a$ avec $y \in \mathbb{F}_{2^t}$, $a \in W$ et $\pi, h : W \to \mathbb{F}_{2^t}$. Alors f est une fonction courbe si et seulement si π est bijective.

Il reste alors à étudier la famille de binômes du type $X^{2^k+2} + \nu X$ et à caractériser ceux qui sont des permutations de $\mathbb{F}_{2^t}$, ce que nous avons déjà présenté au théorème 3.38.

5.2.3 Registre à décalage à rétroaction non linéaire

La dernière application que nous présenterons ici concerne les registres à décalage à rétroaction non linéaire (en anglais *Nonlinear Feedback Shift Registers* ou NLFSRs).

Afin de réaliser des systèmes nécessitant peu de ressources matérielles, de nombreuses primitives cryptographiques ont utilisé comme brique de base des registres à décalage à rétroaction linéaire. Le lecteur n'étant pas familier avec ces objets peut se reporter à la seconde partie où nous présentons en détail ce dispositif. Cependant, après les nombreuses attaques qui ont été menées sur ce type de dispositifs, il apparaît que leur linéarité conduit à une trop grande faiblesse. Beaucoup de constructions récentes, que ce soit des propositions de chiffrements à flot lors de l'appel à proposition européen eSTREAM ou des propositions de fonctions de hachage lors de l'appel à proposition du NIST, se basent maintenant sur des registres à décalage à rétroaction non linéaire.

La recherche de fonction de rétroaction apportant des garanties de sécurité pour de tels dispositifs est un problème important. Une des premières propriétés que l'on peut souhaiter est d'obtenir des cycles de taille maximale mais en général, il est difficile de caractériser les fonctions permettant d'assurer cela. Pour les fonctions de degré deux, Gary Mullen a pu établir une telle classification [Mul89]. Il a aussi montré que l'étude des polynômes de permutation, et en particulier de polynômes de permutation locaux [Mul80b, Mul80a, DHK04] peut être intéressante dans le cadre des NLFSR.

Deuxième partie

Cryptanalyse de registres combinés

Dans cette deuxième partie, nous allons nous intéresser à un modèle de chiffrement très étudié, le modèle des générateurs par combinaison.

Ce modèle utilise comme composant de base des LFSRs qui sont comme nous le verrons dans le chapitre 6 des dispositifs permettant d'implémenter les récurrences linéaires. Cela permet de garantir la période ainsi que certaines propriétés statistiques des séquences produites. Il est ensuite nécessaire d'introduire une partie non linéaire pour espérer obtenir un système plus sûr. Pour cela, on combine les différents générateurs à l'aide d'une fonction booléenne.

Le découpage en plusieurs registres indépendants permet la mise en place des attaques dites par *corrélation*, introduites en 1985 par Siegenthaler [Sie85]. De nombreuses variantes ont ensuite proposées la plupart se basant sur la modélisation en termes de codes correcteurs d'erreurs proposée par Meier et Staffelbach en 1988 [MS88]. Nous décrirons ces modèles dans la section 7.3.

Le travail présenté dans cette partie est un travail en commun avec Frédéric Didier et a été financé partiellement par le CELAR/DGA.

Nous avons proposé une variante de ces attaques [CDLCT07] que nous exposerons dans le chapitre 8. Elle repose sur une analyse plus précise, sans passer par la modélisation en terme de codes correcteurs, dans la lignée des travaux de Sabine Leveiller [Lev04]. Une des propriétés remarquables de cette attaque est de ne pas dépendre des propriétés de résilience de la fonction de filtrage.

Toutes ces attaques requièrent dans une phase de précalcul de trouver des multiples de poids faible de certains polynômes. L'étude générale de ce problème sera l'objet du chapitre 9. En particulier, nous verrons une amélioration que nous avons proposée et qui est particulièrement adaptée au cas des attaques par corrélation sur les générateurs par combinaison [DLC07].

Chapitre 6

Récurrence linéaire sur les corps finis

Avant d'aller plus loin, nous allons rappeler les propriétés essentielles des récurrences linéaires sur les corps finis. Nous nous contenterons d'une présentation succincte en omettant les démonstrations. Pour plus de détails, le lecteur est invité à consulter le livre de Lidl et Niederreiter [LN97] ; on pourra aussi consulter la thèse de Cédric Lauradoux [Lau07] en ce qui concerne l'implémentation matérielle des dispositifs permettant de générer les suites correspondantes.

6.1 LFSR sous forme de Galois

Les registres à décalage à rétroaction linéaire, LFSR sont un des éléments les plus courants dans la conception de chiffrements à flot. Ils sont adaptés à une implémentation matérielle efficace, et comme nous allons le voir, leur structure est intimement liée aux corps finis, ce qui permet de garantir de nombreuses propriétés statistiques par exemple.

Contrairement à la présentation qui en est faite habituellement, nous nous intéresserons ici plus à l'état interne de notre LFSR qu'à la séquence de sortie produite.

Nous commençons par présenter les LFSR dits sous forme de Galois. Ceux-ci correspondent au dispositif représenté sur la figure 6.1.

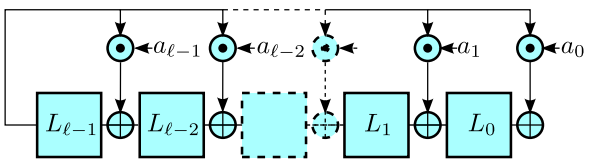

FIG. 6.1 – forme de Galois d'un LFSR

On dispose d'un registre L de taille ℓ, constitué de cellules $(L_0, \ldots, L_{\ell-1})$ que nous appellerons *état interne* du LFSR et qui sont à valeurs dans un corps fini $\mathbb{F}_q$. Cet état interne est mis à jour à chaque pas d'horloge en appliquant la formule de récurrence suivante :

$$L(t+1) = (a_0 L_{\ell-1}(t), L_0(t) + a_1 L_{\ell-1}(t), \ldots, L_{\ell-2}(t) + a_{\ell-1} L_{\ell-1}(t)) .$$

Si on considère que cet état interne représente un polynôme $\sum_{i=0}^{\ell-1} L_i X^i$ de $\mathbb{F}_q[X]$, on remarque alors que la relation de récurrence peut s'écrire

$$L(t+1) = XL(t) \pmod{g(X)}, \quad \text{avec } g(X) = X^\ell - \sum_{i=0}^{\ell-1} a_i X^i. \tag{6.1}$$

Le polynôme g est appelé polynôme *générateur* du LFSR.

En itérant cette définition récursive, on a donc $L(t) = X^t L(0)$ où $L(0) \in \mathbb{F}_q[X]/ < g(X) >$ est appelé *l'état initial*. Or la structure de l'anneau quotient $\mathbb{F}_q[X]/ < g(X) >$ est très bien connue. En particulier, si le polynôme g est irréductible, il s'agit d'un corps fini (corps de Galois d'où l'appellation forme de Galois), et si de plus g est un polynôme primitif, alors l'ordre de X est $q^\ell - 1$. C'est aussi la période de la suite des états internes si l'état initial est non nul. Celui-ci prendra alors toutes les valeurs non nulles possibles. Le plus souvent, on considère seulement la séquence de sortie de notre LFSR, c'est-à-dire la séquence des valeurs $(L_{\ell-1}(t))_{t\geq 0}$, qui est appelée lorsque la période est maximale une m-séquence.

6.2 Cas général

Il existe beaucoup d'autres manières de définir des récurrences linéaires sur un corps fini. Considérons un registre L de longueur ℓ dont les cellules contiennent des éléments d'un corps fini $\mathbb{F}_q$, et dont la fonction de mise à jour est linéaire. En notant G la matrice correspondant à la fonction de mise à jour, l'état interne du registre au temps t, est alors défini par

$$L(t) = L(0) \cdot G^t.$$

Le cas vu précédemment d'un LFSR sous forme de Galois correspond à une matrice $G = C(g)$ de la forme

$$C(g) = \begin{pmatrix} 0 & 0 & \dots & 0 & a_0 \\ 1 & 0 & \dots & 0 & a_1 \\ 0 & 1 & & 0 & a_2 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & a_{\ell-1} \end{pmatrix}^T$$

appelée *matrice compagnon* du polynôme $g(X) = X^\ell - \sum_{i=0}^{\ell-1} a_i X^i$.

Un théorème important d'algèbre linéaire permet de définir une forme canonique des matrices carrées, à travers leurs invariants de similitude. On peut l'énoncer par exemple de la manière suivante.

Théorème 6.1 (Forme de Frobenius)**.** *Soit M une matrice à coefficients dans un corps K. Alors il existe une unique matrice diagonale par bloc semblable à M*

$$Diag(C(P_1), \dots, C(P_k)), \quad P_i \in \mathbb{F}_q[X]$$

avec P_i divisant P_{i+1} pour tout i.

Cette matrice est appelée forme de Frobenius*, ou forme canonique rationnelle, de la matrice M.*

Le polynôme minimal de M est alors P_k.

On rappelle que deux matrices sont dites semblables si elles représentent la même application linéaire dans deux bases différentes. On voit alors qu'à un changement de base près, le dispositif est équivalent à k LFSRs sous forme de Galois indépendants, dont les polynômes de rétroaction vérifient de plus $P_i|P_{i+1}$.

Cela nous donne en particulier que toute récurrence linéaire définie par une matrice dont le polynôme caractéristique est le polynôme minimal, est équivalente à une récurrence définie par un LFSR sous forme de Galois.

Un cas particulier important en pratique correspond à un matrice de récurrence qui est la transposée d'une matrice compagnon. Cela correspond au dispositif de la figure 6.2, appelé LFSR sous forme de Fibonnaci.

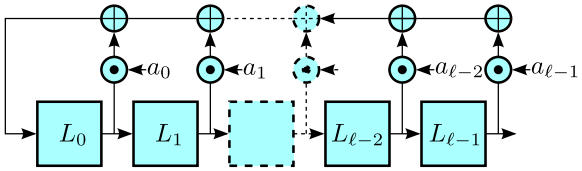

FIG. 6.2 – forme de Fibonnaci d'un LFSR

La récurrence vérifiée par l'état interne de ce LFSR est définie par

$$L(t+1) = \left(\sum_{i=0}^{\ell-1} a_i L_i(t), L_0, \ldots, L_{\ell-2} \right). \tag{6.2}$$

C'est en fait sous cette forme qu'on d'abord été définis les LFSR. On voit en effet que seule une des cellules nécessite réellement le calcul d'une rétroaction linéaire, les autres étant obtenues simplement par décalage.

Le théorème nous permet alors d'énoncer l'équivalence des formes de Galois et de Fibonnaci, au sens où il suffit d'effectuer un changement de base pour passer de l'un à l'autre.

En particulier, la mise à jour d'un LFSR sous forme de Fibonnaci correspond à la multiplication par X dans l'anneau $F_q[X]/ < g(X) >$, exprimée dans la base duale.

Un autre cas particulier intéressant est le cas de registres indépendants de polynômes de rétroaction $P_1, \ldots, P_n$. La matrice correspondante est alors une matrice diagonale par bloc $Diag\,(C(P_1), \ldots, C(P_n))$. Cependant il ne s'agit pas de la forme de Frobenius puisqu'on a pas la propriété de divisibilité. Son polynôme minimal est le plus petit multiple commun des polynômes minimaux de chaque bloc. Ainsi, si l'on considère des registres ayant pour polynômes générateurs des polynômes premiers entre eux deux à deux, on voit alors que la forme de Frobenius de notre matrice est simplement la matrice compagnon $C(P_1 \cdots P_n)$. On obtient donc l'équivalence à un changement de base près entre un registre de polynôme générateur $g(X)$ et des registres indépendants de polynômes générateurs les facteurs irréductibles de $g(X)$.

6.3 Propriétés utiles

Faire avancer un LFSR

Enfin, il est important de noter que l'expression

$$L(t) = L(0) \cdot G^t$$

permet de calculer efficacement l'état interne d'un LFSR à un instant donné sans avoir besoin de calculer toute la séquence, en effectuant une exponentiation rapide de la matrice G. Ce calcul est indépendant de l'état initial, et si l'on veut ensuite connaître la valeur correspondant à un état initial particulier, il suffit d'effectuer un produit matrice vecteur.

Lorsque le polynôme générateur est irréductible, il est aussi possible de revenir en arrière facilement puisqu'il s'agit simplement d'une division dans le corps $\mathbb{F}_q[X]/ < g(X) >$.

Auto-corrélation

Définition 6.2. Soit $(s_t)_{t\geq 0}$ une suite périodique de période T. On définit les coefficients d'auto-corrélation périodique de s par

$$c(\tau) = \sum_{t=0}^{T-1} (-1)^{s_t + s_{t+\tau}}.$$

Les m-séquences ont alors la propriété importante d'avoir un spectre d'auto-corrélation "plat" au sens suivant.

Proposition 6.3. *Soit $(s_t)_{t\geq 0}$ une m-séquence. Alors les coefficients d'auto-corrélation de s vérifient*

$$c(\tau) = \begin{cases} 2^m - 1 & si\ \tau = 0 \\ -1 & sinon\ . \end{cases}$$

Ces suites ayant un nombre impair de termes, c'est la plus petite valeur possible quand $\tau \neq 0$.

6.4 Équations de parité

Sous cette formulation matricielle, en considérant le polynôme minimal de la matrice G, $g(X) = X^\ell + \sum_{i=0}^{\ell-1} a_i X^i$ on remarque que la suite des états internes vérifie aussi la récurrence linéaire

$$L(t+\ell) + \sum_{i=0}^{\ell-1} a_i L(t+i) = (L(0) \cdot G^t) \cdot g(G) = 0.$$

C'est ce que nous appellerons par la suite une *équation de parité*, associée au polynôme g, vérifiée par l'état interne de notre LFSR.

On remarque en particulier que cette équation de parité est vérifiée non seulement par la séquence de sortie produite par notre LFSR, mais aussi par la séquence des états internes complets.

On peut par ailleurs démontrer le résultat suivant.

Proposition 6.4. *Soit L un* LFSR *de polynôme minimal $g(X)$. Alors un polynôme $p(X)$ définit une équation de parité vérifiée à tout instant t et quel que soit l'état initial, si et seulement si $p(X)$ est un multiple de $g(X)$.*

Comme nous le verrons dans le chapitre suivant, l'existence de telles équations de parité, ne dépendant que du polynôme de rétroaction de notre LFSR et pas des conditions initiales ni du temps, est à la base des attaques dites par corrélation. En particulier, pour monter ces attaques, on a le plus souvent besoin de trouver des équations faisant intervenir peu de termes. Cela revient donc à calculer des multiples de petit poids de notre polynôme minimal. Nous reviendrons sur les techniques utilisées pour résoudre ce problème dans le chapitre 9.

Pour fixer les idées prenons un exemple. Sur la figure 6.3, on a représenté les états internes d'un LFSR sous forme de Galois à trois instants t, $t+1$ et $t+19$. Le polynôme générateur est $g(X) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^7$. En remarquant que

$$(1 + x^2 + x^6 + x^7 + x^9 + x^{10} + x^{12}) \cdot g(x) = 1 + x + x^{19},$$

on est assuré que pour tout instant t et tout état initial I on a l'équation de parité

$$L(t) + L(t+1) + L(t+19) = 0,$$

ce qui est bien vérifié ici.

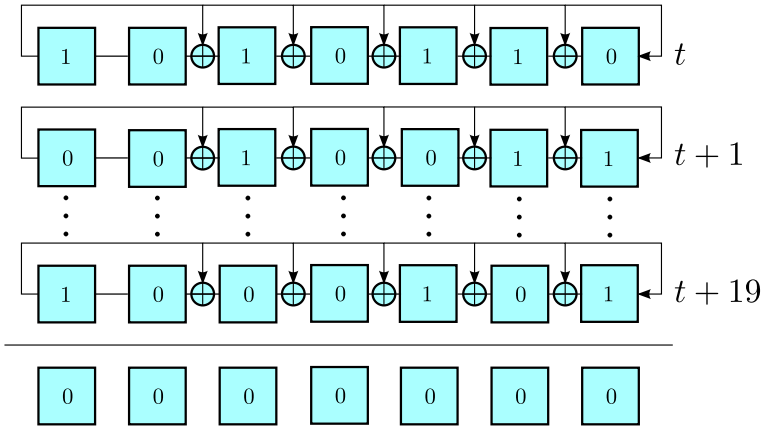

FIG. 6.3 – Équation de parité vérifiée par l'état interne d'un LFSR en mode Galois

Chapitre 7

Les registres combinés

Dans ce chapitre, nous allons présenter un modèle basique de chiffrement à flot très étudié, appelé registres combinés. Il est souvent présenté par opposition au registre filtré, constitué d'un seul LFSR dont l'état interne est pris en entrée d'une fonction booléenne. Ici l'idée est d'utiliser plusieurs registres, de polynômes générateurs premiers entre eux afin d'obtenir une période la plus grande possible. Comme nous l'avons vu précédemment, à un changement de base près on peut se ramener au cas d'un seul LFSR, de polynôme de rétroaction non irréductible. Le but recherché en prenant des registres indépendants est d'obtenir un dispositif s'implémentant mieux en matériel puisqu'en particulier les registres considérés sont plus courts.

7.1 Modèle de registres combinés

Le modèle choisi ici est représenté sur la figure 7.1. Il s'agit en fait d'un système un peu différent de celui pris habituellement puisqu'on autorise notre fonction de filtrage à prendre plusieurs entrées dans chaque registre.

On considère r registres $R_1, \ldots, R_r$ indépendants, de longueurs respectives $L_1, \ldots, L_r$. Ces registres sont combinés par une fonction booléenne F à n variables. Ses entrées sont réparties entre les différents registres. Les n_i bits pris dans le registre R_i seront notés $x_1^i, \ldots, x_{n_i}^i$. Nous ne préciserons pas leur position dans le registre car cela n'intervient pas dans notre attaque. Ces valeurs sont en fait des fonctions du temps t. À chaque pas d'horloge, les états internes des registres sont mis à jour, et la fonction de filtrage F est utilisée pour générer un bit de suite chiffrante. Enfin, le message chiffré c est obtenu par ou exclusif bit à bit du message clair m et de la suite chiffrante.

Avec les notations précédentes, cela peut se résumer ainsi :

$$\begin{aligned} R_i(t+1) &= g_i\left(R_i(t)\right), \quad \forall 1 \leq i \leq r \\ c(t) &= m(t) \oplus F\left(x_1^1(t), \ldots, x_{n_r}^r(t)\right). \end{aligned}$$

Les fonctions g_i de mise à jour des registres R_i peuvent être de nature très diverses. Cependant, pour des raisons d'efficacité, il est très courant de se restreindre à des registres dits à décalage. On a alors

$$g(x_1, x_2, x_3, \ldots, x_{n-1}, x_n) = (f(x_1, x_2, \ldots, x_n), x_1, x_2, \ldots, x_{n-1})$$

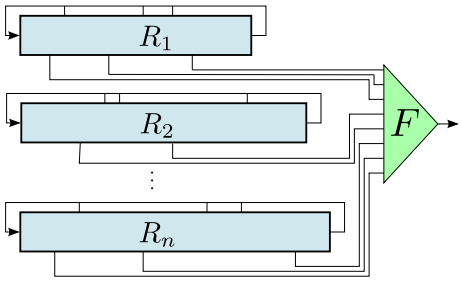

FIG. 7.1 – Modèle de registres combinés

Comme nous l'avons vu à la fin de la première partie de cette thèse 5.2.3, les propriétés de tels registres lorsque f est non-linéaire sont mal connues. Au contraire lorsque f est une fonction linéaire, nous avons vu au chapitre précédent que l'on peut voir la fonction g comme une multiplication dans un corps fini, et ainsi déduire de nombreuses propriétés statistiques. C'est dans ce cadre là que nous allons poursuivre notre étude.

L'utilisation de registres combinés permet en particulier d'obtenir une garantie sur la complexité linéaire de la suite obtenue. On rappelle que la complexité linéaire d'une séquence est la taille du registre du plus petit LFSR générant cette séquence.

Théorème 7.1. *Soient n LFSRs de polynômes de rétroactions primitifs et premiers entre eux et de longueurs respectives L_i. Si on combine ces registres par une fonction booléenne F à n entrées, exprimée sous forme normale algébrique, alors la complexité linéaire de la suite générée est égale à*

$$F(L_1, \ldots, L_n)$$

calculée sur les entiers.

7.2 Principe des attaques par corrélation

Les attaques par corrélation sont des attaques à clair connu. On dispose d'une suite chiffrante $(z_t)_{t\geq 0}$, et on souhaite retrouver l'état initial de nos registres. On suppose de plus que l'on connaît tous les éléments constituants du système, à savoir la fonction F de combinaison et les différents paramètres des LFSRs.

Pour simplifier, nous allons présenter ces attaques lorsque la fonction F ne prend qu'une entrée par registre.

La première idée, due à Siegenthaler [Sie85], est d'utiliser l'indépendance de nos registres pour retrouver l'état initial de chacun indépendamment des autres. Pour cela, on utilise une éventuelle corrélation entre les valeurs d'entrée prises dans un registre et la sortie de la fonction. Soit

$$p_i = Pr\left(F(x_1, \ldots, x_n) = x_i\right).$$

Calculons pour une initialisation I du registre la valeur

$$\widetilde{P}_i(I) = \frac{1}{N} \sum_{t=0}^{N-1} z_t \oplus x_i^t(I)$$

(on note $x_i^t(I)$ la valeur après t itération du bit x_i en utilisant l'initialisation I).

Si on choisit la bonne initialisation, $\widetilde{P}_i$ devrait converger vers une loi normale de moyenne p_i et de variance 1. En effet les bonnes propriétés statistiques des LFSRs assurent que les valeurs des entrées prises dans les autres registres vont être bien distribuées.

Si par contre on choisit une mauvaise initialisation, d'après la propriété 6.3, les suites générées par deux initialisations distinctes sont décorrélées et $\widetilde{P}_i$ converge donc vers une loi normale de moyenne 1/2 et de variance 1.

Si la probabilité p_i est différente de 1/2, on peut donc appliquer l'algorithme 4.

Algorithme 4 Attaque par corrélation

Entrées: Une suite chiffrante $(z_t)_{t\geq 0}$ et les paramètres du système
Sorties: L'état initial du $i$ème registre
  pour les $2^{L_i}-1$ initialisations possibles I du i-ème registre **faire**
    calculer $\widetilde{P}_i(I) = \frac{1}{N}\sum_{i=0}^{N-1} z_t \oplus x_i^t(I)$
  fin pour
  renvoyer L'initialisation I donnant la valeur $P_i(I)$ la plus éloignée de 1/2.

La valeur de N devra être de l'ordre de $(1/2-p_i)^{-2}$ pour pouvoir différencier la bonne initialisation.

Il est possible de contrer cette attaque en choisissant des fonctions sans corrélations, c'est-à-dire avec $p_i = 1/2$.

On peut étendre cette définition à plusieurs entrées.

Définition 7.2. Soit F une fonction booléenne équilibrée à n variables. F est dite *r-résiliente* si toutes les fonctions booléennes obtenues en fixant au plus r variables en entrée de F sont équilibrées.

On appelle *degré de résilience* de F la valeur r maximale pour laquelle F est r-résiliente.

Si la fonction de combinaison employée est r-résiliente, il faudra alors considérer la corrélation entre la suite chiffrante et une combinaison de $r+1$ de ses entrées pour obtenir un biais. On doit alors faire une recherche exhaustive sur l'initialisation de $r+1$ registres ce qui devient rapidement trop coûteux.

Il faut cependant faire des compromis entre le degré de résilience souhaité et le degré total.

Théorème 7.3 (borne de Siegenthaler)**.** *Soit F une fonction booléenne à n variable, de degré d et de degré de résilience r. Alors*

$$d + r \leq n - 1.$$

La meilleure combinaison possible de $r+1$ registre est en fait une combinaison linéaire comme l'ont montré Anne Canteaut et Trabia [CT00].

Proposition 7.4. *Soit F une fonction booléenne à n entrées, de degré de résilience r. Alors il existe une fonction g linéaire telle que*

$$Pr\left(F(x_1,\ldots,x_n) = g(x_{i_0},\ldots,x_{i_r})\right) > \frac{1}{2}.$$

De plus, la fonction g donnant la probabilité maximum est

$$g(x_{i_0}, \ldots, x_{i_r}) = x_{i_0} + \cdots + x_{i_r}.$$

7.3 Attaques par corrélation rapides

Les attaques par corrélation rapides, introduites par Meier et Staffelbach [MS88], sont basées sur une reformulation de notre problème en termes de correction d'erreurs.

On a vu que la suite chiffrante peut être vue comme une version bruitée de la suite $(x_{i_1} + \cdots + x_{i_r})$ avec une probabilité d'erreur

$$p = 1 - Pr\left(F(x_1, \ldots, x_n) = g(x_{i_0}, \ldots, x_{i_r})\right)$$

strictement inférieure à $1/2$. Or nous avons vu lors de la présentation générale des LFSRs que les valeurs $x_i^r(I)$ dépendent linéairement de l'initialisation I. On peut donc considérer que le mot

$$\left(\sum_{j=0}^{r} x_{i_j}^0, \ldots, \sum_{j=0}^{r} x_{i_j}^{N-1}\right)$$

appartient à un code linéaire, de longueur N et de dimension $L = \sum_{i=0}^{r} L_i$ la somme des tailles des registres considérés.

Il nous faut alors un algorithme de décodage rapide et performant. En effet, le taux d'erreur est ici très élevé, beaucoup plus que dans les problèmes de transmission classiques.

Sous cette forme, la longueur minimale de suite nécessaire au décodage est bornée par la théorie de l'information.

Proposition 7.5. *Avec les notations précédentes, la longueur de suite chiffrante nécessaire N pour le décodage vérifie*

$$N > L/C(p), \qquad \textit{avec } C(p) = 1 + p\log_2(p) + (1-p)\log_2(1-p)$$

($C(p)$ est la capacité du canal binaire symétrique de probabilité d'erreur p).

De plus, pour atteindre cette borne, il faut un algorithme de décodage à maximum de vraisemblance. Or ce type de décodage a une complexité importante. De nombreuses variantes ont été proposées [JJ02, JJ00, CJS00, MFI01]. Les deux techniques principales sont les suivantes.

Décodage itératif

On recherche des équations de parité très creuses – en général composées de 3 ou 4 termes – vérifiées par notre code linéaire. Comme nous l'avons vu, ces équations de parité correspondent à des multiples de poids très faible du plus petit commun multiple des polynômes générateurs. On utilise ensuite des techniques de décodage itératif de LDPC telles que l'algorithme de Gallager [Gal62].

Réduction de la dimension

On réduit la dimension du code considéré en cherchant des équations de parité partielles, s'annulant sur les $L-k$ dernières lignes. Cela se fait en cherchant des multiples du plus petit multiple commun des polynômes générateurs des $L-k$ derniers registres. On se retrouve alors avec un code de dimension k, avec une probabilité d'erreur plus grande. Pour minimiser cette probabilité d'erreur, on cherchera encore une fois des équations de parité les plus creuses possibles. Une fois notre nouveau code obtenu, on peut le décoder à maximum de vraisemblance en utilisant une transformée de Walsh en $\mathcal{O}(k2^k)$.

Chapitre 8

Nouvelle cryptanalyse de LFSRs combinés

Nous allons maintenant présenter notre attaque. Nous allons voir que le principe de départ reste similaire, mais nous allons nous intéresser à une probabilité différente. En particulier, cela nous permettra de ne pas avoir à regarder les propriétés de résilience de la fonction de combinaison. Cette attaque se rapproche des attaques par corrélations "vectorielles", cependant notre analyse n'utilise pas la notion de canal binaire symétrique mais reprend les travaux de Sabine Leveiller [Lev04] où le lien entre les équations de parité et la fonction booléenne est complètement exploité. C'est la suite directe des travaux de Frédéric Didier sur le registre filtré [Did07].

8.1 Présentation de l'attaque

On se place dans le modèle le plus général, présenté au chapitre précédent, de LFSRs combinés par une fonction booléenne prenant plusieurs entrées par LFSR. Notre but va être de retrouver l'état initial du registre R_1.

Le théorème central est le suivant :

Théorème 8.1. *Soit F une fonction booléenne équilibrée à n variables et k un entier positif. On définit*

$$P_x(F) = Pr\left(\sum_{i=1}^{2k} F(x_i) = 0 \Big| \sum_{i=1}^{2k} x_i = x\right).$$

On a alors

$$P_0(F) \geq \frac{1}{2}\left(1 + 2^{-n(k-1)}\right). \tag{8.1}$$

et P_0 se distingue des autres probabilités puisque :

$$\min_{x\neq 0}(P_0 - P_x) \geq \frac{1}{2^{n-1}}\left(\frac{\Delta_F}{2^{n-1}}\right)^k \tag{8.2}$$

où $\Delta_F = \frac{1}{2}\left(1 - \max_{u\neq 0}\left(\frac{\sigma_1(u)}{2^n}\right)\right)$.

On recommande en général de prendre des fonctions de filtrage avec de faibles coefficients d'auto-corrélation. Dans ce cas, la valeur de Δ_F sera très proche de $\frac{1}{2}$. La preuve de ce résultat peut être trouvée dans la thèse de Frédéric Didier [Did07]. Elle utilise en particulier la propriété suivante qui va nous être utile par la suite.

Proposition 8.2. *Les probabilités P_x peuvent s'écrire*

$$P_x = \frac{1}{2}\left(1 + \sum_{u \in \mathbb{F}_{2^n}} (-1)^{u \cdot x} \left(\frac{\widehat{F}(u)}{2^n}\right)^{2k}\right) \tag{8.3}$$

où $\widehat{F}$ est la transformée de Walsh de la fonction de filtrage.

Dans cette expression, u et x sont des valeurs vectorielles, et $u \cdot x$ désigne le produit scalaire. Ce résultat peut aussi être trouvé dans la thèse de Sabine Leveiller [Lev04], l'habilitation à diriger des recherches de Anne Canteaut [Can06] ou les travaux de Thomas Johanson [JJ00].

Nous allons maintenant pouvoir décrire notre attaque. On dispose d'une suite chiffrante $(F(x_t(I_0)))_{t \geq 0}$ obtenue à partir d'un état initial I_0 (on se reportera au chapitre 7 pour la présentation du modèle).

L'idée de base est simple : tester toutes les initialisations possibles ; seule celle produisant le biais espéré est la bonne. Malheureusement, il s'agit alors d'une recherche exhaustive. Pour que cette approche soit viable, nous allons profiter de l'indépendance des registres pour éliminer l'influence d'une partie d'entre eux. Si on veut limiter la recherche exhaustive au registre R_1, nous allons avoir besoin de trouver des équations de parité vérifiées par tous les autres registres, et indépendantes de l'initialisation. D'après la propriété 6.4, cela revient à calculer des multiples du plus petit multiple commun des polynômes $P_i(X)$ pour $2 \leq i \leq r$. De plus nous aurons besoin de multiples de poids pair $2k$.

Si $q(X) = \sum_{i=1}^{2k} X^{t_i}$ est un tel polynôme, alors on est assuré que

$$\sum_{i=1}^{2k} x_{t_i}(I) \in \begin{pmatrix} \star \\ 0 \end{pmatrix},$$

où $\begin{pmatrix} \star \\ 0 \end{pmatrix}$ désigne l'ensemble des vecteurs dont les $n - n_1$ derniers bits sont nuls (on ne précisera pas les dimensions afin d'alléger les notations).

Nous verrons dans le chapitre suivant comment calculer de telles équations de parité. Supposons pour l'instant que l'on sache en trouver autant que nécessaire.

Nous allons alors calculer le biais observé pour chaque initialisation en nous restreignant à ces positions. On observe alors la quantité suivante, dépendant de l'état initial I :

$$P(I) = Pr\left(\sum_{i=1}^{2k} F(x_{t_i}(I_0)) = 0 \middle| \sum_{i=1}^{2k} x_{t_i}(I) \in \begin{pmatrix} 0 \\ \star \end{pmatrix} \text{ et } \sum_{i=1}^{2k} x_{t_i}(I_0) \in \begin{pmatrix} \star \\ 0 \end{pmatrix}\right). \tag{8.4}$$

Dans cette expression, les termes $F(x_{t_i}(I_0))$ sont connus puisqu'il s'agit de la suite chiffrante. On peut s'assurer de $\sum_{i=1}^{2k} x_{t_i}(I_0) = \begin{pmatrix} \star \\ 0 \end{pmatrix}$ en ne considérant que les positions

correspondant à nos équations de parité. Enfin la condition $\sum_{i=1}^{2k} x_{t_i}(I) = \vec{0}$ dépend uniquement de notre hypothèse sur l'état initial du premier registre et on peut calculer cette valeur effectivement.

Si nous avons choisi le bon état initial alors $I = I_0$ et on retrouve finalement la probabilité P_0 et le biais attendu (8.1).

Si $I \neq I_0$, en considérant que la séquence d'états générés par le registre avec l'initialisation I et celle avec l'initialisation I_0 sont indépendantes, ce qui est légitime puisque, d'après la propriété 6.3,X leur corrélation est minimale, on obtient

$$P(I) = Pr\left(\sum_{i=1}^{2k} F(x_{t_i}(I_0)) = 0 \middle| \sum_{i=1}^{2k} x_{t_i}(I_0) \in \binom{\star}{0}\right) \tag{8.5}$$

$$= \frac{1}{2^{n_1}} \sum_{a \in \binom{\star}{0}} P_a \tag{8.6}$$

En utilisant l'expression (8.3), on a alors

$$\begin{aligned} P_0 - P(I) &= \frac{1}{2^{n_1}} \sum_{a \in \binom{\star}{0}} (P_0 - P_a) \\ &= \frac{1}{2^{n_1}} \sum_{a \in \binom{\star}{0}} \frac{1}{2} \sum_{u \in \mathbb{F}_{2^n}} \left(1 - (-1)^{(u \cdot a)}\right) \left(\frac{\widehat{F}(u)}{2^n}\right)^{2k} \\ &= \frac{1}{2^{n_1}} \sum_{a \in \binom{\star}{0}} \sum_{u \in \mathbb{F}_{2^n}} (u \cdot a) \left(\frac{\widehat{F}(u)}{2^n}\right)^{2k} \\ &= \sum_{u \in \mathbb{F}_{2^n}} \left(\frac{1}{2^{n_1}} \sum_{a \in \binom{\star}{0}} (u \cdot a)\right) \left(\frac{\widehat{F}(u)}{2^n}\right)^{2k}. \end{aligned}$$

Or, la somme $\sum_{a \in \binom{\star}{0}} (u \cdot a)$ vaut 0 si $u \in \binom{0}{\star}$, et $2^{n_1 - 1}$ sinon. Cela donne

$$P_0 - P(I) = \frac{1}{2} \sum_{u \in \mathbb{F}_{2^n} \setminus \binom{0}{\star}} \left(\frac{\widehat{F}(u)}{2^n}\right)^{2k}.$$

Notons S la valeur obtenue pour k égal à 1.

Le lemme 8.3 donné ci-dessous, avec $m = 2^n - 2^{n-n_1}$ et $s = 2S$, permet alors d'obtenir la borne suivante :

$$P_0 - P(I) \geq \frac{1}{2}\left(2^n - 2^{n-n_1}\right)\left(\frac{2S}{2^n - 2^{n-n_1}}\right)^k = \varepsilon. \tag{8.7}$$

En utilisant l'autre expression de P_a, on trouve

$$S = \frac{1}{2}\left(1 - \frac{1}{2^{n_1}} \sum_{a \in \binom{\star}{0}} \frac{\sigma_1(a)}{2^n}\right) \geq \Delta_F$$

Nous avons utilisé le lemme suivant dont la démonstration repose sur l'inégalité de Hölder et peut être trouvée dans la thèse de Frédéric Didier [Did07].

Lemme 8.3 (Somme de puissances)**.** *Étant donné n nombres réels positifs $a_1, \ldots, a_n$ de somme $s = \sum_{i=1}^{n} a_i$ et un entier $p > 0$, on a l'inégalité*

$$\sum_{i=1}^{n} a_i^p \geqslant \left(\frac{s}{n}\right)^p ;$$

avec égalité lorsque tous les a_i sont égaux à s/n.

Parmi les propriétés cryptographiques souhaitées pour la fonction de filtrage, on requiert souvent de faibles coefficients d'auto-corrélation. Dans ce cas, S sera proche de $1/2$, ce qui est le cas qui nous est le plus favorable.

Pour distinguer P_0 des probabilités $P(I)$ avec $I \neq I_0$, il faut disposer d'environ ε^{-2} réalisations de la variable aléatoire d'après les bornes de Tchernov. En prenant $S = 1/2$ on obtient un nombre d'équations de l'ordre de $2^{2(n+1)(k-1)+4}$.

Comme nous ne contrôlons pas *a priori* les n_1 premiers bits obtenus avec nos équations de parité et que nous ne gardons que celles où on obtient 0, nous n'en utilisons en moyenne qu'une sur 2^{n_1}. Nous avons donc besoin de précalculer $2^{n_1}\varepsilon^{-2}$ équations de parité. On notera $\mathcal{M}$ l'ensemble de ces multiples. La longueur de suite chiffrante dépends du plus haut degré de ces équations et nous verrons dans le chapitre consacré aux calculs de multiples de petits poids qu'elle est de l'ordre de 2.

À ce stade, nous avons déjà obtenu une attaque potentiellement efficace. Pour chaque initialisation, la tester demande $|\mathcal{M}|$ opérations. La complexité de la phase active est alors de $2^{r_1}|\mathcal{M}|$.

Nous allons voir maintenant comment utiliser une transformée de Walsh afin d'effectuer cette recherche exhaustive sur le premier registre de manière beaucoup plus efficace.

8.1.1 Estimation efficace des $P(I)$.

Pour chaque équation de parité m, nous noterons x_m le vecteur obtenu en sommant les vecteurs d'entrée aux positions correspondant à m, et s_m l'opposé de la valeur obtenue en sommant les bits de suite chiffrante à ces mêmes positions :

$$x_m(I) = \sum_{i=1}^{2k} x_{m_i}(I) \text{ et } s_m = 1 - \sum_{i=1}^{2k} F\left(x_{m_i}(I_0)\right).$$

Nous voulons alors calculer pour tout I

$$\widetilde{P}(I) = \sum_{m \in \mathcal{M}} \delta_{\vec{0}}(x_m(I)) s_m$$

et nous prendrons pour I_0 celui donnant la valeur maximale.

On peut alors voir notre problème comme un problème de décodage d'un code linéaire de très grande longueur et de faible dimension comme dans le cas des attaques par corrélation rapides classiques. En effet, pour chaque équation de parité, chaque bit du vecteur x_m dépend linéairement de l'initialisation puisque nos registres sont des LFSRs. Nous avons vu qu'il existe alors un vecteur E_m^j tel que le j-ième bit de x_m soit obtenu par simple produit scalaire (voir remarque 6.3) :

$$x_m^j(I) = \left\langle I | E_m^j \right\rangle.$$

Pour tout $j \in [1, v]$, les vecteurs

$$\left(x_m^j(I)\right)_{m\in\mathcal{M}}$$

sont donc des mots d'un code linéaire de dimension r_1 et de longueur $|\mathcal{M}|$. On cherche le mot le plus proche de notre suite chiffrante. Lorsque v est égal à 1, il s'agit donc d'un simple décodage sur le canal binaire symétrique. On sait alors qu'il est possible d'utiliser une transformée de Walsh pour réaliser ce décodage avec une complexité en $\mathcal{O}\left(|\mathcal{M}| + r_1 2^{r_1}\right)$. En effet,

$$\mathcal{F}\left[\sum_{m\in\mathcal{M}} s_m \delta_{E_m^j}\right](I) = \sum_{m\in\mathcal{M}} s_m (-1)^{\left\langle I|E_m^j\right\rangle} = -2\tilde{P}(I) + \sum_{m\in\mathcal{M}} s_m. \tag{8.8}$$

Pour v plus grand que 1 cependant, $\delta_{\vec{0}}(x_m)$ n'est plus directement obtenu de manière linéaire.

Notons E_m^u la somme des vecteurs E_m^j tels que le j-ème bit de u vaut 1. On remarque alors que

$$\frac{1}{2^{n_1}} \prod_{j=1}^{n_1} \left(1 + (-1)^{\left\langle I|E_m^j\right\rangle}\right) = \frac{1}{2^{n_1}} \sum_{u\in\{0,1\}^{n_1}} (-1)^{\langle I|E_m^u\rangle}. \tag{8.9}$$

Et cette valeur vaut 1 si pour tout j on a $\langle I|E_m^j\rangle = 0$, et 0 sinon. Ainsi,

$$\delta_{\vec{0}}(x_m(I)) = \frac{1}{2^{n_1}} \sum_{u\in\{0,1\}^{n_1}} (-1)^{\langle I|E_m^u\rangle}.$$

On obtient ainsi en utilisant (8.8), (8.9) et la linéarité de la transformée de Walsh :

$$\mathcal{F}\left[\frac{1}{2^{n_1}} \sum_{m\in\mathcal{M}} \sum_{u\in\{0,1\}^{n_1}} s_m \delta_{E_m^u}\right](I) = \tilde{P}(I).$$

La recherche du registre peut donc se faire à l'aide de l'algorithme suivant.

La complexité en mémoire est en $\mathcal{O}(2^{r_1})$ tandis que la complexité en temps est en $\mathcal{O}(2^{n_1}|\mathcal{M}| + r_1 2^{r_1})$, à comparer avec $\mathcal{O}(2^{r_1}|\mathcal{M}|)$ par la méthode naïve. Enfin, la taille de la suite chiffrante nécessaire dépend de notre capacité à trouver des équations de parité. Nous verrons qu'elle est asymptotiquement en $\mathcal{O}\left(k\left(|\mathcal{M}|2^{r-r_1}\right)^{1/k}\right)$.

En reprenant notre borne sur le biais attendu 8.2, pour $\Delta_F = 1/2$, cela donne une complexité de $\mathcal{O}\left(2^{v+2(n+1)(k-1)} + r_1 2^{r_1}\right)$. On voit en particulier que cette complexité augmente beaucoup avec k et on se limitera le plus souvent à $k = 2$, c'est-à-dire des équations de parité de poids 4.

Algorithme 5 Recherche de l'initialisation d'un registre

Entrées: La suite chiffrante z émise par des registres combinés. Suffisamment d'équations de parités.

Sorties: L'initialisation I_0 du registre.

Initialiser à 0 un tableau T de taille 2^{r_1}.
pour chaque équation de parité m **faire**
  $s_m = 1 - \bigoplus_{i=1}^{2k} z(m_i)$
  pour u de 1 à 2^{n_1} **faire**
    $T[E_m^u] = T[E_m^u] + s_m$
  fin pour
fin pour
/* À ce stade, $T[I] = \sum_{m \in \mathcal{M}} \sum_{u \in \{0,1\}^{n_1}} s_m \delta_{E_m^u}$. */
Calculer la transformée de Walsh de T
renvoyer l'indice contenant la valeur maximale de T

8.2 Complexité totale de l'attaque

Afin de retrouver l'état initial de tous nos registres, il suffit de recommencer la même attaque sur chaque registre. L'ordre dans lequel on attaque les différents registres est cependant important.

Les équations de parité nous servent à éliminer l'influence des variables prises dans des registres pour lesquels nous ne connaissons pas encore l'initialisation. Dans les phases suivantes de l'attaque, on n'aura donc pas besoin d'annuler l'influence d'un registre attaqué précédemment. Si le premier registre attaqué est le plus grand, la phase de précalcul sera plus simple et la longueur de suite chiffrante nécessaire pour monter l'attaque plus petite.

La complexité de la phase active de l'attaque correspond le plus souvent à la complexité de la plus grande transformée de Walsh que l'on effectue. Lorsque l'on connaît tous les registres sauf le dernier, presque toutes les entrées de notre fonction de filtrage sont connues et si le nombre d'entrées inconnues est suffisamment faible, il est probable que l'on puisse trouver l'initialisation du dernier registre de manière aisée sans avoir recours à la transformée de Walsh. Dans ce cas, on aura intérêt à attaquer le registre le plus grand en dernier.

8.3 Résultats expérimentaux

Nous avons effectué cette attaque sur un système comprenant trois registres de tailles respectives 29, 31 et 37. La fonction de filtrage utilisée prend $n = 9$ bits en entrée, à raison de 3 par registre. Elle présente de bonnes propriétés cryptographiques, en particulier elle est 3-résiliente, ce qui rend les attaques classiques inenvisageables.

On peut voir dans le tableau 8.1 que les biais observés sont proches des estimations théoriques.

Suivant le registre dont nous retrouvons l'initialisation en premier, nous avons proposé deux variantes de l'attaque, toutes deux basées sur des équations de poids 4. La première variante utilise 2^{25} bits de suite chiffrante, et retrouve les initialisations des registres en

Algorithme 6 Attaque de LFSRs combinés

Entrées: La suite chiffrante
Sorties: L'état initial

1: /* Précalcul */
2: **pour** i de 1 à r **faire**
3: Trouver suffisamment (environ 2^{2n+n_i+6}) de multiples de poids faibles 4 du produit des polynômes de rétroaction (cf. algorithme 9) des registres $i+1$ à r. On note $\mathcal{M}_i$ ces multiples.
4: Calculer les vecteurs d'équation E_t^j correspondant aux entrées j aux instants t
5: **fin pour**
6: /* Phase active */
7: **pour** i de 1 à r **faire**
8: Sélectionner dans $\mathcal{M}_i$ les équations de parité pour lesquelles les entrées prises dans les registres connus valent toutes 0.
9: Utiliser l'algorithme 5 pour retrouver l'état initial du registre i
10: **fin pour**
11: **renvoyer** l'état initial complet.

TAB. 8.1 – Biais observé pour une fonction cryptographique à 9 entrées

poids des équations	Positions libres	$\log_2(biais)$ (borne $S = 1/2$)	$\log_2(biais)$ (réel)
4	f_1	-9.80735	-9.13417
4	f_2	-9.80735	-9.04281
4	f_3	-9.80735	-9.08431
6	f_1	-18.6147	-16.8315
6	f_2	-18.6147	-16.7268
6	f_3	-18.6147	-16.7556

environ 7 minutes ; tandis que la deuxième variante utilise seulement $2^{22.6}$ bits de suite chiffrante mais nécessite quelques heures de calculs. Les temps de calculs cités ont été obtenus en utilisant une machine dotée d'un processeur Intel Core 2 Quad CPU Q9550 à 2.83GHz, en n'utilisant qu'un cœur et $2Go$ de mémoire vive.

L'attaque 1 minimise le temps de calcul. Elle retrouve l'initialisation des registres de tailles 29, 31 puis 37, dans cet ordre. Pour le premier registre, nous avons donc les paramètres suivants :

$$n_1 = 3; \quad n_2 = 6; \quad L_1 = 29; \quad L_2 = 68.$$

Le nombre d'équations nécessaire est alors $L_1 2^{2n+n_1+1} \simeq 2^{26.86}$ ce qui implique de disposer d'environ 2^{25} bits de suite chiffrante et plus précisément ici $3Mo$.

La seconde attaque minimise quant à elle la longueur de suite chiffrante. Elle retrouve l'initialisation des registres de taille 37, 29 puis 31, dans cet ordre. Pour le premier registre, nous avons donc les paramètres suivants :

$$n_1 = 3; \quad n_2 = 6; \quad L_1 = 37; \quad L_2 = 60.$$

Il suffit cette fois de disposer d'environ 2^{23} bits de suite chiffrante et plus précisément ici $985Ko$, pour obtenir les équations de parité. La recherche de l'état initial nécessite cette fois le calcul d'une transformée de Walsh de taille 2^{37} et est donc beaucoup plus longue.

Les deux tableaux suivants montrent les complexités des différentes parties de l'attaque.

	Précalcul	Phase active	longueur de suite chiffrante
Attaque 1	12min27s	7min01s	3.06Mo
Attaque 2	3min02s	6h18min	985Ko

TAB. 8.2 – Comparaison globale des deux attaques

	1er LFSR	2ème LFSR	3ème LFSR
Attaque 1	51s	6min10s	0s
Attaque 2	6h17min	1min27s	0s

TAB. 8.3 – Comparaison des différentes parties de la phase active

Chapitre 9

Multiples de poids faible d'un polynôme

Le calcul de multiples de poids faible de polynômes est un outil important et récurrent en cryptographie. En particulier, beaucoup d'attaques par corrélation ainsi que l'attaque présentée au chapitre précédent utilisent des relations de parité de poids faible vérifiées par les séquences générées par des LFSR et nous avons vu au chapitre 6 que cela correspond à des multiples de poids faible du polynôme de rétroaction. De nombreux travaux existent et il y a en fait toute une gamme d'algorithmes disponibles suivant les paramètres auxquels on se trouve confronté.

Nous allons ici présenter différentes méthodes, et plus particulièrement celles adaptées aux besoins de l'attaque présentée au chapitre précédent. Nous verrons que ces algorithmes reposent tous sur des compromis temps-mémoire pour la recherche de collision. L'algorithme les plus efficace est sans doute celui de Chose, Joux et Mitton [CJM02]. Nous présenterons aussi une approche essayant d'utiliser la structure multiplicative à travers le calcul de logarithmes discrets qui peut se révéler utile en pratique, et qui est spécialement adaptée dans le cas de registres combinés. Cette méthode a donné lieu à une publication en collaboration avec Frédéric Didier [DLC07].

Enfin, nous rappellerons brièvement les autres algorithmes existants lorsque les paramètres (degré et poids recherché, nombre de solutions souhaitées) sont d'un autre ordre, en précisant à chaque fois leur domaine d'utilisation.

9.1 Définition du problème

Nous nous limiterons ici au cas de polynômes à coefficients dans $\mathbb{F}_2$, même si la plupart des méthodes sont applicables sur d'autres corps.

Le problème que nous allons considérer peut être énoncé comme suit :

Définition 9.1 (Recherche de multiples de poids faible)**.**
Données : $P \in \mathbb{F}_2[X]$ un polynôme de degré n et trois entiers D, w et N.
But : trouver N *multiples* de P de degré strictement inférieur à D et de poids w.

Les différents paramètres qui vont influencer le choix de l'algorithme utilisé sont :
- le poids cible
- la différence $D - n$ entre le degré de P et le degré maximal autorisé
- le rapport entre N, le nombre de multiples souhaités et le nombre de multiples existants.

On peut estimer le nombre de polynômes solution à l'aide de l'heuristique suivante.

En supposant que les $Q(X) \pmod{P(X)}$ sont répartis de manière uniforme, un polynôme de poids w pris au hasard parmi les D^w possibles à une probabilité d'environ 2^{-n} d'être un multiple de P. Si le polynôme P est irréductible, cela donne une probabilité de 2^{-n}. Si w est négligeable devant D, on en déduit alors que le nombre de solutions est environ :

$$\frac{\binom{D}{w}}{2^n} \sim \frac{D^w}{w!2^n} \text{ pour } w \ll D.$$

Pour un poids donné, on espère donc obtenir les premiers polynômes solutions pour un degré de l'ordre de

$$D = \frac{w}{e} 2^{n/w}.$$

9.2 Trouver tous les multiples

Dans le cas des attaques par corrélation, le poids w souhaité sera très petit, typiquement 3, 4 ou 5. Le degré D devra donc être élevé. Comme celui-ci correspondra à la longueur de suite chiffrante nécessaire pour monter l'attaque, on aura intérêt à le prendre aussi petit que possible. On doit alors essayer de trouver tous les multiples satisfaisant ces contraintes.

Nous allons voir en détail les algorithmes adaptés à ce cas de figure.

9.2.1 Algorithme de Chose-Joux-Mitton

On cherche les multiples de P, c'est-à-dire les polynômes congrus à 0 modulo P. On considère alors la fonction de hachage qui à un polynôme Q associe $Q \pmod P$.

Nous utiliserons la notation suivante :

– $H[i] \leftarrow A$ désigne l'insertion dans une table de hachage de la valeur A indexée par la clé i.

Algorithme de base : compromis temps-mémoire

Dans cet algorithme, $\underline{A}_{i\ldots j}$ désigne le polynôme constitué des termes de degré i à j de A.

L'idée première est de considérer les polynômes candidats comme somme de deux polynômes :

$$1 + \sum_{i=1}^{w-1} X^{e_i} = 1 + \left(\sum_{i=1}^{w_1} X^{e_i}\right) + \left(\sum_{i=1}^{w_2} X^{e_{w_1+i}}\right),$$

avec $1 < e_1 < \cdots < e_{w-1} < D$ et $w_1 + w_2 = w - 1$.

Cela donne l'algorithme 7.

Énoncé ainsi, la complexité en mémoire est en $\mathcal{O}(D^{w_1})$ et la complexité en temps est en $\mathcal{O}(D^{w_2})$ et le compromis classique consiste à choisir $w_1 = \lfloor (w-1)/2 \rfloor$ et $w_2 = \lceil (w-1)/2 \rceil$.

Algorithme de Chose, Joux et Mitton

On peut en fait se contenter d'une complexité en mémoire beaucoup plus petite en suivant l'approche de Chose, Joux et Mitton [CJM02]. Il s'agit essentiellement de fixer un nombre h bien choisi de bits de la valeur que l'on souhaite obtenir lors de la collision.

Nous allons ici devoir découper les polynômes candidats en 4 parties et pour simplifier nous ne présentons l'algorithme que pour $w - 1 = 0 \pmod 4$.

Lors de la recherche de collisions, en moyenne, il n'y a qu'un élément par clé à cause du choix de h. Pour chaque α la complexité d'un passage dans la boucle ligne 11 est donc un $\mathcal{O}(2^h)$, le nombre de couples (u, v). Au total, on obtient donc une complexité en temps en $\mathcal{O}(2^{2h})$, soit $\mathcal{O}(D^{\lceil (w-1)/2 \rceil})$.

La complexité en mémoire est donc en $\mathcal{O}(D^{w_1})$ pour l'étape de mise en table, et $\mathcal{O}(2^h)$ pour l'étape de recherche de collision. Ces deux valeurs sont du même ordre et on trouve au final $\mathcal{O}(D^{\lceil (w-1)/4 \rceil})$.

Algorithme 7 Compromis temps-mémoire pour la recherche de multiples de poids faibles

Entrées: Un polynôme P, le poids cible w et le degré maximal D.
Sorties: Tous les multiples de P de poids w de degré inférieur à D.

1: /* Précalcul */
2: **pour** $0 < i < D$ **faire**
3: $Xmod[i] \leftarrow X^i \mod P$
4: **fin pour**
5: /* Mise en table */
6: Soient w_1 et w_2 deux entiers tels que $w = 1 + w_1 + w_2$ et $w_1 \leq w_2$.
7: **pour tout** w_1-uplets $A = (a_1, \ldots, a_{w_1})$ avec $0 < a_1 < \cdots < a_{w_1} < D$ **faire**
8:

$$H\left[\sum_{i=1}^{w_1} Xmod[a_i]\right] \leftarrow A$$

9: **fin pour**
10: /* Recherche de collisions */
11: **pour tout** w_2-uplet $B = (b_1, \ldots, b_{w_2})$ avec $0 < b_1 < \cdots < b_{w_2} < D$ **faire**
12: Calculer $1 + \sum_{i=1}^{w_2} Xmod[b_i]$. Si on trouve une collision A, alors

$$1 + \sum_{i=1}^{w_1} X^{a_i} + \sum_{i=1}^{w_2} X^{b_i}$$

est un multiple de P.
13: **fin pour**

On remarque que l'on va obtenir plusieurs fois chaque solution puisqu'il existe plusieurs découpages en 4 parties des w exposants. Si l'on veut obtenir une version efficace de l'algorithme, il faudra veiller à réduire autant que possible les calculs redondants. Dans le cas où w vaut 5, il est possible de ne considérer qu'une fois chaque polynôme mais nous ne détaillerons pas plus ici.

9.2.2 Utilisation du logarithme discret

Dans les algorithmes précédents, on utilise seulement la structure additive de l'anneau $\mathbb{F}_2[X]/ < P >$ mais on ne tient pas du tout compte de la structure multiplicative pourtant forte.

En pratique, les polynômes de rétroaction P utilisés pour les LFSRs sont le plus souvent irréductibles afin de garantir un période maximale. $\mathbb{F}_2[X]/ < P >$ est alors un corps et on peut définir un logarithme discret, que nous noterons Log.

Définition 9.2 (Logarithme discret)**.** Soit G un groupe monogène engendré par g. Le problème du logarithme discret est, étant donné un élément x de G, de retrouver l'entier i tel que $g^i = x$.

Il s'agit d'un problème difficile en général. Lorsque le groupe G considéré est le groupe multiplicatif d'un corps fini, il existe des algorithmes sous-exponentiels. Nous verrons cela plus en détail par la suite, mais voyons pour l'instant comment nous pouvons les utiliser pour calculer des multiples de petits poids.

Il s'agit d'une idée qui remonte à l'article peu connu [PK95]. Nous allons généraliser leur approche et en donner une analyse plus précise.

L'idée est toujours de considérer les polynômes candidats comme somme de deux polynômes, mais cette fois-ci le découpage se fait ainsi :

$$1 + \sum_{i=1}^{w-1} X^{e_i} = \left(1 + \sum_{i=1}^{w_1} X^{e_i}\right) + X^{e_{w_1+1}} \left(1 + \sum_{i=1}^{w_2} X^{e_{w_1+1+i}}\right),$$

avec $1 < e_1 < \cdots < e_{w-1} < D$ et $w_1 + w_2 = w - 2$.

On va alors chercher à exprimer la relation qui doit être vérifiée par les logarithmes $Log\left(1 + \sum_{i=1}^{w_1} X^{e_i}\right)$ et $Log\left(1 + \sum_{i=1}^{w_2} X^{e_{w_1+1+i}}\right)$.

Considérons un w_1-uplet

$$A = (a_1, \ldots, a_{w_1}) \text{ avec } 0 < a_1 < \cdots < a_{w_1} < D$$

et un w_2-uplet

$$B = (b_1, \ldots, b_{w_2}) \text{ avec } 0 < b_1 < \cdots < b_{w_2} < D.$$

On définit alors les logarithmes discrets

$$L_A = Log\left(1 + \sum_{i=1}^{w_1} X^{a_i}\right) \text{ et } L_B = Log\left(1 + \sum_{i=1}^{w_2} X^{b_i}\right),$$

ainsi que

$$e(A, B) = L_A - L_B \pmod{2^n}.$$

On a alors deux multiples de P de poids au plus w :

$$\left(1 + \sum_{i=1}^{w_1} X^{a_i}\right) + X^{e(A,B)} \left(1 + \sum_{i=1}^{w_2} X^{b_i}\right)$$

et

$$X^{e(B,A)} \left(1 + \sum_{i=1}^{w_1} X^{a_i}\right) + \left(1 + \sum_{i=1}^{w_2} X^{b_i}\right).$$

Ces deux multiples sont de coefficients constant 1, sauf si $e(A, B)$ est égal à zéro. Dans ce cas, on obtient un multiple de P de poids inférieur à w qu'il faut diviser par une puissance de X pour revenir à une forme standard.

L'un de ces deux multiples est de degré inférieur à D si l'une des conditions suivantes est vérifiée :

$$e(A, B) + b_{w_2} < D, \tag{9.1}$$
$$e(B, A) + a_{w_1} < D. \tag{9.2}$$

On peut remarquer que ces relations s'excluent l'une l'autre lorsque $D < 2^{n-1}$.

Enfin la notation $A_{i\ldots j}$ signifie maintenant l'entier constitué des bits i à j de l'entier A. On obtient ainsi l'algorithme 9.

Si l'on ne tient pas compte de la complexité du calcul des logarithmes discrets, la complexité en mémoire est en $\mathcal{O}(D^{w_1})$ pour la table de hachage. La complexité en temps est dominée par la boucle ligne 11. Le fait d'indexer la table de hachage par les bits de poids fort du logarithme permet de regrouper les w_1-uplets appartenant à des intervalles de taille environ D. Le nombre moyen d'éléments par clé est alors $\frac{\binom{D}{w_1}}{2^n/D}$. Pour des valeurs de D raisonnables, cela permet d'effectuer la recherche d'éléments satisfaisant les équations (9.1) et (9.2) en $\mathcal{O}(1)$. La complexité en temps est alors en $\mathcal{O}(D^{w_2})$.

Il est encore une fois usuel de chercher à équilibrer les complexités en choisissant $w_1 = \left\lfloor \frac{w-2}{2} \right\rfloor$ et $w_2 = \left\lceil \frac{w-2}{2} \right\rceil$.

Notons L la complexité de calcul d'un logarithme discret dans notre corps. On peut comparer la complexité des différentes méthodes dont nous avons parlé.

	$w = 2p$		$w = 2p + 1$	
Algorithme	**Temps**	**Mémoire**	**Temps**	**Mémoire**
TMTO (Algo. 7)	D^p	D^{p-1}	D^p	D^p
CJM (Algo. 8)	D^p	$D^{\lceil p/2 \rceil}$	D^p	$D^{\lceil p/2 \rceil}$
LogTMTO (Algo. 9)	$D^{p-1}L$	D^{p-1}	D^pL	D^{p-1}

TAB. 9.1 – Comparaison des méthodes à base de hachage pour le calcul de multiples de petit poids

On voit donc que dans le cas d'un poids w impair, l'algorithme 8 de Chose, Joux et Mitton est toujours le meilleur. Pour w pair par contre, si $L < D$, notre algorithme devient compétitif. Pour des longueurs de LFSRs utilisées en pratique cela est souvent le cas. Il nécessite cependant beaucoup plus de mémoire et ne sera intéressant en pratique que pour w valant 4 ou éventuellement 6. C'est tout de même très utile puisque les multiples de poids 4 ont une importance particulière pour les attaques par corrélation comme nous l'avons vu au chapitre 8.

Enfin, comme pour les algorithmes précédents, nous risquons d'obtenir plusieurs fois le même multiple puisqu'il existe plusieurs découpages possibles. Afin de limiter cet effet et d'accélérer un peu la recherche, on peut utiliser le lemme suivant :

Lemme 9.3. *Soit $m(X)$ un polynôme de poids $2 + w_1 + w_2$ et de degré inférieur à D. Alors il existe e et deux polynômes $a(X)$ et $b(X)$ de poids respectifs w_1 et w_2 et de degrés*

respectifs au plus D *et* $Dw_2/(w-1)$ *tels que*

$$m(X) = (1 + a(X)) + X^e\,(1 + b(X))$$

ou

$$m(X) = X^e\,(1 + a(X)) + (1 + b(X))\,.$$

Ainsi, à la ligne 11 de l'algorithme 9, on peut se contenter de regarder les w_2-uplets avec $b_{w_2} < Dw_2/(w-1)$.

Remarques sur les calculs de logarithmes discrets

Le calcul de logarithmes discrets sur les corps finis est un problème difficile et cela limite l'interêt de cette méthode. Cependant en pratique les registres utilisés seront de taille réduite.

En utilisant les algorithmes classiques "pas de bébé, pas de géants" et Pohlig-Hellman, on peut par exemple calculer très efficacement les logarithmes discrets dans les corps $\mathbb{F}_{2^n}$ pour tout les n jusqu'à 78 sauf

$$\{31, 49, 59, 61, 62, 65, 67, 69, 73, 74, 77\}.$$

En effet, pour ces valeurs, on peut mettre entièrement en table les groupes nécessaires au calcul de logarithme discret.

D'autres valeurs donnant lieu à des groupes particulièrement friables sont aussi envisageables, en particulier $\{96, 110, 156, 210\}$.

De plus, si l'on fixe le poids $w = 4$, le degré minimal pour obtenir des multiples est approximativement $D = \frac{4}{e}2^{n/4}$. Il croît donc de manière exponentielle avec n, or pour le calcul de logarithme discret, il existe des algorithmes sous exponentiels. Ainsi, asymptotiquement, la complexité DL est dans ce cas inférieure à D^2.

Cas où P n'est pas irréductible

Il est intéressant de noter que si le polynôme P est un produit de plusieurs facteurs irréductibles premiers entre eux deux à deux $P_1, \ldots, P_k$, il suffit de se placer dans le groupe produit $F_2[X]/ < P_1 > \times \cdots \times F_2[X]/ < P_k >$. Le calcul du logarithme discret se ramène alors via le théorème des restes chinois à un calcul de logarithme discret dans chacun des groupes $F_2[X]/ < P_i >$. Pour un même degré n, il semble donc beaucoup plus facile de trouver des multiples de poids faible d'un polynôme très friable que d'un polynôme irréductible, ce qui n'apparaît pas du tout avec les autres algorithmes.

Ce cas est d'autant plus significatif que c'est précisément dans ce cadre là que l'on se place pour l'attaque des LFSRs combinés du chapitre 8.

9.2.3 En degré faible

Enfin pour être exhaustif, nous citerons aussi le cas où le degré recherché est petit, typiquement de l'ordre de 2 fois le degré de notre polynôme de départ. Dans ce cas, on utilisera des techniques d'énumération de mots de poids minimum d'un code linéaire quelconque comme l'algorithme de Ziemmerman [BFK+98]. Cependant ce cas n'étant pas celui qui nous intéresse ici, nous ne le détaillerons pas plus.

9.3 Le cas d'un seul multiple

9.3.1 Recherche de collisions

Les algorithmes 7, 8 et 9 nous donnent tous les multiples de poids w de degré inférieur à D. Dans le cas où l'on ne souhaiterait en obtenir qu'un seul, les équilibrages des phases de mise en table et de recherche se feraient différemment.

Avant de voir les complexités obtenues, nous allons donner un algorithme très simple basé sur le paradoxe des anniversaires auquel nous pourrons nous comparer. Cet algorithme générique est décrit dans l'algorithme 10.

L'espace des clés de notre fonction de hachage est de taille 2^n, si l'on considère que la répartition est uniforme, la première collision arrive en moyenne au bout de $\mathcal{O}(2^{n/2})$ insertions. La complexité en temps et en mémoire est donc en $\mathcal{O}(2^{n/2})$ quels que soient le poids et le degré recherchés si une solution existe. Il est important de noter que les éléments doivent être parcourus dans un ordre aléatoire, et on doit de plus avoir $\binom{D}{w} \gg 2^n$.

En utilisant l'algorithme de Chose, Joux et Mitton, la première collision sera trouvée en moyenne après avoir inséré $2^{(n-h)/2}$ valeurs soit environ $\sqrt{2^n/D^{w_1}}$. On cherchera donc à équilibrer la phase de mise en table et la phase de recherche de collisions, et donc $\sqrt{2^n/D^{w_1}}$ et D^{w_1}. Cela nous donne w_1 de l'ordre de $\frac{n}{3\log_2(D)}$. On obtient alors une complexité en mémoire et en temps en $\mathcal{O}(2^{n/3})$ ce qui représente une nette amélioration par rapport à l'algorithme 10.

Il faut cependant noter que dans le cas où il existe de nombreux multiples, c'est-à-dire $\binom{D}{w} \gg 2^n$, l'algorithme le plus performant est celui décrit par Wagner dans [Wag02] utilisant le paradoxe des anniversaires généralisé dont nous ne détaillons pas ici l'analyse de complexité.

Enfin, avec l'approche basée sur les logarithmes discrets, le premier multiple sera trouvé après avoir testé en moyenne $2^n/D^{w_1+1}$ valeurs pour B. On cherchera donc à équilibrer D^{w_1} et $2^n/D^{w_1+1}$ ce qui donne une complexité en temps et en mémoire en $\mathcal{O}\left(\frac{2^{n/2}}{\sqrt{D}}\right)$. Cela ne peut être intéressant par rapport à l'algorithme de Wagner que pour un degré D très élevé

$$D > 2^{n\frac{\log_2 w-1}{\log_2 w+1}}.$$

TAB. 9.2 – Comparaison des algorithmes de recherche d'un seul multiple de petit poids

Algorithme	Complexité
Anniv. (Algo. 10)	$2^{n/2}$
CJM (Algo. 8)	$2^{n/3}$
logTMTO (Algo. 9)	$2^{n/2}/\sqrt{D}$
Wagner	$2^{n/(1+\lfloor\log_2(w)\rfloor)}$

9.3.2 Algorithme de Canteaut-Chabaud

Pour terminer, dans le cas où le degré cherché est faible, la méthode à privilégier est encore une fois différente et l'on choisira plutôt cette fois-ci l'algorithme de Canteaut et Chabaud [CC98].

Considérons la matrice $M \in \mathcal{M}_{D,D-n}$ dont la i-ème ligne est formée des coefficients de $X^i P$.

$$M = \begin{pmatrix} P \\ XP \\ X^2P \\ \cdots \\ X^{D-n-1}P \end{pmatrix}$$

Un multiple de poids w de P correspond alors à un mot de poids w dans le code de matrice génératrice M. On peut par ailleurs généralement considérer que les propriétés de ce code sont proches de celles d'un code binaire aléatoire. Lorsque le poids des mots recherchés est proche du poids minimum de ce code, on aura intérêt à utiliser un algorithme adapté comme celui de Canteaut Chabaud que nous présentons ici brièvement. Il s'agit d'une amélioration de l'algorithme de Stern [Ste88].

Ce sera l'algorithme de choix quand D est peu différent de n (typiquement $D = 2n$) et que l'on recherche un mot de poids minimal. L'évaluation précise de la complexité de cet algorithme nécessite alors l'étude d'une chaîne de Markov. Le lecteur intéressé pourra se reporter à l'article original [CC98].

9.4 Conclusion

La recherche de mot de petit poids dans un code linéaire est un problème de base de la théorie des codes. Nous avons vu que la recherche d'un multiple de petit poids d'un polynôme peut être réalisée en utilisant les mêmes méthodes. Cependant, il semble dommage que la plupart des méthodes utilisées n'utilisent pas la spécificité de ce problème, à savoir la structure multiplicative. C'est ce que nous avons tenté d'apporter en proposant d'utiliser les logarithmes discrets dans le groupe produit.

Algorithme 8 Algorithme de Chose, Joux, Mitton

Entrées: Un polynôme P, le poids cible w et le degré maximal D.
Sorties: Tous les multiples de P de poids w de degré inférieur à D.

/* Précalcul */
pour $0 < i < D$ **faire**
  $Xmod[i] \leftarrow X^i \mod P$
fin pour
/* Mise en table */
Soit $w_1 = \lceil (w-1)/4 \rceil$ et $h = \log_2 \left(\binom{D}{w_1} \right)$.
pour tout w_1-uplet $A = (a_1, \ldots, a_{w_1})$ avec $0 < a_1 < \cdots < a_{w_1} < D$ **faire**

$$H_1 \left[\underbrace{\sum_{i=1}^{w_1} Xmod[a_i]}_{0 \ldots h} \right] \leftarrow \left(\sum_{i=1}^{w_1} Xmod[a_i], A \right).$$

fin pour
/* Recherche de collisions */
pour tout $\alpha \in \mathbb{F}_2[X]$ de degré inférieur à h **faire**
  vider H_2
  pour tout $(u, v) \in \mathbb{F}_2[X]^2$ de degré inférieur à h tels que $u + v = \alpha + 1$ **faire**
    pour tout $(\sum_{i=1}^{w_1} Xmod[a_i], A) \in H_1[u]$ **faire**
      pour tout $(\sum_{i=1}^{w_1} Xmod[b_i], B) \in H_1[v]$ **faire**

$$H_2 \left[\underbrace{\sum_{i=1}^{w_1} Xmod[a_i] + Xmod[b_i]}_{h+1 \ldots n} \right] \leftarrow (A, B)$$

        Si on trouve une collision avec un élément (C, D), alors

$$\sum_{i=1}^{w_1} X^{a_i} + X^{b_i} + X^{c_i} + X^{d_i}$$

        est un multiple de P.
      fin pour
    fin pour
  fin pour
fin pour

Algorithme 9 Utilisation du logarithme discret pour la recherche de multiples de petit poids [DLC07]

Entrées: Un polynôme P, le poids cible w et le degré maximal D.
Sorties: Tous les multiples de P de poids w de degré inférieur à D.

1: /* Précalcul */
2: **pour** $0 < i < D$ **faire**
3: $\quad Xmod[i] \leftarrow X^i \mod P$
4: **fin pour**
5: /* Mise en table */
6: Soient w_1 et w_2 deux entiers tels que $w = 2 + w_1 + w_2$ et $w_1 \leq w_2$.
7: **pour tout** w_1-uplet $A = (a_1, \ldots, a_{w_1})$ avec $0 < a_1 < \cdots < a_{w_1} < D$ **faire**
8:

$$H\left[\underline{L_A}_{\log_2(D)\ldots n-1}\right] \leftarrow (L_A, A)$$

9: **fin pour**
10: /* Recherche de collisions */
11: **pour tout** w_2-uplet $B = (b_1, \ldots, b_{w_2})$ avec $0 < b_1 < \cdots < b_{w_2} < D$ **faire**
12: $\quad$ Calculer L_B et chercher dans la table de hachage H des éléments satisfaisants (9.1) ou (9.2).
13: $\quad$ Si on en trouve, sortir les multiples correspondants.
14: **fin pour**

Algorithme 10 Recherche d'un multiple de petit poids à l'aide du paradoxe des anniversaires

Entrées: Un polynôme P, le poids cible w et le degré maximal D.
Sorties: Un seul multiple de P de poids w de degré inférieur à D.

/* Précalcul */
pour $0 < i < D$ **faire**
$\quad Xmod[i] \leftarrow X^i \mod P$
fin pour
/* Recherche de collisions */
Soit $w_1 = \lceil (w-1)/2 \rceil$.
pour tout w_1-uplet $A = (a_1, \ldots, a_{w_1})$ *pris au hasard*
avec $0 < a_1 < \cdots < a_{w_1} < D$ **faire**

$$H\left[\sum_{i=1}^{w_1} Xmod[a_i]\right] \leftarrow A$$

S'il y a une collision avec un élément B, alors

$$1 + \sum_{i=1}^{w_1} X^{a_i} + \sum_{i=1}^{w_1} X^{b_i}$$

est un multiple de P.
fin pour

Algorithme 11 Algorithme de Canteaut Chabaud

Entrées: Une matrice génératrice M, trois paramètres h, p et w.
Sorties: Un mot $m \in \mathcal{C}(m)$ de poids inférieur à w.

Choisir aléatoirement un ensemble d'information I.
Appliquer une élimination de Gauss pour obtenir la matrice systématique $S_I = (I_k|Z_I)$.

répéter
Séparer aléatoirement S_I en deux ensembles S_1 et S_2 de lignes de tailles $\lceil k/2 \rceil$ et $\lfloor k/2 \rfloor$.
Choisir aléatoirement un ensemble H de taille h de positions hors de I.
Trouver m, parmi toutes les combinaisons linéaires de p lignes de S_1 et p lignes de S_2 nulles aux positions de H, qui soit de poids minimum.
Échanger au hasard une position de I et une position hors de I et mettre S_I à jour.
jusqu'à $poids(m) \leq w$

Annexe A

Test des polynômes de permutation

Algorithme 12 Test probabiliste simple

```
TestProbabilisteSimple:=function(P,eps)
 R<x> := Parent(P);
 F := BaseRing(R);
 n := Degree(P);
 r := Ceiling( 2*n*Log(eps^-1) );
 random_set := [Random(F): i in [1..r]]
 res := not exists { a :
         a in random_set |
         Degree(Gcd(p-a,Modexp(X,#F,P-a)-X)) ne 1
        };
 return res;
end function;
```

Algorithme 13 Représentation Euclidienne (non-optimal)

```
EuclidianRepresentation:=function(a0,a1)
 res:=[];
 A:=a0;
 B:=a1;
 while (B ne 0) do
  temp:=B;
  Q,B:=Quotrem(A,B);
  A:=temp;
  Append(~res,Q);
 end while;
 Append(~res,A);
 return res;
end function;
```

Algorithme 14 Test probabiliste quasi-linéaire

```
TestProbabilisteQuasiLineaire:=function(f,eps)
 K:=BaseRing(Parent(f));
 q:=#K;
 m:=1+Ceiling(Log(q,2/eps));
 print "m: ",m;
 F:=ext<K|m>;
 R<X>:=PolynomialRing(F);
 u:=Random(F);
 a0:=X^q-X;
 a1:=(R!f)-u;
 a2:=Modexp(X,q,a1)-X;
 Eucl:=EuclidianRepresentation(a1,a2);
 if Degree(Eucl[#Eucl]) ge 1 then return true; end if;
 n:=[ (i ge 2) select (Self(i)-Degree(Eucl[i-1]))
       else ((i eq 1) select Degree(f)
       else #K ) : i in [0..#Eucl] ];
 s:=(&+[(n[i]*n[i+1] mod 2): i in [1..#Eucl]]) mod 2;
 L:=[LeadingCoefficient(Eucl[i]):i in [1..#Eucl]];
 v:=(-1)^s*&*[L[i-1]^(-(n[i]+n[i+1])):i in [2..#Eucl]]-(-1)^(#K)*(u^q-u);
 if (v eq 0) then return true; else return false; end if;
end function;
```

Bibliographie

[AAW08] Amir Akbary, Sean Alaric, and Qiang Wang. On some classes of permutation polynomials. *Int. J. Number Theory*, 4(1) :121–133, 2008. 3.4

[AGW08] Amin Akbary, Dragos Ghioca, and Qiang Wang. On permutation polynomials of prescribed shape. *à paraître dans Finite Fields and their Applications*, Décembre 2008. 4.1

[AW06] Amir Akbary and Qiang Wang. A generalized Lucas sequence and permutation binomials. *Proc. Amer. Math. Soc.*, 134(1) :15–22 (electronic), 2006. 3.4, 3.5.1

[AW07] Amir Akbary and Qiang Wang. On polynomials of the form $x^r f(x^{(q-1)/l})$. *Int. J. Math. Math. Sci.*, pages Art. ID 23408, 7, 2007. 3.4

[BBL95] Daniel Bleichenbacher, Wieb Bosma, and Arjen K. Lenstra. Some remarks on Lucas-based cryptosystems. In *Advances in cryptology—CRYPTO '95 (Santa Barbara, CA, 1995)*, volume 963 of *Lecture Notes in Comput. Sci.*, pages 386–396. Springer, Berlin, 1995. 3.6

[BCCLC05] T.P. Berger, Anne Canteaut, Pascale Charpin, and Yann Laigle-Chapuy. On almost perfect nonlinear mappings. In *IEEE International Symposium on Information Theory*, pages 2002–2006, septembre 2005. (document)

[BCCLC06] Thierry P. Berger, Anne Canteaut, Pascale Charpin, and Yann Laigle-Chapuy. On almost perfect nonlinear functions over $\mathbb{F}_{2^n}$. *IEEE Transactions on Information Theory*, 52(9) :4160–4170, 2006. (document), 2.2.1, 5.5, 5.2.2

[BCHO99] Aart Blokhuis, Robert S. Coulter, Marie Henderson, and Christine M. O'Keefe. Permutations amongst the Dembowski-Ostrom polynomials. In Dieter Jungnickel and Harald Niederreiter, editors, *Fifth International Conference on Finite Fields and Applications*. Springer, 1999. 3.3.2, 3.3.2

[BCL08] Lilya Budaghyan, Claude Carlet, and Gregor Leander. Two classes of quadratic APN binomials inequivalent to power functions. *IEEE Transactions on Information Theory*, 54(9) :4218–4229, 2008. 3.4

[BFK+98] A. Betten, H. Fripertinger, A. Kerber, A. Wassermann, and K.-H. Zimmermann. *Codierungstheorie-Konstruktion und Anwendung linearer Codes*. Springer-Verlag, 1998. 9.2.3

[BL77] Joel V. Brawley and Jack Levine. Some cryptographic applications of permutation polynomials. *Cryptologia*, 1(1) :76,92, 1977. 5.1

[BM75] Allan Borodin and Ian Munro. *The computational complexity of algebraic and numeric problems*. American Elsevier Publishing Co., Inc., New York-

London-Amsterdam, 1975. Elsevier Computer Science Library ; Theory of Computation Series, No. 1. 2.2.2

[Can06] Anne Canteaut. Analyse et conception de chiffrements à clé secrète, 2006. Habilitation à diriger des recherches. 8.1

[Car62] Leonard Carlitz. Some theorems on permutation polynomials. *Bull. Amer. Math. Soc.*, 68 :120–122, 1962. 3.5.1

[Cav63] S. R. Cavior. A note on octic permutation polynomials. *Math. Comp.*, 17 :450–452, 1963. 3.1

[CC98] Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code : Application to mceliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1) :367–378, 1998. 9.3.2, 9.3.2

[CDLCT07] Anne Canteaut, Frédéric Didier, Yann Laigle-Chapuy, and Jean-Pierre Tillich. Veille technologique dans le domaine du décodage itératif, *contrat de recherche CELAR*, novembre 2007. (document), II

[CF95] Stephen D. Cohen and Michael D. Fried. Lenstra's proof of the Carlitz-Wan conjecture on exceptional polynomials : an elementary version. *Finite Fields Appl.*, 1(3) :372–375, 1995. I, 1.2

[CH04] Robert Coulter and Marie Henderson. A note on the roots of trinomials over a finite field. *Bull. Austral. Math. Soc.*, 69 :429–432, 2004. 3.4

[Che] William Cherowitzo. Hyperoval page. `http://www.math.cudenver.edu/~wcherowi/research/hyperoval/hypero.html`. I, 3.7.2

[Cho88] Wun Seng Chou. Binomial permutations of finite fields. *Bull. Austral. Math. Soc.*, 38(3) :325–327, 1988. 3.1

[CJM02] Philippe Chose, Antoine Joux, and Michel Mitton. Fast correlation attacks : An algorithmic point of view. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 209–221. Springer, 2002. 9, 9.2.1

[CJS00] Vladimor V. Chepyzhov, Thomas Johansson, and Ben J. M. Smeets. A simple algorithm for fast correlation attacks on stream ciphers. In Bruce Schneier, editor, *FSE*, volume 1978 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2000. 7.3

[CK08] Pascale Charpin and Gohar M. M. Kyureghyan. Cubic Monomial Bent Functions : A Subclass of M. *SIAM J. Discrete Math.*, 22(2) :650–665, 2008. 3.5.1, 5.2.2

[CM97] Robert S. Coulter and Rex W. Matthews. Planar functions and planes of Lenz-Barlotti class II. *Des. Codes Cryptogr.*, 10(2) :167–184, 1997. I

[Coh90] Stephen D. Cohen. Proof of a conjecture of Chowla and Zassenhaus on permutation polynomials. *Canad. Math. Bull.*, 33(2) :230–234, 1990. I, 4.3

[Coh91] Stephen D. Cohen. Permutation polynomials and primitive permutation groups. *Arch. Math. (Basel)*, 57(5) :417–423, 1991. 1.2

[Coh94] Stephen D. Cohen. Dickson polynomials of the second kind that are permutations. *Canad. J. Math.*, 46 :225–238, 1994. I, 3.6

[CT00] Anne Canteaut and Michaël Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In *EUROCRYPT*, pages 573–588, 2000. 7.2

[CV07] Guilhem Castagnos and Damien Vergnaud. Trapdoor permutation polynomials of $\mathbb{Z}/n\mathbb{Z}$ and public key cryptosystems. In Juan A. Garay, Arjen K. Lenstra, Masahiro Mambo, and René Peralta, editors, *ISC*, volume 4779 of *Lecture Notes in Computer Science*, pages 333–350. Springer, 2007. 5.1.1

[DGG99] Zong Duo Dai, Solomon W. Golomb, and Guang Gong. Generating all linear orthomorphisms without repetition. *Discrete Math.*, 205(1-3) :47–55, 1999. 3.2, 4.3

[DHK04] Wiebke S. Diestelkamp, Stephen G. Hartke, and Rachael H. Kenney. On the degree of local permutation polynomials. *J. Combin. Math. Combin. Comput.*, 50 :129–140, 2004. I, 5.2.3

[Dic97] Leonard Eugene Dickson. The analytic representation of substitutions on a power of a prime number of letters with a discussion of the linear group. *Ann. of Math.*, 11(1-6) :161–183, 1896-97. I

[Dic 7] Leonard Eugene Dickson. Analytic representation of substitution. *Ann. of Math.*, 11(3-4) :65–120, 1896-7. 3.1

[Did07] Frédéric Didier. *Codes de Reed-Muller et cryptanalyse du registre filtré*. PhD thesis, École Polytechnique, décembre 2007. 8, 8.1, 8.1

[DLC07] Frédéric Didier and Yann Laigle-Chapuy. Finding low-weight polynomial multiples using discrete logarithm. In *IEEE International Symposium on Information Theory*, juin 2007. (document), II, 9, 9, A

[Dob99] Hans Dobbertin. Kasami power functions, permutation polynomials and cyclic difference sets. In *Difference sets, sequences and their correlation properties (Bad Windsheim, 1998)*, volume 542 of *NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci.*, pages 133–158. Kluwer Acad. Publ., Dordrecht, 1999. 3.10

[Dob02] Hans Dobbertin. Uniformly representable permutation polynomials. In *Sequences and their applications (Bergen, 2001)*, Discrete Math. Theor. Comput. Sci. (Lond.), pages 1–22. Springer, London, 2002. 3.4, 5.2.1

[EGN92] Ronald J. Evans, John Greene, and Harald Niederreiter. Linearized polynomials and permutation polynomials of finite fields. *Michigan Math. J.*, 39(3) :405–413, 1992. I

[FGS93] Michael D. Fried, Robert Guralnick, and Jan Saxl. Schur covers and Carlitz's conjecture. *Israel J. Math.*, 82(1-3) :157–225, 1993. I, 1.2, 1.2, 3.8

[Fri70] Michael Fried. On a conjecture of Schur. *Michigan Mathematical Journal*, 17(1) :41–55, 1970. 3.6

[Gal62] Robert Gray Gallager. Low density parity check codes. *Transactions of the IRE Professional Group on Information Theory*, IT-8 :21–28, Janvier 1962. 7.3

[GG03] Joachim Von Zur Gathen and Jurgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, New York, NY, USA, 2003. 2.1, 2.4.2, 2.4.2

[GKL04] Shuhong Gao, Erich Kaltofen, and Alan G. B. Lauder. Deterministic distinct-degree factorization of polynomials over finite fields. *J. Symb. Comput.*, 38(6) :1461–1470, 2004. 2.3

[GMZ] Robert M Guralnick, P Müller, and Michael E. Zieve. Exceptional polynomials of affine type, revisited. en préparation. 3.8

[GRZ] Robert M. Guralnick, Joel E. Rosenbergand, and Michael E. Zieve. A new family of exceptional polynomials in characteristic two. *Annals of Mathematics*. to appear. 1.2, 3.8

[GZ] Robert M. Guralnick and Michael E. Zieve. Polynomials with PSL(2) monodromy. *Annals of Mathematics*. to appear. 1.2, 3.8

[Hay67] D. R. Hayes. A geometric approach to permutation polynomials over a finite field. *Duke Math. J.*, 34 :293–305, 1967. 1.2

[Her63] Charles Hermite. Sur les fonctions de sept lettres. *C. R. Acad. Sci. Paris*, 57 :750–757, 1863. I

[HM95] Marie Henderson and Rex Mathews. Permutation properties of chebyshev polynomials of the second kind over a finite field. *Finite Fields Appl*, 1 :115–124, 1995. I, 3.6

[HM98] Marie Henderson and Rex Mathews. Dickson polynomials of the second kind which are permutation polynomials over a finite field. *New Zealand J. Math.*, 27 :227–244, 1998. I, 3.6

[JJ00] Thomas Johansson and Fredrik Jönsson. Fast correlation attacks through reconstruction of linear polynomials. In Mihir Bellare, editor, *CRYPTO*, volume 1880 of *Lecture Notes in Computer Science*, pages 300–315. Springer, 2000. 7.3, 8.1

[JJ02] Fredrik Jönsson and Thomas Johansson. A fast correlation attack on LILI-128. *Inf. Process. Lett.*, 81(3) :127–132, 2002. 7.3

[JQ96] Marc Joye and Jean-Jacques Quisquater. Efficient computation of full Lucas sequences. *Electronic Letters*, 32(6) :537–538, 1996. 5.1.1

[Kay05] Neeraj Kayal. Recognizing permutation functions in polynomial time. `http://eccc.hpi-web.de/eccc-reports/2005/TR05-008`, 2005. 2, 2.3, 2.5

[KP02] Sergei Konyagin and Francesco Pappalardi. Enumerating permutation polynomials over finite fields by degree. *Finite Fields Appl.*, 8(4) :548–553, 2002. 4.8

[KP06] Sergei Konyagin and Francesco Pappalardi. Enumerating permutation polynomials over finite fields by degree. II. *Finite Fields Appl.*, 12(1) :26–37, 2006. 4.7

[Lau07] Cédric Lauradoux. *Conception et synthèse en cryptographie symétrique*. PhD thesis, École Polytechnique, 2007. 6

[LC07a] Yann Laigle-Chapuy. A note on a class of quadratic permutations over $\mathbb{F}_{2^n}$. In Serdar Boztas and Hsiao feng Lu, editors, *AAECC*, volume 4851 of *Lecture Notes in Computer Science*, pages 130–137. Springer, 2007. (document), I, 3.3.1, 3.15

[LC07b] Yann Laigle-Chapuy. Permutation polynomials and applications to coding theory. *Finite Fields Appl.*, 13(1) :58–70, 2007. (document), I, 3.4, 3.5, 3.5.1, 4.1, 4.1

[Lev04] Sabine Leveiller. *Quelques algorithmes de cryptanalyse du registre filtré*. PhD thesis, École Nationale Supérieure des Télécommunications, 2004. II, 8, 8.1

[Lid85] Rudolf Lidl. On cryptosystems based on polynomials and finite fields. In *Advances in cryptology (Paris, 1984)*, volume 209 of *Lecture Notes in Comput. Sci.*, pages 10–15. Springer, Berlin, 1985. I

[LJZ96] Hendrik W. Lenstra Jr. and Michael E. Zieve. A family of exceptional polynomials in characteristic three. In *FFA'95 : Proceedings of the third international conference on Finite fields and applications*, pages 209–218, New York, NY, USA, 1996. Cambridge University Press. 1.2, 3.8

[LM84a] Rudolf Lidl and W. B. Müller. A note on polynomials and functions in algebraic cryptography. *Ars Combin.*, 17(A) :223–229, 1984. I

[LM84b] Rudolf Lidl and Winfried B. Müller. Permutation polynomials in RSA-cryptosystems. In *Advances in cryptology (Santa Barbara, Calif., 1983)*, pages 293–301. Plenum, New York, 1984. I, 3.6

[LM88] Rudolf Lidl and G.L. Mullen. When does a polynomial over a finite field permute the elements of the field ? *Amer. Math. Month.*, 95 :243–246, 1988. I, 2, 3

[LM91] Rudolf Lidl and Gary L. Mullen. Cycle structure of Dickson permutation polynomials. *Mathematical Journal of Okayama University*, 33(1), January 1991. 3.6

[LM93] Rudolf Lidl and G.L. Mullen. When does a polynomial over a finite field permute the elements of the field ? *Amer. Math. Month.*, 100 :71–74, 1993. I, 3.4

[LMT93] Rudolf Lidl, G. L. Mullen, and G. Turnwald. *Dickson polynomials*, volume 65 of *Pitman Monographs and Surveys in Pure and Applied Mathematics*. Longman Scientific & Technical, Harlow, 1993. 3.6

[LN97] Rudolf Lidl and Harald Niederreiter. *Finite fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, second edition, 1997. With a foreword by P. M. Cohn. 1, 1.1.1, 1.2, 1.2, 2.1, 3.2, 4.1, 6

[Luc78] Édouard Lucas. Sur les congruences des nombres eulériens et des coefficients différentiels des fonctions trigonométriques, suivant un module premier. *Bulletin de la Société Mathématique de France*, 6 :49–54, 1878. 2.4.1

[Mat82] Rex Matthews. *Permutation polynomials in one and several variables*. PhD thesis, University of Tasmania, 1982. I, 3.6

[Mat93] Mitsuru Matsui. Linear cryptoanalysis method for des cipher. In *EUROCRYPT*, pages 386–397, 1993. 5.2.1

[McE87] Robert J. McEliece. *Finite field for scientists and engineers*. Kluwer Academic Publishers, Norwell, MA, USA, 1987. 1

[MFI01] Miodrag J. Mihaljevic, Marc P. C. Fossorier, and Hideki Imai. Fast correlation attack algorithm with list decoding and an application. In Mitsuru Matsui, editor, *FSE*, volume 2355 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 2001. 7.3

[MI83] Tsutomu Matsumoto and Hideki Imai. A class of asymmetric cryptosystems based on polynomials over finite rings. In *IEEE International Symposium on Information Theory*, pages 131–132, September 1983. Abstract of Papers. 5.1.2

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynominal-tuples for efficient signature-verification and message-encryption. In Christoph G. Günther, editor, *EUROCRYPT*, volume 330 of *Lecture Notes in Computer Science*, pages 419–453. Springer, 1988. 5.1.2

[MIHM85] Tsutomu Matsumoto, Hideki Imai, Hiroshi Harashima, and Hiroshi Miyakawa. A cryptographically useful theorem on the connection between uni and multivariate polynomials. *Transactions of the IECE of Japan*, 68(3) :139–146, March 1985. 5.1.2

[MN81] Winfried B. Müller and Wilfried Nöbauer. Some remarks on public-key cryptosystems. *Studia Sci. Math. Hungar.*, 16(1-2) :71–76, 1981. I, 3.6, 5.1.1

[MN85] Winfried B. Müller and Rupert Nöbauer. Cryptanalysis of the Dickson scheme. In Franz Pichler, editor, *EUROCRYPT*, volume 219 of *Lecture Notes in Computer Science*, pages 50–61. Springer, 1985. 5.1.1

[MPW06] Ariane M. Masuda, Daniel Panario, and Qiang Wang. The number of permutation binomials over $\mathbb{F}_{4p+1}$ where p and 4p+1 are primes. *Electr. J. Comb.*, 13(1), 2006. 4.1

[MS88] Willi Meier and Othmar Staffelbach. Fast correlation attacks on stream ciphers (extended abstract). In *EUROCRYPT*, pages 301–314, 1988. II, 7.3

[Mul80a] Gary L. Mullen. Local permutation polynomials in three variables over Z_p. *Fibonacci Quart.*, 18(3) :208–214, 1980. I, 5.2.3

[Mul80b] Gary L. Mullen. Local permutation polynomials over Z_p. *Fibonacci Quart.*, 18(2) :104–108, 1980. I, 5.2.3

[Mul89] Gary L. Mullen. Permutation polynomials and nonsingular feedback shift registers over finite fields. *IEEE Trans. Inform. Theory*, 35(4) :900–902, 1989. 5.2.3

[MvzG95] Keju Ma and Joachim von zur Gathen. The computational complexity of recognizing permutation functions. *Comput. Complexity*, 5(1) :76–97, 1995. 2.3, 2.4.2

[MZ07] Ariane M. Masuda and Michael E. Zieve. Nonexistence of permutation binomials of certain shapes. *Electr. J. Comb.*, 14(1), 2007. 4.1

[MZar] Ariane Masuda and Michael Zieve. Permutation binomials over finite fields. *Transactions of the American Mathematical Society*, to appear. to appear. 3.4, 3.5.1, 4.1

[Nöb88] Rupert Nöbauer. Cryptanalysis of a public-key cryptosystem based on Dickson-polynomials. *Math. Slovaca*, 38(4) :309–323, 1988. 3.6, 5.1.1

[NR82] Harald Niederreiter and Karl H. Robinson. Complete mappings of finite fields. *J. Austral. Math. Soc. Ser. A*, 33(2) :197–212, 1982. 3.5.1

[Pat95] Jacques Patarin. Cryptoanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88. In Don Coppersmith, editor, *CRYPTO*, volume 963 of *Lecture Notes in Computer Science*, pages 248–261. Springer, 1995. 5.1.2

[Pat96a] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP) : Two new families of asymmetric algorithms. In *EUROCRYPT*, pages 33–48, 1996. 3.3, 5.1.2

[Pat96b] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP) : Two new families of asymmetric algorithms – extended version. `www.minrank.org/hfe.pdf`, 1996. 3.3, 5.1.2

[Pay71] Stanley E. Payne. A complete determination of translation ovoids in finite Desarguian planes. *Atti Accad. Naz. Lincei Rend. Cl. Sci. Fis. Mat. Natur. (8)*, 51 :328–331, 1971. 3.47

[Pet97] Peter Bürgisser and Michael Clausen and Mohammad Amin Shokrollahi. *Algebraic Complexity Theory*, volume 315 of *Grundlehren der mathematischen Wissenschaften*. Springer Verlag, 1997. 2.4.2

[Pie93] Reinhold Pieper. Cryptanalysis of Rédei- and Dickson permutations on arbitrary finite rings. *Appl. Algebra Engrg. Comm. Comput.*, 4(1) :59–76, 1993. I, 3.6

[PK95] W.T. Penzhorn and G.J. Kühn. Computation of low-weight parity checks for correlation attacks on stream ciphers. In *Cryptography and Coding - 5th IMA Conference*, volume 1025 of *Lecture Notes in Computer Science*, pages 74–83. Springer-Verlag, 1995. 9.2.2

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, 1978. 5.1.1

[SH98] Jörg Schwenk and Klaus Huber. Public key encryption and digital signatures based on permutation polynomials. *Electronic Letters*, 34(8) :759–760, 1998. 5.1

[Shp92] I. E. Shparlinski. A deterministic test for permutation polynomials. *Comput. Complexity*, 2(2) :129–132, 1992. 2

[Sie85] Thomas Siegenthaler. Decrypting a class of stream ciphers using ciphertext only. *IEEE Trans. Computers*, 34(1) :81–85, 1985. II, 7.2

[SL93] Peter J. Smith and Michael J. J. Lennon. LUC : A new public key system. In *IFIP/Sec '93 : Proceedings of the IFIP TC11, Ninth International Conference on Information Security*, pages 103–117, Amsterdam, The Netherlands, The Netherlands, 1993. North-Holland Publishing Co. 5.1.1

[Sma90] Charles Small. Permutation binomials. *Internat. J. Math. Math. Sci.*, 13(2) :337–342, 1990. 3.5.1

[Ste88] Jacques Stern. A method for finding codewords of small weight. In Gérard D. Cohen and Jacques Wolfmann, editors, *Coding Theory and Applications*,

volume 388 of *Lecture Notes in Computer Science*, pages 106–113. Springer, 1988. 9.3.2

[SZP02] Qi Sun, Qi Fan Zhang, and Guo Hua Peng. A new algorithm for the Dickson polynomial $g_e(x, 1)$ public key cryptosystem. *Sichuan Daxue Xuebao*, 39(1) :18–23, 2002. 3.6

[Tur88] Gerhard Turnwald. Permutation polynomials of binomial type. In *Contributions to general algebra, 6*, pages 281–286. Hölder-Pichler-Tempsky, Vienna, 1988. 3.1, 3.4, 3.5.1

[vzG91a] Joachim von zur Gathen. Tests for permutation polynomials. *SIAM J. Comput.*, 20(3) :591–602, 1991. 2, 2.4.1, 2.4.2

[vzG91b] Joachim von zur Gathen. Values of polynomials over finite fields. *Bull. Austral. Math. Soc.*, 43(1) :141–146, 1991. 1.14, 1.2

[Wag02] David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002. 9.3.1

[Wan87] Da Qing Wan. On a conjecture of Carlitz. *J. Austral. Math. Soc. Ser. A*, 43(3) :375–384, 1987. I, 1.2

[Wan94] Da Qing Wan. Permutation binomials over finite fields. *Acta Math. Sinica (N.S.)*, 10(Special Issue) :30–35, 1994. 3.5.1

[Wan02] Luyan Wang. On permutation polynomials. *Finite Fields Appl.*, 8(3) :311–322, 2002. 3.27, 3.5.1

[Wil71] Kenneth S. Williams. Notes on Dickson's permutation polynomials. *Duke Mathematical Journal*, 38(4) :659–665, 1971. 3.6

[WL91] Da Qing Wan and Rudolf Lidl. Permutation polynomials of the form $x^r f(x^{(q-1)/d})$ and their group structure. *Monatsh. Math.*, 112(2) :149–163, 1991. 3.29, 3.30, 3.31, 3.32

[Wol05] Christopher Wolf. *Multivariate Quadratic Polynomials in Public Key Cryptography*. PhD thesis, Katholieke Universiteit Leuven, 2005. 5.1.2

[XLP99] Jin Tao Xiong, Hong Xiu Liu, and De Zhong Pi. The Lucas public-key cryptosystem and its security. *Dianzi Keji Daxue Xuebao*, 28(4) :397–401, 1999. 3.6

[ZC68] Hans Zassenhaus and Sarvadaman Chowla. Some conjectures concerning finite fields. *Norske Vid. Selsk. Forhdl.*, 41 :34–35, 1968. 4.3

[Ziea] Michael Zieve. On a theorem of Carlitz. submitted, `http://arxiv.org/abs/0810.2834`. 1.1.1

[Zieb] Michael Zieve. On some permutation polynomials over $GF(q)$ of the form $x^r h(x^{(q-1)/d})$. submitted, `http://arxiv.org/abs/0707.1110v1`. 3.4, 3.5.1

Table des figures

Liste des algorithmes

Liste des tableaux

Table des matières