Propriétés Différentielles des Permutations et Application en Cryptographie Symétrique

Valentin Suder

THÈSE DE DOCTORAT DE
l'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Valentin SUDER

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Propriétés différentielles des permutations et application en cryptographie symétrique

soutenue le 5 Novembre 2014

devant le jury composé de :

Directrice	
Pascale Charpin	Directrice de recherche Inria
Rapporteurs	
Guang Gong	Professeur à l'Université de Waterloo, Ontario, Canada
Thierry Berger	Professeur à l'Université de Limoges
Examinateurs	
Didier Alquié	Délégation Générale de l'Armement
Christine Bachoc	Professeur à l'Université de Bordeaux
Jean-Claude Bajard	Professeur à l'Université Pierre et Marie Curie, Paris 6
Anne Canteaut	Directrice de recherche Inria
Sihem Mesnager	Maître de conférence à l'Université Paris VIII

Propriétés différentielles des permutations et application en cryptographie symétrique

Valentin Suder

Remerciements

La présence de Pascale Charpin, directrice à toute épreuve de ma thèse, en tête de ces remerciements, n'est pas le seul fruit de la coutume. Ses connaissances, sa patience et l'intérêt qu'elle porte à ses étudiants (mais pas seulement), justifient amplement cette première place. Je te remercie Pascale de m'avoir donné l'opportunité d'apprendre à tes côtés, même si je ne suis pas sûr que ces quelques lignes fassent honneur à ce que tu représentes pour moi, tant humainement que scientifiquement. Je souhaite sincérement à tout étudiant en thèse de t'avoir pour mentor.

Je tiens à remercier les rapporteurs de ma thèse : Thierry Berger et Guang Gong. Thierry, je te remercie pour avoir lu attentivement et t'être intéressé à ce manuscrit, pour m'avoir donné le goût de la recherche en cryptographie lors du Master, et aussi pour m'avoir introduit auprès de Pascale et de l'équipe-projet SECRET. Guang, thank you so much. Even though it must have been a lot of work because of the french writing, you made very helpful and great comments on this document and more generally on my work. I would also thank you for giving me a postdoctoral fellowship in your team, so that I can pursue my research work and explore new topics.

Je remercie sincérement Didier Alquié, Christine Bachoc, Jean-Claude Bajard, Anne Canteaut et Sihem Mesnager, d'accepter de participer à la soutenance en qualité de jury. Je vous remercie pour l'intérêt que vous avez porté à mes travaux ainsi que pour avoir pris de votre précieux temps afin d'assister à cette soutenance.

Mes sentiments se portent aussi tout naturellement vers les personnes avec qui j'ai eu la chance de collaborer : Gohar Kyureghyan, Pascale C., Christina Boura, Marine Minier et María Naya Plasencia.

Thank you Gohar for the wonderful time we spent together at *inria*, discussing about power functions and whatever else ...

Merci à vous, Christina, Marine et María, pour la qualité de votre travail et pour m'avoir fait découvrir le (bon) stress des deadlines nocturnes, mais toujours dans la convivialité. Travailler avec vous ne peut se faire qu'avec un large sourire.

Que seraient ces trois (presque quatre) années passées au sein du projet SECRET sans les personnes qui le composent, y passent, y restent, et que je considère aujourd'hui comme des amis. Je vous remercie pour l'ambiance à la fois chaleureuse, fraternelle et studieuse qui règne au bâtiment 25. Énumérer la liste de toutes les personnes que j'ai pu y croiser est un exercice périlleux que je ne peux malheureusement pas satisfaire exhaustivement. Je remercie tout de même les permanents : André, Anne (merci pour la gestion exemplaire et ta présence amicale quotidienne), Anthony, Christelle (merci pour ton efficacité et ta gentillesse), Gaëtan (merci d'être si bon public tout le temps), Jean-Pierre (merci pour les leçons de baby-foot), María (merci d'être le capital bonne humeur du projet) et Nicolas S.

Je remercie aussi les membres du bureau 1, avec qui j'ai passé le plus de (bon) temps : Ayoub (merci pour ton bureau), Céline B. (merci d'avoir partagé ton savoir), Christina (merci pour le sport, les cocktails, tes histoires incroyables mais surtout pour ton amitié), Gregory (merci tout spécial à ma page de `man` préférée), Marion (merci pour la joie qui t'entoure constamment), Virginie (merci pour tes post-its réguliers) et Adrien (merci d'avoir représenté un peu plus le Master de Limoges).

Merci aussi à tous les anciens du projet, visiteurs impromptus et personnes que j'ai pu rencontrer lors de conférences, séminaires et/ou ... *ailleurs* : Alain, Alexander, Andréa, Antonia, Audrey, Baudoin, Ben, Cécile, Cédric, Daniel, David, Denise, Dimitris, Françoise, Frédéric, Gaël, Irene (merci pour ton soutien aux moments les plus durs de la rédaction et pour le reste), Itai, Jérémy J., Joëlle, John, Julia C., Julia P., Olivier, Mamdouh, Maria C., Mathieu, Matthieu (merci pour ton humilité face à mes assauts de questions stupides), Maxime, Nicky, Rafael (mon brésilien préféré), Rodolphe, Romain P., Sébastien, Stéphane, Thomas P., Thomas R., Vincent H. (merci de m'avoir initié au projet), Yann H., Yann L-C. et tant d'autres ...

Je tiens à remercier les membres de l'ANR BLOC qui ne figurent pas encore dans ces remerciements, à savoir, Brice, Henri, Jean-René, Thomas B., Thomas F. et Yannick. Merci d'avoir fait de ces réunions semestrielles des journées intensément riches. J'ai beaucoup appris avec vous.

Bien des personnes ont déjà été remerciées, mais cette thèse n'aurait pas pu s'accomplir sans les personnes m'ayant initié à la vie parisienne : les Fils de Butte™ ®. Je tiens en premier lieu à vous remercier, vous mes ex-colocataires : Céline T., Jean, ma cousine Lydie (merci d'être toujours présente), mon cousin Simon (merci de me faire rire tout le temps) et Vincent T. Je pense aussi à vous tous qui avaient transformé mes moments de détente en moments inoubliables (des *caviars de moments* en somme...) : Clémence, Émilie, François, Gauthier, Gilles, Jérémy L., Marco, Marie D., Marie Héloïse, Nicolas L., Nour, Ophélie, Pauline et certainement tant d'autres dont je m'en veux d'oublier. Vous avez changé ma vie à jamais durant ces quelques années.

Tous mes amis n'étaient cependant pas à Paris durant cette période. Ils n'en étaient cependant pas moins présents à chaque fois que je le désirais. Je tiens à vous remercier sincérement pour tout : Hélène D., Hélène M., Loïc, Lolita, Marie B., Romain B., Sylvain et Vincent G.

C'est finalement avec une pensée particulièrement émue que je voudrais remercier *l'ensemble* de ma famille, dont mes parents, Marie-Christine et Christian, et mon frère Thomas S., en tête de liste. Même si parfois les mots ont manqué pour exprimer ma gratitude, sachez que c'est un peu à vous que je dédie ce manuscrit.

Cette thèse n'aurait, de toute façon, pas été la même sans la bienveillance et le soutien dont vous avez fait part, tous, à mon égard.

Merci. Merci à vous tous.

Table des matières

Overview

In this document, I present the work that I have carried out between 2011 and 2014 as a PhD student of the SECRET Project-Team at Inria Paris-Rocquencourt. The two main subjects of my thesis are the study of differential criteria of permutations over binary finite fields with applications to symmetric cryptography, and improvements of impossible differential cryptanalysis.

Over the years, cryptography, designated in the beginning for military purposes, started equally to be used for the purposes of the everyday life. This was mainly done to preserve people's privacy in a world where planetary communications became more and more common. Academic scientists formalized new concepts that appeared in *computer science*, and in our case in symmetric cryptography. It is now obvious that we must study carefully the underlying mathematical functions. Indeed, we know that some functions should be preferred with regard to some cryptographic properties. My work lies in this context.

This document is divided in two parts. The first one relies on the study of functions that could be used as nonlinear transformations for building iterated block ciphers. Following Shannon's principles, these functions form the so-called *substitution layer* and are generally called *S-boxes*. Most of the time, S-boxes must be bijective, and in this case it is of great interest to study the inverse of these permutations. We are particularly interested in the differential criterion of functions over binary fields since this criterion measures the resistance against differential cryptanalysis, which is one of the most powerful attacks against block ciphers.

The first chapter reminds basic definitions and properties concerning vectorial Boolean functions, polynomials over binary finite fields as well as iterated block ciphers and some cryptographic criteria. This is the essential background needed for the understanding of the rest of this thesis.

Chapter 2 provides results concerning inverses of bijective power functions, that is inverses of monomial permutations. This work is a joint work with Gohar Kyureghyan. It has been presented at the *International Symposium on Information Theory* ISIT'12 [132], the *11th International conference on finite fields and their applications Fq11* [189] and has been published in *Finite Fields and Their Applications* journal [133] in 2014.

In Chapter 3, I present classes of sparse permutations, that is permutation polynomials with few nonzero coefficients, that have a good resistance against differential cryptanalysis. We study algebraic properties such as the compositional inverse, algebraic degree and differential uniformity. This is a joint work with Pascale Charpin and Gohar Kyureghyan, and it led to a publication in *Finite Fields and Their Applications* journal [69] in 2014.

In Chapter 4, I present a personal ongoing work about higher order differences, antidifferences and a new way to construct quadratic APN functions from 2-to-1 linear functions.

The second part of this manuscript is more dedicated to cryptanalysis of iterated block ciphers. Chapter 5 introduces preliminary and advanced notions of differential cryptanalysis. In Chapter 6, I present a joint work with Christina Boura, Marine Minier and María Naya-Plasencia which will appear in ASIACRYPT 2014 [34, 35, 36]. In this

work, we scrutinized and improved impossible differential cryptanalysis by formalizing the evaluation complexity and by developing new techniques that reduce the overall complexity. We prove the strength of our method by applying it to some Feistel oriented block ciphers. Some of these cryptanalyses are, up to our knowledge, the best known attacks against those ciphers.

First Part: Design

In this part, we investigate the issue of studying and finding bijective functions with "good" differential properties.

Chapter 1: Preliminaries

In this chapter, we explain the interest of investigating and studying functions, that could be used as SBoxes in iterated block ciphers. We introduce notations that are used throughout this document. We equally demonstrate precisely the link between vectorial Boolean functions, that is $\mathbb{F}_2^n \to \mathbb{F}_2^n$ functions, and univariate polynomials on finite fields $\mathbb{F}_{2^n}$.

We then discuss the permutation polynomials, that is polynomials with coefficients lying in a finite field and such that their associated functions are bijective on this finite field. Today, only few infinite classes of permutation polynomials are known, and we explain why it is interesting to find new ones. We also introduce the problem of finding the expression of their inverses for the composition.

Finally, we enumerate the known criteria related to some cryptanalysis. In particular, we explain the relation between the size of the image set of differences of a function and its resistance to differential cryptanalysis, the relation between the distance of its component functions to linear Boolean functions and its resistance to linear cryptanalysis, the relation between its algebraic degree and its resistance to algebraic cryptanalysis, higher-order differential cryptanalysis, etc ... We also describe the known invariant transformations that preserve some of these criteria.

Chapter 2: On Inversion in $\mathbb{Z}_{2^n-1}$

In this chapter, we are interested in finding and computing the inverse of bijective power functions. Bijective binary power functions are functions of the shape

$$\begin{array}{ccc} \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ x & \mapsto & x^d \end{array}$$

where d is a positive integer, called the *exponent*, such that $\gcd(d, 2^n - 1) = 1$ (which ensures the bijectivity on $\mathbb{F}_{2^n}$). We want to find the *least positive residue*, $1 \leq e < 2^n - 1$, of the multiplicative inverse of d modulo $2^n - 1$:

$$e \equiv d^{-1} \pmod{2^n - 1}.$$

In this case, the function $x \mapsto x^e$ is the compositional inverse of the function $x \mapsto x^d$. Indeed, for all $x \in \mathbb{F}_{2^n}$, we have that $(x^d)^e \equiv x \pmod{x^{2^n} + x}$.

In symmetric cryptography, power functions are widely used because of their nice implementation properties in both software and hardware platforms. Moreover, these

functions are quite well understood since they are closely related to the cyclic error correcting codes.

We are particularly interested in the *Almost Perfect Nonlinear* (APN) functions, since they offer the best resistance against differential cryptanalysis. To this day, there exist only few known classes of what we call the *APN exponents*, *i.e.* exponents such that their related power functions are APN. A list of these exponents can be found in Tables 2.1 and 2.2. Even if the classes that are relatively prime to $2^n - 1$, for some n, and their binary representations are known, very few is known about their inverses. It is then of great interest to have more information about these inverses, such as their binary representations and their binary Hamming weight, which correspond to the algebraic degrees of the corresponding functions.

Following this idea, we first study exponents which depend on n to be APN on $\mathbb{F}_{2^n}$. These are the Dobbertin exponents, Niho exponents and Welch exponents. We notably provide the binary representations of these exponents as well as their binary Hamming weight. The multiplicative inverse function case is straightforward since it is an involution function. In particular, we find that the inverses of Dobbertin exponents have a binary Hamming weight equal to $(n+3)/2$, and thus prove in another way that the Dobbertin power functions are not *Almost Bent* (AB) since, due to their high algebraic degree, their inverses are not.

We then study Gold and Kasami exponents, which can be APN on infinitely many extensions $\mathbb{F}_{2^n}$. In order to find their inverses, we face a more general problem. This problem is to find what we can say about the inverses of a fixed exponent, d modulo $2^n - 1$, for all suitable n (*i.e.* $n \in \mathbb{N}_d = \{n \in \mathbb{N} \mid \gcd(d, 2^n - 1) = 1\}$). We thus describe the function

$$\mathrm{Inv}_d : \mathbb{N}_d \to \mathbb{N},$$

whose output for an integer n is the least positive integer describing the inverse of d modulo $2^n - 1$:

$$\mathrm{Inv}_d(n) \times d \equiv 1 \pmod{2^n - 1}.$$

We find that, for a fixed integer d, only a finite number of evaluations of the function Inv_d is needed to completely determine the infinite number of images of Inv_d. This number of evaluations does not exceed $\lfloor \theta_d / 2 \rfloor$, where θ_d is the (multiplicative) order of 2 modulo d, that is

$$\theta_d = \min_o \{o > 0 \mid 2^o \equiv 1 \pmod{d}\}.$$

Indeed, the knowledge of $\mathrm{Inv}_d(r)$ for all $r \in \mathbb{N}_d$ with $0 < r < \theta_d$ is enough to completely determine $\mathrm{Inv}_d(n)$ for all $n \in \mathbb{N}_d$ as can be seen in Theorem 2.3.8. Moreover, there is a one-to-one correspondence between $\mathrm{Inv}_d(r)$ and $\mathrm{Inv}_d(\theta_d - r)$ for all $r < \theta_d$, $r \in \mathbb{N}_d$, as depicted in Proposition 2.3.6 or in Corollary 2.3.7.

Corollary 2.3.7. Let $1 \leq r \leq \theta_d - 1$ be such that $r \in \mathbb{N}_d$. Then

$$\frac{d \cdot \mathrm{Inv}_d(\theta_d - r) - 1}{2^{\theta_d - r} - 1} + \frac{d \cdot \mathrm{Inv}_d(r) - 1}{2^r - 1} = d + 1. \tag{1}$$

Theorem 2.3.8. Let $n \in \mathbb{N}_d$ be a positive integer and let $1 \leq r \leq \theta_d - 1$ be such that $n \equiv r \pmod{\theta_d}$. Then

$$\mathrm{Inv}_d(n) = \mathrm{Inv}_d(r) \cdot \sum_{i=0}^{m} 2^{\theta_d \cdot i} + \left(2^{\theta_d - r} - 1 - \mathrm{Inv}_d(\theta_d - r)\right) 2^r \sum_{i=0}^{m-1} 2^{\theta_d \cdot i}. \tag{2}$$

It is possible to extract other formulas from this last theorem. Some of these formulas are particularly useful for determining the *nice* binary representations of images of Inv_d.

Corollary 2.3.9. Let n be a positive integer with $\gcd(d, 2^n - 1) = 1$ and $1 \leq r < \theta_d$ be an integer such that $r \equiv n \pmod{\theta_d}$. Let

- $\boldsymbol{t_n}$ be the binary sequence of length n representing $\mathrm{Inv}_d(n)$;
- $\boldsymbol{u_{\theta_d}}$ be the binary sequence of length θ_d representing

$$\left(\frac{d \cdot \mathrm{Inv}_d(r) - 1}{2^r - 1} - 1\right) \cdot \frac{2^{\theta_d} - 1}{d} = \mathrm{Inv}_d(r) + 2^r(2^{\theta_d - r} - 1 - \mathrm{Inv}_d(\theta_d - r));$$

- $\boldsymbol{a_r}$ be the binary sequence of length r representing $\mathrm{Inv}_d(r)$;
- $\boldsymbol{\bar{b}_{\theta_d - r}}$ be the complementary sequence of the binary sequence of length $\theta_d - r$ representing the $\mathrm{Inv}_d(\theta_d - r)$.

Then $\boldsymbol{t_n}$ is obtained by concatenating the sequences $\boldsymbol{a_r}$, $\boldsymbol{\bar{b}_{\theta_d - r}}$ and $\boldsymbol{u_{\theta_d}} = \boldsymbol{\bar{b}_{\theta_d - r}} \,|\, \boldsymbol{a_r}$ as follows:

$$\boldsymbol{t_n} = \boldsymbol{a_r} \,|\, \underbrace{\boldsymbol{u_{\theta_d}} \,|\, \boldsymbol{u_{\theta_d}} \,|\, \ldots \,|\, \boldsymbol{u_{\theta_d}}}_{m} = \boldsymbol{a_r} \,|\, \boldsymbol{\bar{b}_{\theta_d - r}} \,|\, \boldsymbol{a_r} \,|\, \ldots \,|\, \boldsymbol{\bar{b}_{\theta_d - r}} \,|\, \boldsymbol{a_r}, \text{ where } m = \frac{n - r}{\theta_d}.$$

We apply these last theorem and corollaries to provide results about quadratic exponents, that is exponents of binary Hamming weight two (Gold exponents are APN quadratic exponents). We provide an efficient recursive algorithm to compute their inverses (see Algorithm 2). We also prove that the algebraic degree of the inverse of a quadratic power function $x \mapsto x^{2^k+1}$ on $\mathbb{F}_{2^n}$ is equal to $(n - \gcd(n, k) + 1)/2$, which is a relatively high algebraic degree.

Concerning the Kasami exponents, that is exponents of the shape $d = 2^{2k} - 2^k + 1$, we establish, in Lemma 2.3.24, a necessary and sufficient condition for such exponents to be invertible modulo $2^n - 1$ (*i.e.* $\gcd(d, 2^n - 1) = 1$). We then give some partial results about their inverses.

Lemma 2.3.24. Let k and n be positive integers. Then $\gcd(2^{2k} - 2^k + 1, 2^n - 1) = 1$ if and only if one of the following conditions is satisfied:

(a) $\frac{n}{\gcd(k,n)}$ is odd, *i.e.* $\gcd(2k, n) = \gcd(k, n)$;

(b) $\frac{n}{\gcd(k,n)}$ and k are even and $\gcd(3k, n) = \gcd(k, n)$.

Finally we study another specific class of exponents that has shown to be very interesting for cryptographic purpose, that is the exponents of the shape $2^k - 1$.

Theorem 2.3.31. Let $n, k \geq 1$ be relatively prime integers and let $0 < k_n^{-1} < n$ be the least positive integer verifying $k \times k_n^{-1} \equiv 1 \pmod{n}$. Then

$$\mathrm{Inv}_{2^k-1}(n) = \sum_{i=0}^{k_n^{-1}-1} 2^{ki} \pmod{n}.$$

Moreover, $wt(\mathrm{Inv}_{2^k-1}(n)) = k_n^{-1}$.

Chapter 3: Sparse Permutations with Low Differential Uniformity

This chapter deals with functions defined as

$$\begin{array}{rccc} F_{s,t,\gamma}: & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^s + \gamma \mathrm{Tr}(x^t). \end{array}$$

This kind of functions is of practical interest since they are composed with only two monomials and so they offer good implementation performances.

We describe the set of such functions which are bijective and give the explicit expressions of their compositional inverses in Theorem 3.2.3 and Theorem 3.2.10 respectively.

Theorem 3.2.3. Let $F_{s,t,\gamma} : x \mapsto x^s + \gamma \mathrm{Tr}(x^t)$ be a function on $\mathbb{F}_{2^n}$ with $\gamma \in \mathbb{F}_{2^n}$. Then $F_{s,t,\gamma}$ is bijective on $\mathbb{F}_{2^n}$ if and only if $\gcd(s, 2^n - 1) = 1$,

$$t \equiv 2^j(2^i + 1)s \pmod{2^n - 1}, \text{ for some integers } 0 \leq i, j \leq n-1, \ i \neq n/2,$$

and either condition (a) or (b) is satisfied:

(a) $i = 0$ and $\mathrm{Tr}(\gamma) = 0$;

(b) $i > 0$ and $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$ with $\mathrm{Tr}(\gamma^{2^i+1}) = 0$.

Theorem 3.2.10. Let $F_{s,t,\gamma}$ be a function on $\mathbb{F}_{2^n}$ satisfying the conditions of Theorem 3.2.3 with $t = s(2^i + 1)$ for some integers i. Let $\sigma = s^{-1} \pmod{2^n - 1}$. Then

$$\begin{aligned} F^{-1}_{s,t,\gamma}(x) &= \left(x + \gamma \mathrm{Tr}(x^{2^i+1})\right)^{\sigma} \\ &= x^{\sigma} + \left(\sum_{j \prec \sigma} x^j \gamma^{\sigma - j} \right) \mathrm{Tr}(x^{2^i+1}), \end{aligned}$$

for all $x \in \mathbb{F}_{2^n}$. Furthermore,

$$\deg(F^{-1}_{s,t,\gamma}) \leq wt(\sigma) + 1.$$

Next, we analyze such functions for some specific positive integers s and t. We determine the size of their image set, their algebraic degree and their differential uniformity. More precisely, we characterize the bijective functions $F_{s,t,\gamma}$ for some values of s and t.

We demonstrate that $F_{s,1,\gamma}$ is bijective on $\mathbb{F}_{2^n}$ only when s is the inverse modulo $2^n - 1$ of a quadratic exponent.

Theorem 3.3.1. A function $F_{s,1,\gamma}$ is bijective if and only if

$$s = \frac{2^j}{2^i + 1} \pmod{2^n - 1} \text{ for some } i, j \text{ with } i > 0,$$

and both of the following conditions hold:

- n/ℓ odd, with $\ell = \gcd(i, n)$;
- $\gamma \in \mathbb{F}_{2^k}$, $k = \gcd(2i, n)$, such that $\mathrm{Tr}(\gamma^{2^i+1}) = 0$.

In this case,

$$\delta(F_{s,1,\gamma}) = 2^{\ell}, \quad \mathcal{NL}(F_{s,\gamma}) = 2^{n-1} - 2^{(n+k-2)/2}, \quad \text{and} \quad \deg(F_{s,1,\gamma}) = \frac{n-\ell+2}{2}.$$

Moreover

$$F^{-1}_{s,1,\gamma} = x^{2^i+1} + \left(\gamma^{2^i+1} + \gamma^{2^i} x + \gamma x^{2^i}\right) \mathrm{Tr}(x^{2^i+1})$$

All component functions of $F_{s,1,\gamma}$ and of its inverse are plateaued.

Note that permutations with differential uniformity 4, when n is even, are then obtained.

We also consider functions constructed from the multiplicative inverse function in a more general way and obtain the following result.

Lemma 3.3.5. Let $F : x \mapsto x^{-1} + \gamma \mathrm{Tr}(H(x))$ be a function on $\mathbb{F}_{2^n}$, where $H : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ and $\gamma \in \mathbb{F}^*_{2^n}$. Then

$$\delta(F) = \begin{cases} \{2, 4\} & \text{when } n \text{ is odd,} \\ \{4, 6\} & \text{when } n \text{ is even.} \end{cases}$$

When n is even, we specifically obtain bijective functions $F_{-1,t,\gamma}$ on $\mathbb{F}_{2^n}$ having a differential uniformity of 6.

Finally, we extensively study functions $F_{s,t,\gamma}$ composed by two quadratic exponents, with a particular insight on their difference functions. Corollary 3.4.12 gives a necessary and sufficient condition for such functions to be APN.

Corollary 3.4.12. Let $F(x) = x^{2^j+1} + \gamma\, \mathrm{Tr}(x^{2^k+1})$ with $\gcd(j, n) = 1$ and $k > 0$. Let $\mu \in \mathbb{F}^*_{2^n}$ such that $\mathrm{Tr}(\mu) = 1$ and define the linear function $L_{j,\mu} : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ by

$$L_{j,\mu}(y) = \sum_{i=0}^{n-1} y^{2^{ij}} \sum_{\ell=0}^{i} \mu^{2^{\ell j}}.$$

Then F is APN if and only if for all $\alpha \in \mathbb{F}^*_{2^n}$ such that $\mathrm{Tr}\left(\gamma/\alpha^{2^j+1}\right) = 0$, we have

$$\mathrm{Tr}\left(\alpha^{2^k+1}\left((L_{j,\mu}(A))^{2^k} + L_{j,\mu}(A)\right)\right) = 0, \text{ where } A = \frac{\gamma}{\alpha^{2^j+1}}.$$

Moreover we end up with some interesting classes of bijective functions:

Theorem 3.4.8. Let $0 < \ell < n$ be integers such that $1 = \gcd(\ell, n)$ and let $\mu \in \mathbb{F}_{2^n}$ be an element such that $\mathrm{Tr}(\mu) = 1$. Then, the function

$$\begin{array}{rccl} F & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^{2^\ell} + x + \mu \mathrm{Tr}(\mu x) \end{array}$$

is bijective on $\mathbb{F}_{2^n}$. Moreover, its compositional inverse is

$$\begin{array}{rccl} F^{-1} & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & L_{\ell,\mu}(x) + \left(1 + L^*_{\ell,\mu}(\mu)\right) \mathrm{Tr}(x). \end{array}$$

where $L^*_{\ell,\mu}$ is the adjoint polynomial of $L_{\ell,\mu}$, that is polynomial satisfying:

$$\mathrm{Tr}(x L_{\ell,\mu}(y)) = \mathrm{Tr}(L^*_{\ell,\mu}(x) y),$$

for all $x, y \in \mathbb{F}_{2^n}$.

As an example, we have:

Proposition 3.4.15. Let n be a positive integer. Then the function

$$\begin{array}{rccl} G & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^3 + \mathrm{Tr}(x^9 + x^3) \end{array}$$

is bijective and has a differential uniformity of 4.

Chapter 4: Antidifferences in $\mathbb{F}_{2^n}$

This chapter is only the introduction to an ongoing study of *difference functions* on $\mathbb{F}_{2^n}$. In the context of this thesis, the term difference functions denotes the difference of a function $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ in a direction $\alpha \in \mathbb{F}_{2^n}^*$:

$$\begin{array}{rccl} \Delta_\alpha F : & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & F(x) + F(x+\alpha). \end{array}$$

We adopt a matrix point of view of functions on $\mathbb{F}_{2^n}$ and use linear algebra in order to have a fresh look on difference functions and their algebraic objects. In other terms, instead of considering a function $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ in its polynomial form, $F(x) = \sum_{i=0}^{2^n-1} f_i x^i \in \mathbb{F}_{2^n}[x]$, we consider the vector of its coefficients:

$$\begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_{2^n-1} \end{pmatrix} \in \mathbb{F}_{2^n}^{2^n}.$$

Obtaining the difference functions of F in a certain direction becomes then the same as evaluating a linear system. We show therefore quite easily an original characterization of difference functions and functions with 0-linear structure:

Corollary 4.3.6. Let $\alpha \in \mathbb{F}_{2^n}$ and $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Then, $\Delta_\alpha G$ is the null function *if and only if* there exists a function $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ such that $\Delta_\alpha F = G$.

With Theorem 4.3.13, we also prove that if the higher order difference of a function, say F, is the null function, then F can be written as a sum of difference functions:

Theorem 4.3.13. Let $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_{2^n}$ be linearly independent over $\mathbb{F}_2$ and let $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Then, the higher-order difference of F in $\alpha_1, \ldots, \alpha_m$ is the null function *if and only if* there exist functions $F_i : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ such that $\Delta_{\alpha_i} F_i = 0$, $1 \le i \le m$, and $F = F_1 + \cdots + F_m$.

These last results in addition to some more properties allow us to define the notion of antidifferences on $\mathbb{F}_{2^n}$. We define also a new equivalence notion of functions, that we call *differential equivalence*. We say then that two functions are in the same class of equivalence, relatively to a set $V \in \mathbb{F}_{2^n}$, if they share the same difference functions in all the directions $\alpha \in V$.

Consecutively, we give a method to obtain antidifference functions from a set of difference functions.

Theorem 4.4.5. Let $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_{2^n}$ be linearly over $\mathbb{F}_2$ independent and let $G_i : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, $1 \le i \le m$, be functions such that $\Delta_{\alpha_i} G_i$, $1 \le i \le m$, is the null function. Then, it exists at least one function $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ such that

$$\Delta_{\alpha_i} F = G_i \quad \forall\ 1 \le i \le m$$

if and only if

$$\Delta_{\alpha_j} G_i = \Delta_{\alpha_i} G_j \quad \forall\ 1 \leq i, j \leq m.$$

Finally, we explain how our results can be applied to combine linear 2-to-1 functions and thus obtain quadratic APN functions.

Second Part: Cryptanalysis

The security evaluation of primitives in symmetric cryptography can not be accomplished in the same way as in asymmetric cryptography. Indeed, the security proofs of symmetric cryptosystems are mostly based on unrealistic models. This security evaluation is thus performed by analyzing independently the resistance of each cipher against as many known attacks as possible. If a cipher has been extensively analyzed by the community without any important weaknesses found, then this cipher can be considered as robust and can be trusted by the community. This is why the role of cryptanalysis is extremely important in symmetric cryptography. As new powerful attacks are discovered by cryptanalysts, new ciphers resisting these new attacks are designed. The responsibility of the attackers is to evaluate as precisely as possible the complexities of the cryptanalysis, even if these are really high. In other terms, the cryptanalysts have to evaluate the amount of time, data and memory needed to recover the secret information (the *key* in the case of ciphers) from the cryptographic function. This is the only way to evaluate the reliability of symmetric functions that are used to offer communication privacy.

To this day, differential cryptanalysis is one of the most efficient forms of cryptanalysis. Differential cryptanalysis has been extensively studied during the last 20 years. Many generalizations and many developments of this attack have appeared during the security evaluation of different block ciphers and hash functions.

In this part, our aim is to provide a complete study of *impossible differential cryptanalysis*, and to present our improvements to these attacks.

Chapter 5: Introduction to Differential Cryptanalysis

This chapter introduces some basic definitions and concepts of the differential cryptanalysis seen as a statistical cryptanalysis. The role of this chapter is to ease the understanding of the technical parts of the cryptanalysis techniques presented in the next chapter.

Chapter 6: Improving Differential Cryptanalysis

In this chapter, we present our contributions to impossible differential cryptanalysis against Feistel-type block ciphers.

Several new designs of block ciphers have been proposed since the rise of the so called *lightweight cryptography*. Lightweight cryptography was born by the need of efficient algorithms for applications used in very constrained environments such as embedded systems, RFID (Radio Frequency IDentification), sensors networks, etc ... However, the security margin of lightweight block ciphers, which are designed to possess very efficient implementations, are often very limited. Since lightweight block ciphers are more and more used in the everyday life, it is really important to evaluate precisely the security that they actually offer.

Impossible differential cryptanalysis exploits mostly the poor diffusion of a block cipher and has been shown to be very efficient for attacking Feistel-type lightweight block ciphers.

Principle. Unlike differential cryptanalysis that uses differential paths of high probability, the aim of impossible differential cryptanalysis is to use differentials that have a probability zero to occur in order to eliminate the key candidates leading to such impossible differentials.

Impossible differential cryptanalysis can be divided in two steps. The first step is to find an impossible differential covering the maximum number of rounds of the considered iterated block cipher. Then one extends this differential by some rounds to both directions. If a candidate key partially encrypts a pair of plaintexts to the impossible differential, then this candidate certainly cannot be the right key and is thus rejected.

Flaws during the key sieving phase. Despite the fact that impossible differential cryptanalysis has been extensively employed, the key sieving step of the attack does not seem yet fully understood. This part of the procedure is highly technical and most of the time, it is hard to verify whether the results presented in the published papers are correct or not.

Some impossible differential cryptanalysis against block ciphers were disproved afterwards by finding flaws in the complexity evaluation or in the computation. A non-exhaustive list of flaws is described in Table 6.1. For instance, some impossible cryptanalysis of the standardized block ciphers CLEFIA and Camellia do not work. Moreover, the complexity evaluation of impossible differential cryptanalysis on the block cipher LBlock lacks optimality. The impossible differential cryptanalysis of the recently designed block cipher Simon also shows errors, invalidating thus these attacks.

Formalization of the complexities evaluation. We first propose an improvement on evaluation of the probability to discard a key from the candidate key set. Usually, cryptanalysts choose this probability such that, at the end, only the right key remains. We show that this probability could be chosen to keep half of the candidate keys in the possible key set. Although we have to verify afterwards each key separately, leading to an increase of the time complexity, we may need less data during the key sieving phase.

We then propose the first generic solution to the following problem:

Problem 6.2.1 . How can we determine the amount of necessary data, say C_N, to obtain N pairs satisfying a given truncated differential?

Finally, after having explained the correct way to extract the attack parameters (differential probabilities, number of key bits involved, ...) from the differential paths, we end up with a nice formula for accurately computing the time complexity of an impossible differential attack.

Automated tool. After having formalized the procedure of the key sieving step and having provided the complexity formulas, we present the software that we developed. This program offers an automated verification of all the impossible differential cryptanalysis against a given Feistel-type block cipher. In other words, this tool eases the tedious task of finding the *best* impossible differential cryptanalysis against a given cipher.

New advanced techniques. Our fine understanding of impossible differential cryptanalysis allows us to develop two new methods that reduce the overall complexities of such attacks.

The first one is based on *multiple impossible differentials*. We formalize them and use them to show that they can be used to reduce the data complexity drastically, that is the amount of data that are needed for the attack to work.

The second one is a totally original one. During the key sieving step, it often happens that the size of the internal state (of the encryption function) that has to be known is smaller than the number of key bits on which it depends. The technique, that we call the *state-test* technique, consists in making a test for some part of the internal state instead of guessing the necessary key bits for computing it. More precisely, the *state-test* technique works by fixing f bits of the plaintexts, something that allows us to reduce the number of key bits that have to be guessed by f. For a partially guessed key, the test we realize consists, under some conditions, in checking if the given part of the internal state can take all the possible values. If so, we deduce that the guessed key cannot be the right one. Since we have to guess less key bits, the time complexity is expected to be smaller.

Applications. Finally, we demonstrate that the precise study of impossible differential cryptanalysis that we carried out can bring new such attacks against several block ciphers.

We notably apply the *state-test* technique and the multiple impossible differentials technique, as well as a combination of both, to improve the the time complexity of the best impossible differential cryptanalysis against CLEFIA-128 by a factor 2^5 and the data complexity by a factor of 2^6. We have also improved the complexity of the previous best known attacks for all the versions of Camellia, and propose the first 14-round impossible differential cryptanalysis beginning by the first round of a slightly modified version of the Camellia-256 cipher.

Our rigorous complexity analysis permits us to extend the best known attack against the LBlock block cipher by one more round. We also provide a large complexity improvement of the previous best known impossible differential cryptanalysis against this cipher.

Finally, using multiple impossible differentials, we present impossible differential cryptanalysis of all the Simon block ciphers' family.

Most of the impossible differential attacks that we present are the best known attack (all cryptanalysis considered) against these ciphers. A summary of our attacks can be found in Table 6.12.

Perspectives

Every work represented in this document offers many openings for further investigation. For instance, by studying the inverse of APN power functions, a challenging task could be to find new exponents that possess good differential properties. Or yet, many more classes of sparse permutations remain to be considered.

Since it is an ongoing work, antidifference functions on $\mathbb{F}_{2^n}$ need more analysis but they could lead to promising methods for building new functions with interesting properties.

Concerning the impossible differential cryptanalysis, a lot of block ciphers remain to be analyzed. As in the work presented in this thesis, only Feistel-type block ciphers have been considered, we could apply our rigorous methods to other constructions, such as Substitution-Permutation Networks (SPN) and check if the known impossible differential cryptanalysis could be improved.

Résumé

Les travaux de recherches exposés dans ce document se situent à l'interface des mathématiques discrètes, corps finis et théorie algébrique des codes, et de la cryptographie symétrique. Ils concernent dans un premier temps l'étude de certaines permutations dans les corps finis binaires, l'expression de leurs inverses et leurs propriétés différentielles. Dans un second temps, ils traitent l'étude approfondie de la cryptanalyse différentielle impossible des chiffrements par blocs itérés.

La cryptographie symétrique est la forme la plus ancienne de cryptographie. Il s'agit de l'ensemble des fonctions et des protocoles permettant des communications entre deux parties partageant le même secret (la clé dans le cas des chiffrements). Le problème d'échanger ce secret s'est alors posé, et à la fin du XX$^{\text{e}}$ siècle sont apparues de nouvelles primitives qui constituent désormais la cryptographie asymétrique et qui répondent notamment à ce problème. Cependant, cette nouvelle forme de cryptographie n'a pas rendu obsolète l'utilisation des fonctions à clés secrètes. En effet, les chiffrements (qu'ils soient par blocs ou par flot) se révèlent très efficaces en terme de capacité de traitement des données.

L'étude des fonctions mathématiques composant les primitives de cryptographie permet d'optimiser la conception et l'analyse de ces dernières. En cryptographie symétrique, les *boîtes-S* (S-boxes en anglais) sont des fonctions de petites tailles souvent utilisées (par exemple dans les chiffrements par blocs, fonctions de hachage) pour assurer la confusion dans les systèmes. Ce sont des fonctions *non-linéaires* qui jouent un rôle important dans la sécurité des fonctions qui les utilisent. Il existe des critères mathématiques permettant de quantifier la résistance des boîtes-S contre des attaques connues, notamment les cryptanalyses linéaires et différentielles.

Dans la première partie de ce document, c'est sur ces boîtes-S que nous portons notre étude. Plus particulièrement, sur la construction de telles fonctions, qui sont bijectives, et qui possèdent une bonne résistance à la cryptanalyse différentielle. Le chapitre 2 traite des inverses (pour la composition de fonction) des fonctions monomiales qui offrent la meilleure résistance à la cryptanalyse différentielle. Ces travaux, réalisés conjointement avec Gohar Kyureghyan ont donné lieu à des présentations aux conférences *IEEE International Symposium on Information Theory* ISIT'12 [132] et *11th International conference on finite fields and their applications Fq11* [189], ainsi qu'à une publication dans le journal *Finite Fields and Their Applications* [133]. Dans le chapitre 3, nous étudions les fonctions du type

$$\begin{array}{ccc} \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ x & \mapsto & x^s + \gamma \mathrm{Tr}(x^t). \end{array}$$

Nous décrivons l'ensemble de ces fonctions qui sont bijectives sur $\mathbb{F}_{2^n}$ et étudions leur inverse ainsi que leur uniformité différentielle. Ces travaux, réalisés avec Pascale Charpin et Gohar Kyureghyan, ont été publiés dans le journal *Finite Fields and Their Applications* [69]. Le chapitre 4 présente un travail personnel en cours de réalisation, et qui concerne les différentielles d'ordre supérieur des fonctions sur $\mathbb{F}_{2^n}$. Nous proposons no-

tamment une méthode pour construire des fonctions à partir de ses différentielles et nous l'utilisons pour construire des fonctions APN quadratiques.

La deuxième partie de ce document est réservée à la cryptanalyse différentielle impossible. Le chapitre 6 traite en profondeur de cette attaque. Dans ce travail, réalisé avec Christina Boura, Marine Minier et María Naya-Plasencia, nous formalisons les concepts inhérents à cette attaque et nous mettons en équation l'évaluation des complexités en temps, données et mémoire. De plus, nous développons de nouvelles méthodes permettant d'optimiser ces attaques. Nous détaillons aussi des algorithmes et un programme permettant de faciliter de manière significative la recherche de la meilleure cryptanalyse différentielle impossible contre un chiffrement par blocs de type Feistel. Nous illustrons ensuite la force de nos résultats sur une multitude de chiffrements par blocs couramment étudiée. Concernant ces chiffrements, nous améliorons les cryptanalyses différentielles impossibles, et dans certains cas, nous donnons les meilleures attaques connues contre ces chiffrements. Une version préliminaire de ces travaux a été déposée sur le site d'archive ouverte *eprint.iacr.org* [34] tandis que la version complète est acceptée pour publication à la conférence ASIACRYPT 2014 [35, 36].

Études des permutations

1. Préliminaires

Le désir de communiquer de manière confidentielle n'est pas récent, mais les moyens *efficaces* qui le permettent, eux, le sont. Jusqu'au XIX^{e} siècle, la cryptographie était réalisée de manière plus ou moins empirique et pour des besoins essentiellement militaires. Avec l'avènement des télécommunications, la cryptographie fut théorisée puis démocratisée. En 1883, August Kerckhoffs énonce des " desiderata de la cryptographie militaire" dont l'un d'eux est devenu un principe qui posa les bases de la cryptographie moderne : « La sécurité d'un cryptosystème ne doit pas reposer sur le fait que l'algorithme n'est pas public ». Plus tard, Claude Shannon résumera ce principe par un simple : « l'adversaire connaît le système ». L'idée générale de cet énoncé est que l'algorithme utilisé pour chiffrer peut éventuellement tomber entre les mains de l'ennemi. Il faut donc que la sécurité repose sur autre chose que la connaissance de cet algorithme, un secret que seules partagent des personnes désirant communiquer entre elles : une clé.

La cryptographie moderne peut être divisée en deux grandes familles : la cryptographie symétrique et la cryptographie asymétrique. La cryptographie symétrique rassemble les primitives où le secret est partagé par les personnes voulant communiquer entre elles. Un des inconvénients est qu'il faut échanger en toute sécurité le secret. C'est dans cette optique qu'est apparue la cryptographie asymétrique : chaque utilisateur possède une paire de clés : une clé publique qu'il peut distribuer à volonté et qui sert à chiffrer les messages qu'on voudrait lui transmettre, et une clé privée qu'il garde pour lui afin d'être le seul à pouvoir déchiffrer les messages qui lui sont destinés. Dans ce document, nous nous intéresserons exclusivement à la cryptographie symétrique.

En cryptographie symétrique, il existe deux grandes classes de chiffrements : les chiffrements par blocs, et les chiffrements à flot. Nous nous intéresserons surtout à la conception de chiffrements par blocs, et plus particulièrement aux fonctions mathématiques qui les composent. Cependant, les mêmes outils mathématiques sont utilisés pour concevoir d'autres primitives symétriques, comme les fonctions de hachage par exemple.

Nous débuterons ce chapitre en posant le problème initial, qui est de concevoir des chiffrements par blocs. Nous en profiterons pour introduire des définitions et des notations qui serviront tout au long de ce document. Nous verrons les manières classiques de concevoir des chiffrements par blocs, et en quoi l'étude des fonctions mathématiques qui les composent est importante.

Nous verrons, dans un second temps, les notions essentielles (en général sans les preuves) concernant les fonctions booléennes vectorielles. Les fonctions booléennes sont aussi des objets mathématiques essentiels dans d'autres domaines tels que les codes correcteurs d'erreurs, les séquences, etc ... Nous verrons donc certaines intersections entre ces domaines. Cela nous permettra parfois de mieux comprendre les liens et les apports des différentes applications. De nombreux auteurs ont écrit des ouvrages didactiques traitant des fonctions booléennes et vectorielles : Claude Carlet [58, 59] et Caroline Fontaine [98, Première Partie] pour ne citer qu'eux. Nous encourageons donc le lecteur à les consulter autant que possible.

Enfin, la dernière section de ce chapitre sera consacrée aux propriétés cryptographiques

qu'il est important de vérifier lors du *design*, de la conception de chiffrements par blocs.

1.1. Les chiffrements par blocs

Définition 1.1.1 (Chiffrement par blocs)**.** Un chiffrement par blocs est une famille, notée E, de fonctions bijectives, de l'espace des mots binaires de longueur m, paramétrée par une *clé* $K \in \mathbb{F}_2^\kappa$. Il prend en entrée un *bloc* binaire $P \in \mathbb{F}_2^m$, appelé *clair*, et son image est un *bloc* binaire $C \in \mathbb{F}_2^m$, appelé *chiffré*. Plus formellement, un chiffrement par blocs est donné par la fonction

$$\begin{array}{rccl} E: & \mathbb{F}_2^\kappa \times \mathbb{F}_2^m & \to & \mathbb{F}_2^m \\ & (K, P) & \mapsto & C = E(K, P), \end{array}$$

telle que E est bijective sur $\mathbb{F}_2^m$ pour toute valeur fixée de la clé $K \in \mathbb{F}_2^\kappa$. Pour une clé fixée K, nous noterons $E_K : P \mapsto C$ la fonction de chiffrement.

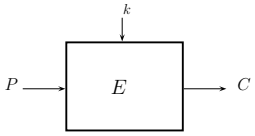


FIGURE 1.1. : Schéma d'un chiffrement par blocs.

Le fait que la fonction de chiffrement, pour une clé fixée, soit une permutation de $\mathbb{F}_2^m$, est naturellement un pré-requis pour pouvoir déchiffrer.

Remarque 1.1.2. Le terme *bloc* est employé, car dans la pratique, les messages que l'on veut chiffrer sont de taille arbitraire. Nous les découpons donc en *blocs* de taille fixe et appliquons le chiffrement par blocs E à plusieurs reprises pour obtenir le message chiffré final. La manière dont on enchaîne les applications de E est définie par un *mode opératoire*.

Concevoir un chiffrement par blocs, c'est donc concevoir une famille de 2^κ permutations de l'espace $\mathbb{F}_2^m$. En raison des contraintes pratiques, et de sécurité, ce n'est pas forcément chose aisée, comme nous allons le voir dans la sous-section suivante. Une fois ce problème posé, nous verrons les deux constructions classiques, actuelles, de chiffrements par blocs : les schémas de Feistel, et les réseaux de substitution-permutation.

1.1.1. Problème de conception

Puisque le chiffrement par blocs E est supposé connu, la sécurité d'une fonction de chiffrement E_K repose sur la seule partie secrète : la clé. Lors de la conception d'un chiffrement par blocs, nous devons donc faire en sorte que la dimension de l'espace des clés (*i.e.* la taille de la famille de permutations) soit suffisamment grande. En effet, nous ne voudrions pas qu'un attaquant possédant un couple de blocs clair/chiffré puisse tester toutes les clés (*i.e.* toutes les permutations) possibles afin de retrouver celle qui a servi au chiffrement. Cependant, pour des raisons pratiques d'implémentation, cette dimension doit aussi être raisonnable. En général, les clés qui serviront dans les chiffrements par blocs seront de taille ≥ 80 bits.

Par exemple l'AES [79] (pour "Advanced Encryption Standard"), qui est le chiffrement par blocs standardisé par le NIST, admet des clés de 128, 192 ou 256 bits.

Pour les mêmes raisons pratiques, la taille des blocs à chiffrer/déchiffrer, doit rester raisonnable. En règle générale, cette taille est ≥ 64 bits et préférentiellement un multiple de 8. Par exemple, la taille des blocs que peut chiffrer l'AES est de 128 bits.

En d'autres termes, concevoir un chiffrement par blocs, c'est sélectionner certaines permutations parmi toutes les permutations d'un espace donné.

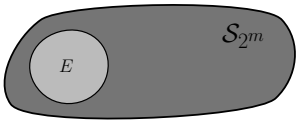


FIGURE 1.2. : Ensemble des permutations du chiffrement par blocs E parmi l'ensemble des permutations de 2^m éléments, $\mathcal{S}_{2^m}$.

De plus, pour des raisons de sécurité, chaque permutation doit être *indistinguable* d'une permutation aléatoire. Autrement dit, un attaquant ne doit pas pouvoir déduire à partir de couples $(x, \mathcal{P}(x))$ si $\mathcal{P}$ est une permutation aléatoire, ou si $\mathcal{P} = E_K$ pour $K \in \mathbb{F}_2^\kappa$.

Le problème est donc le suivant :

Problème 1.1.3. Comment choisir et décrire 2^κ permutations de l'ensemble $\mathcal{S}_{2^m}$ qui soient indistinguables de permutations aléatoires ?

Par exemple, si l'on considère $\kappa = 80$ et $m = 64$, concevoir un chiffrement par blocs c'est choisir une des $\binom{2^{64}!}{2^{80}}$ familles de permutations.

Le moyen le plus efficace à ce jour de répondre à ce problème est de considérer des permutations paramétrées plus simples, appelées *tours du chiffrement* et notées F_i, que nous composerons entre elles (voir figure 1.3). Pour des raisons de taille de circuit, les permutations F_i sont en général très similaires (pour un algorithme de chiffrement donné), ne différant les unes des autres typiquement que d'une constante. Pour un chiffrement donnée, le nombre de tours à effectuer dépend de la sécurité voulue, avec toujours cette contrainte d'implémentation. Nous appelons ces chiffrements par blocs, des *chiffrements itératifs* par blocs.

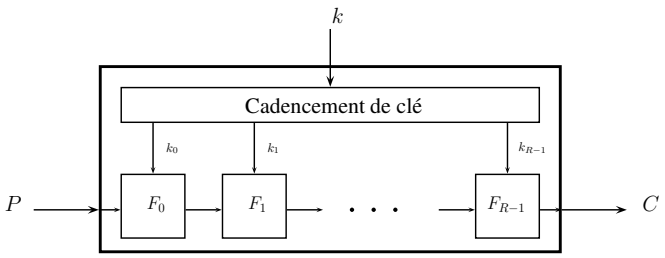


FIGURE 1.3. : Principe d'un chiffrement itératif par blocs.

Remarque 1.1.4. Cette construction nécessite un algorithme, appelé *cadencement de clé*, permettant d'étendre la clé K en le nombre désiré de *sous-clés* RK_i.

Il existe principalement deux manières de concevoir des chiffrements itératifs par blocs que nous détaillons dans les deux sous-sections suivantes. Ces deux structures

suivent généralement les deux principes fondamentaux de conception énoncés par Claude Shannon [183] : *confusion* et *diffusion*.

La confusion permet de rendre la relation entre le clair, la clé et le chiffré la plus complexe possible.

La diffusion permet de dissiper les données statistiques du clair dans le chiffré.

La confusion est en général assurée par des fonctions non-linéaires. Toutefois, il est compliqué de mettre en oeuvre de telles fonctions qui s'appliqueraient sur un bloc entier. C'est pour cela qu'il est courant de diviser un bloc en de plus petits mots, et de leur appliquer une plus petite fonction non-linéaire à chacun, appelées *boîtes-S* ("Sboxes" en anglais). Typiquement, les boîtes-S sont de taille 4 ou 8 bits, plus rarement un nombre impair de bits. Dans l'AES par exemple, la boîte-S est une fonction de $\mathbb{F}_2^8$ vers $\mathbb{F}_2^8$. La diffusion, quant à elle, est le plus souvent assurée par une fonction linéaire.

C'est dans ce contexte que se situe la première partie de ce document : étudier les spécificités des fonctions qui peuvent prendre le rôle de boîtes-S dans les chiffrements par blocs.

1.1.2. Schéma de Feistel

Dans un schéma de Feistel[1], les bits d'entrée sont séparés en deux parties de tailles égales, mais ne subissent pas le même traitement. Un tour d'un schéma de Feistel est décrit à la Figure 1.4 et est donné par :

$$\begin{array}{rcl} \mathbb{F}_2^m \times \mathbb{F}_2^m & \to & \mathbb{F}_2^m \\ (L_i, R_i) & \mapsto & (L_{i+1}, R_{i+1}) = (R_i \oplus F(L_i, RK_i), L_i). \end{array}$$

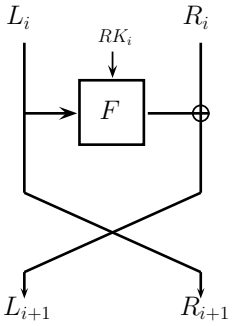


FIGURE 1.4. : Un tour d'un schéma de Feistel.

De par sa conception, un schéma de Feistel est forcément bijectif pour une clé donnée RK_i. À transposition près des deux moitiés, la fonction de tour est involutive :

$$(R_{i+1}, L_{i+1}) \mapsto (L_{i+1} \oplus F(R_{i+1}, RK_i), R_{i+1}) = (R_i \oplus F(L_i, RK_i) \oplus F(L_i, RK_i), L_i) = (R_i, L_i).$$

Par conséquent, il suffit d'inverser l'ordre des sous-clés et d'appliquer le même processus de chiffrement au chiffré pour retrouver le clair.

L'attention du concepteur quant à la sécurité de son chiffrement se porte donc sur la conception de la fonction interne F, qui assure la confusion du chiffrement. Elle peut

[1] du nom du cryptologue Horst Feistel.

elle-même être décomposée en de plus petites parties, auxquelles on applique des boîtes-S. La diffusion est assurée par le xor et l'échange des deux moitiés de l'état, mais peut être en plus présente dans la fonction interne F.

Le DES (pour "Data Encryption Standard"), était le standard de chiffrement par blocs utilisé à partir de 1977. C'est un schéma de Feistel.

En 1996, Kaisa Nyberg [168] généralisa les schémas de Feistel, leur autorisant plus de branches et de branchements. De nombreux chiffrements par blocs actuels utilisent des schémas de Feistel généralisés, notamment CLEFIA [184] et Camellia [7] qui ont tout deux été standardisés par l'ISO/IEC en 2007 et 2000 respectivement.

1.1.3. Réseau de substitution-permutation

Un réseau de substitution-permutation, ou SPN (pour "Substitution Permutation Network") est composé de l'itération de tours qui sont la composition de trois fonctions distinctes (voir figure 1.5).

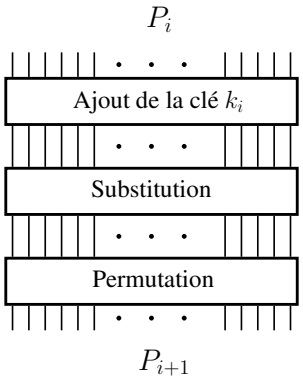


FIGURE 1.5. : Un tour d'un SPN.

Ces trois fonctions, qui doivent être bijectives pour pouvoir déchiffrer, sont :

1. La phase d'ajout de clé permet d'insérer le secret à chaque tour. Il s'agit en général du xor bit à bit de la sous-clé RK_i avec le bloc P_i. On parle dans ce cas là de "key-alternating cipher"[2], sur lesquels il existe des preuves de sécurité.

2. La phase de substitution constitue la partie de confusion du chiffrement. Pour des raisons d'implémentation, elle est en général composée de boîtes-S, différentes ou non, positionnées en parallèle. Cette fonction de substitution constitue la seule partie non-linéaire du chiffrement.

3. La phase de permutation permet quant à elle de diffuser et de mélanger linéairement les sorties des boîtes-S de la fonction de substitution. Elle peut être soit une permutation de bits ou de mots binaires plus longs, ou bien une combinaison linéaire de ces bits/mots.

L'AES est un réseau de substitution-permutation dont la représentation est particulière. En effet, son état peut se représenter sous forme d'une matrice carrée et il admet deux couches de diffusion linéaire : une qui échange les positions des mots de 8 bits sur les

[2]une traduction possible serait : *chiffrement alternant clés et permutations.*

lignes, et une qui combine linéairement des mots de 8 bits quatre par quatre sur les colonnes.

1.2. Les fonctions booléennes

Une fonction booléenne à n variables est une fonction projective de l'espace vectoriel $\mathbb{F}_2^n$ vers $\mathbb{F}_2$.

Définition 1.2.1. Une fonction booléenne f à n variables est de la forme

$$\begin{array}{rccc} f: & \mathbb{F}_2^n & \to & \mathbb{F}_2 \\ & (x_0, x_1, \ldots, x_{n-1}) & \mapsto & f(x_0, x_1, \ldots, x_{n-1}). \end{array}$$

Nous notons $\mathcal{B}_n$ l'ensemble des fonctions booléennes à n variables.

L'étude des fonctions booléennes est en étroite relation avec la théorie algébrique des codes, notamment des codes de Reed-Muller. Nous renvoyons le lecteur vers l'ouvrage incontournable en théorie des codes correcteurs d'erreurs de Florence MacWilliams et Neil Sloane [150], ou bien les travaux de Tadao Kasami sur les codes de Reed-Muller [121, 120], pour de plus amples informations à ce sujet.

Une fonction booléenne peut être représentée de plusieurs façons. Dans cette introduction, nous nous intéresserons à la représentation d'une fonction booléenne par son vecteur de valeurs, sa forme algébrique normale, et sa forme trace. Ce ne sont pas les seules façons de représenter une fonction booléenne (voir [58]). Précisons que ces deux premières représentations sont uniques pour une fonction booléenne donnée.

Définition 1.2.2 (Vecteur de valeurs)**.** Le *vecteur de valeurs* d'une fonction booléenne f à n variables est un vecteur dont les composantes correspondent aux 2^n valeurs $f(x)$, $x \in \mathbb{F}_2^n$. On parle aussi de *table de valeurs*, quelquefois de *table de vérité* par analogie avec le vocabulaire de la logique booléenne.

Notons que l'unicité du vecteur de valeurs dépend de *l'ordre* dans lequel nous évaluons la fonction f sur les 2^n éléments de $\mathbb{F}_2^n$.

D'après cette définition, l'ensemble des fonctions booléennes peut être caractérisé ainsi :

$$\mathcal{B}_n = \mathbb{F}_2^{2^n}. \tag{1.1}$$

Au vu du lien évident que partage cette représentation avec les mots de codes, nous pouvons utiliser le vocabulaire de la théorie des codes. Le support d'un mot binaire à n coordonnées $x = (x_0, \ldots, x_{n-1}) \in \mathbb{F}_2^n$, que l'on notera $supp(x)$, est l'ensemble de ses coordonnées non nulles. Le cardinal du support, que l'on notera $wt(x)$, correspond au poids de Hamming binaire de x. On peut également définir la distance de Hamming séparant deux mots binaires x et y par $wt(x + y)$. De plus, on dira que x *recouvre* (resp. strictement) y, et que l'on notera $x \succeq y$ (resp. $x \succ y$), si $supp(x) \supseteq supp(y)$ (resp. $supp(x) \supset supp(y)$).

Rappelons qu'un entier positif, $d < 2^n$, admet une unique forme binaire $d = \sum_{i=0}^{n-1} d_i 2^i$ avec $d_i \in \mathbb{F}_2$. Nous pouvons donc associer l'entier d avec le vecteur $(d_0, \ldots, d_{n-1}) \in \mathbb{F}_2^n$. Par la suite, nous utiliserons les notations ci-dessus indifféremment pour des vecteurs binaires, ou pour des entiers positifs.

Exemple 1.2.3. Soit l'entier $13 = 1 + 2^2 + 2^3$. Nous pouvons donc écrire 13 comme le mot binaire $(1, 0, 1, 1)$ sur $\mathbb{F}_2^4$. Ainsi $supp(13) = \{0, 2, 3\}$ et $wt(13) = 3$. De plus, nous avons que $13 \succ 5$ car $supp(5) = \{0, 2\} \subset supp(13)$.

Définition 1.2.4. Le support d'une fonction booléenne f à n variables, noté $supp(f)$, est l'ensemble des éléments $x \in \mathbb{F}_2^n$ tels que $f(x) = 1$. Le poids de f, noté $wt(f)$, est le cardinal de son support. Nous notons la distance qui sépare deux fonctions booléennes f et g par $wt(f + g)$.

Exemple 1.2.5 (Table de vérité)**.** Soit f une fonction booléenne à 3 variables.

x_0	0	1	0	1	0	1	0	1
x_1	0	0	1	1	0	0	1	1
x_2	0	0	0	0	1	1	1	1
$f(x_0, x_1, x_2)$	0	1	1	1	0	0	0	0

Le poids de la fonction f est : $wt(f) = 3$.

Définition 1.2.6. Une fonction booléenne f à n variables est dite équilibrée si son vecteur de valeurs possède autant de fois la valeur 1 que la valeur 0, *i.e.* si $wt(f) = 2^{n-1}$.

Puisque nous travaillons sur l'espace vectoriel $\mathbb{F}_2^n$, nous pouvons définir le produit scalaire usuel.

Définition 1.2.7. Soient $x = (x_0, \ldots, x_{n-1})$ et $y = (y_0, \ldots, y_{n-1})$ des éléments de $\mathbb{F}_2^n$. Le produit scalaire sur $\mathbb{F}_2^n$ est la fonction $\mathbb{F}_2^n \to \mathbb{F}_2$ donnée par

$$x \cdot y = \bigoplus_{i=0}^{n-1} x_i y_i.$$

Définition 1.2.8 (Forme algébrique normale)**.** La *forme algébrique normale* (ANF, pour 'Algebraic Normal Form' en anglais) d'une fonction booléenne f à n variables est l'unique polynôme multivarié de l'anneau $\mathbb{F}_2[x_0, \ldots, x_{n-1}]/(x_0^2 + x_0, \ldots, x_{n-1}^2 + x_{n-1})$ tel que

$$f(x_0, \ldots, x_{n-1}) = \bigoplus_{(u_0,\ldots,u_{n-1}) \in \mathbb{F}_2^n} c(u_0, \ldots, u_{n-1}) x_0^{u_0} \ldots x_{n-1}^{u_{n-1}},$$

où $c(u_0, \ldots, u_{n-1}) \in \mathbb{F}_2$. Pour des soucis de concision, nous écrirons

$$f(x) = \bigoplus_{u \in \mathbb{F}_2^n} c(u) x^u,$$

où $x = (x_0, \ldots, x_{n-1})$, $u = (u_0, \ldots, u_{n-1})$ et $x^u = x_0^{u_0} \ldots x_{n-1}^{u_{n-1}}$.

Propriété 1.2.9 (Transformée de Moebius)**.** *Soit f une fonction booléenne à n variables dont l'ANF est*

$$f(x) = \bigoplus_{u \in \mathbb{F}_2^n} c(u) x^u,$$

où $c(u_0, \ldots, u_{n-1}) \in \mathbb{F}_2$. Alors,

$$f(x) = \bigoplus_{u \preceq x} c(u) \qquad et \qquad c(u) = \bigoplus_{x \preceq u} f(x).$$

Nous verrons la représentation sous forme de trace dans le paragraphe traitant des fonctions vectorielles. Nous introduisons maintenant le degré algébrique d'une fonction booléenne.

Définition 1.2.10 (Degré algébrique)**.** Le degré algébrique d'une fonction booléenne f à n variables correspond au degré multivarié de son ANF $f(x) = \bigoplus c(u)x^u$:

$$\deg(f) = \max_{u \in \mathbb{F}_2^n} \{wt(u) \mid c(u) \neq 0\}.$$

Dans certains ouvrages, le degré algébrique est parfois appelé *ordre non-linéaire*.

Proposition 1.2.11. *Soit f une fonction booléenne à n variables. Alors, $wt(f)$ est impair si et seulement si* $\deg(f) = n$.

Les fonctions booléennes de degré algébrique 1 sont les fonctions booléennes affines. Elles sont de la forme $\phi_u : x \mapsto u \cdot x + c$, avec $u \in \mathbb{F}_2^n$ et $c \in \mathbb{F}_2$, et vérifient bien la propriété de linéarité lorsque $c = 0$

$$\forall x, y \in \mathbb{F}_2^n, \qquad \phi_u(x \oplus y) = u \cdot (x \oplus y) = u \cdot x \oplus u \cdot y = \phi_u(x) \oplus \phi_u(y).$$

1.3. Fonctions booléennes vectorielles

Définition 1.3.1. Une fonction booléenne vectorielle à n variables et m coordonnées est un produit cartésien de m fonctions booléennes :

$$\begin{array}{rccc} F : & \mathbb{F}_2^n & \to & \mathbb{F}_2^m \\ & (x_0, \ldots, x_{n-1}) & \mapsto & (f_0(x_0, \ldots, x_{n-1}), \ldots, f_{m-1}(x_0, \ldots, x_{\ -1})), \end{array}$$

où les $f_0, \ldots, f_{m-1}$ sont des fonctions booléennes appelées les *fonctions coordonnées*.

Évidemment, les fonctions booléennes sont un cas particulier ($m = 1$) des fonctions booléennes vectorielles.

L'ensemble des combinaisons linéaires de ces fonctions coordonnées dans $\mathbb{F}_2^m$ sont appelées les *fonctions composantes*. De manière plus formelle, la définition des fonctions composantes est la suivante :

Définition 1.3.2 (Fonctions composantes)**.** Soit une fonction booléenne vectorielle F à n variables et m coordonnées. Les *fonctions composantes* de F sont les fonctions

$$\begin{array}{rccc} \tilde{f}_\lambda : & \mathbb{F}_2^n & \to & \mathbb{F}_2 \\ & x & \mapsto & \lambda \cdot F(x), \end{array}$$

pour tout $\lambda \in \mathbb{F}_2^m$.

Remarque 1.3.3. Lorsque $\lambda = 0$, la fonction booléenne composante $\tilde{f}_0$ correspond à la fonction nulle.

De la même façon que nous l'avons fait pour les fonctions booléennes, nous pouvons définir un *degré algébrique* sur les fonction booléennes vectorielles.

Définition 1.3.4 (Degré algébrique d'une fonction booléenne vectorielle)**.** Le degré algébrique d'une fonction booléenne vectorielle F à n variables et m coordonnées est égal au maximum des degrés algébriques de ces coordonnées :

$$\deg(F) = \max_{0 \leq i < m} \deg(f_i),$$

où les fonctions booléennes f_i sont les fonctions coordonnées de la fonction booléenne vectorielle F.

Comme à la section précédente, les fonctions booléennes vectorielles affines sont les fonctions booléennes vectorielles de degré algébrique 1. Les fonctions coordonnées de telles fonctions booléennes vectorielles sont toutes des fonctions booléennes affines ou constantes.

1.3.1. Relation entre l'espace vectoriel $\mathbb{F}_2^n$ et le corps fini $\mathbb{F}_{2^n}$

La théorie des corps finis étant vaste et très riche, nous renvoyons le lecteur désireux d'approfondir ses connaissances vers des auteurs ayant écrit des ouvrages considérés comme "de référence" : Rudolf Lidl et Harald Niederreiter [146], Robert J. McEliece [156], Zhe-Xian Wan [205], ou bien encore Gary L. Mullen et Daniel Panario [161]. En voici quand même une définition.

Définition 1.3.5 (Corps fini)**.** Un *corps fini* est un anneau fini sans diviseur de zéro.

Dans cette sous-section, nous n'aborderons que les aspects de cette théorie essentiels à la compréhension du reste de cette thèse. Dans la majeure partie de ce document, nous préférerons travailler dans les corps finis de caractéristique paire plutôt que dans les espaces vectoriels assimilés. En effet, les corps finis offrent une structure algébrique plus riche. Nous allons aussi expliciter les liens importants unissant l'espace vectoriel $\mathbb{F}_2^n$ et le corps fini $\mathbb{F}_{2^n}$.

Définition 1.3.6 (Trace)**.** Soient des entiers positifs n et m tels que $m \mid n$, c'est à dire que $\mathbb{F}_{2^m}$ est un sous-corps de $\mathbb{F}_{2^n}$. La trace de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_{2^m}$, notée $\mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}$ est une fonction définie par

$$\begin{array}{rccl} \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}} : & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^m} \\ & x & \mapsto & x + x^{2^m} + x^{2^{2m}} + \cdots + x^{2^{(n/m-1)m}}. \end{array}$$

La fonction trace est une fonction $\mathbb{F}_{2^m}$-linéaire. En effet, pour tout $x, y \in \mathbb{F}_{2^n}$ et tout $\mu \in \mathbb{F}_{2^m}$, nous avons $\mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\mu x + y) = \mu \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(x) + \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(y)$. De plus, nous savons que pour tout $\alpha, \beta \in \mathbb{F}_{2^n}$ nous avons $(\alpha+\beta)^2 = \alpha^2 + \beta^2$ et $\alpha^{2^n} = \alpha$ (*i.e.* $\alpha^{2^n-1} = 1$ si $\alpha \neq 0$). En conséquence, la fonction trace vérifie les propriétés

$$\left(\mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha)\right)^{2^m} = \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha) \quad \text{et} \quad \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha^{2^m}) = \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha). \tag{1.2}$$

Lorsque $m = 1$, la fonction trace est appelée *trace absolue* sur $\mathbb{F}_{2^n}$. Nous la noterons simplement Tr pour alléger les notations.

Définition 1.3.7 (Base)**.** Une *base* du corps $\mathbb{F}_{2^n}$ sur un sous-corps $\mathbb{F}_{2^m}$ est un sous-ensemble

$$\left\{\alpha_0, \ldots, \alpha_{n/m-1}\right\} \subset \mathbb{F}_{2^n}^*$$

tel que

$$\forall x \in \mathbb{F}_{2^n},\ \exists!\ (c_0, \ldots, c_{n/m-1}) \in \mathbb{F}_{2^m}^{n/m} \quad \text{vérifiant} \quad x = \sum_{i=0}^{n/m-1} c_i \alpha_i.$$

Une base d'un corps fini $\mathbb{F}_{2^n}$ sur un sous-corps $\mathbb{F}_{2^m}$ est donc un sous-ensemble de $\mathbb{F}_{2^n}$ tel que tout élément de $\mathbb{F}_{2^n}$ s'écrive comme une unique combinaison linéaire des éléments de la base sur $\mathbb{F}_{2^m}$.

Nous remarquons qu'une extension de corps $\mathbb{F}_{2^n}/\mathbb{F}_{2^m}$ ne possède pas qu'une unique base. De plus, à chaque base correspond une base dite *duale*.

Définition 1.3.8 (Base duale)**.** Soient $\{\beta_0, \ldots, \beta_{n/m-1}\}$ et $\{\alpha_0, \ldots, \alpha_{n/m-1}\}$ deux bases de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_{2^m}$. Ces deux bases sont dites *duales* si pour tout $0 \leq i, j < n/m$ nous avons

$$\mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\beta_i \alpha_j) = \delta_{ij},$$

où δ_{ij} est le symbole de Kronecker[3].

Propriété 1.3.9. *Pour chaque base de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_{2^m}$, il existe une unique base duale.*

Lorsque $m = 1$, nous pouvons écrire chaque élément de $\mathbb{F}_{2^n}$ comme un vecteur binaire de $\mathbb{F}_2^n$ relatif à une certaine base de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$. Pour une certaine base $\mathcal{B} = \{\beta_0, \ldots, \beta_{n-1}\}$ de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$, nous avons donc l'isomorphisme, noté $\varphi_{\mathcal{B}}$, reliant $\mathbb{F}_2^n$ à $\mathbb{F}_{2^n}$.

$$\begin{array}{rccl} \varphi_{\mathcal{B}} : & \mathbb{F}_2^n & \to & \mathbb{F}_{2^n} \\ & (x_0, \ldots, x_{n-1}) & \mapsto & x_0\beta_0 + \cdots + x_{n-1}\beta_{n-1}. \end{array} \tag{1.3}$$

La notion de produit scalaire est définie par la fonction trace dans le corps fini $\mathbb{F}_{2^n}$:

$$\forall x, y \in \mathbb{F}_2^n, \qquad x \cdot y = \mathrm{Tr}\left(\varphi_{\mathcal{B}}(x)\varphi_{\mathcal{A}}(y)\right),$$

où $\mathcal{A}$ est la base duale de la base $\mathcal{B}$. En effet, posons $\mathcal{B} = \{\beta_0, \ldots, \beta_{n-1}\}$ et $\mathcal{A} = \{\alpha_0, \ldots, \alpha_{n-1}\}$, et soient $x = (x_0, \ldots, x_{n-1})$ et $y = (y_0, \ldots, y_{n-1})$, nous avons alors

$$\begin{aligned} \mathrm{Tr}(\varphi_{\mathcal{B}}(x)\varphi_{\mathcal{A}}(y)) &= \mathrm{Tr}\left(\left(\sum_{i=0}^{n-1} x_i\beta_i\right)\left(\sum_{i=0}^{n-1} y_i\alpha_i\right)\right) \\ &= \mathrm{Tr}\left(x_0\beta_0\left(\sum_{i=0}^{n-1} y_i\alpha_i\right) + \cdots + x_{n-1}\beta_{n-1}\left(\sum_{i=0}^{n-1} y_i\alpha_i\right)\right) \\ &= \sum_{i=0}^{n-1}(x_0 y_i \mathrm{Tr}(\beta_0\alpha_i)) + \cdots + \sum_{i=0}^{n-1}(x_{n-1} y_i \mathrm{Tr}(\beta_{n-1}\alpha_i)) \\ &= x_0y_0 + x_1y_1 + \cdots + x_{n-1}y_{n-1}, \end{aligned}$$

puisque les bases $\mathcal{B}$ et $\mathcal{A}$ sont duales.

Définition 1.3.10 (Hyperplan)**.** Soit un entier positif n. Un *hyperplan* relatif à un élément $\gamma \in \mathbb{F}_{2^n}$, noté H_γ, est donné par :

$$H_\gamma = \{x \in \mathbb{F}_{2^n} \mid \mathrm{Tr}(\gamma x) = 0\} = \left\{x \in \mathbb{F}_{2^n} \mid \varphi_{\mathcal{B}}^{-1}(\gamma) \cdot \varphi_{\mathcal{A}}^{-1}(x) = 0\right\},$$

où $\mathcal{B}$ et $\mathcal{A}$ sont n'importe quelles bases duales de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$.
Le *complémentaire* de l'hyperplan H_γ, noté $\overline{H_\gamma}$ est défini comme

$$\overline{H_\gamma} = \{x \in \mathbb{F}_{2^n} \mid \mathrm{Tr}(\gamma x) = 1\} = \left\{x \in \mathbb{F}_{2^n} \mid \varphi_{\mathcal{B}}^{-1}(\gamma) \cdot \varphi_{\mathcal{A}}^{-1}(x) = 1\right\}.$$

[3] $\delta_{ij} = 1$ si $i = j$ et $\delta_{ij} = 0$ sinon.

1.3.2. Représentation polynomiale des fonctions sur $\mathbb{F}_{2^n}$

À la section précédente, nous avons vu qu'il était possible d'identifier l'espace vectoriel $\mathbb{F}_2^n$ avec le corps fini $\mathbb{F}_{2^n}$. Le théorème suivant est une adaptation de l'*interpolation de Lagrange* dans les corps finis. Dans cette section, nous expliquons quelle est la relation entre les fonctions $\mathbb{F}_2^n \to \mathbb{F}_2^n$ et les fonctions $\mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$.

Théorème 1.3.11 (Interpolation de Lagrange)**.** *Soit une fonction $f : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Alors, il existe un unique polynôme $P_f(x) \in \mathbb{F}_{2^n}[x]$ de degré (polynomial) au plus $2^n - 1$ représentant la fonction f, c'est-à-dire $P_f(\beta) = f(\beta)$ pour tout élément $\beta \in \mathbb{F}_{2^n}$. Ce polynôme est égal à*

$$P_f(x) = \sum_{\alpha \in \mathbb{F}_{2^n}} f(\alpha) \left(1 - (x - \alpha)^{2^n - 1}\right).$$

En règle générale, nous utiliserons la même notation pour parler d'une fonction ou du polynôme la représentant. Nous préciserons à chaque fois si le contexte n'est pas assez clair.

L'exemple suivant présente une fonction des plus intéressantes sur $\mathbb{F}_{2^n}$, il s'agit de la fonction *inverse*, qui renvoie l'inverse multiplicatif d'un élément.

Exemple 1.3.12. La fonction *inverse pour la multiplication* dans le corps $\mathbb{F}_{2^n}$ est donnée par

$$\begin{array}{rccc} f: & 0 & \mapsto & 0 \\ & x \in \mathbb{F}_{2^n}^* & \mapsto & x^{-1}. \end{array}$$

Le polynôme représentant cette fonction est

$$f(x) = x^{2^n - 2} \in \mathbb{F}_{2^n}[x].$$

Par abus de langage, dans ce document, nous appellerons la fonction $x \mapsto x^{-1}$ la fonction inverse, même si le cas où $x = 0$ est un cas limite.

Nous appellerons *noyau* d'une fonction F sur $\mathbb{F}_{2^n}$, l'ensemble des racines du polynômes $F(x) \in \mathbb{F}_{2^n}[x]$.

Polynômes linéarisés

Les polynômes linéarisés sont parmi les classes remarquables de polynômes sur le corps fini $\mathbb{F}_{2^n}$. Dans le reste de ce document, nous reparlerons beaucoup de cette classe de fonctions.

Définition 1.3.13 (Polynôme linéarisé)**.** Soit le corps $\mathbb{F}_{2^n}$ une extension du corps $\mathbb{F}_{2^m}$. Un polynôme de la forme

$$L(x) = \sum_{i=0}^{m/n-1} \ell_i x^{2^{mi}} \in \mathbb{F}_{2^n}[x], \tag{1.4}$$

est appelé *polynôme linéarisé* sur $\mathbb{F}_{2^m}$, ou bien parfois [146] 2^m-polynôme. Le polynôme $L(x) + c$, $c \in \mathbb{F}_{2^n}$ est appelé *polynôme affine*.

Les polynômes linéarisés (resp. affines) sur $\mathbb{F}_{2^m}$ représentent les fonctions $\mathbb{F}_{2^m}$-linéaires (resp. affines), c'est-à-dire les fonctions linéaires (resp. affines) sur $\mathbb{F}_{2^m}$. En effet, pour tout $x, y \in \mathbb{F}_{2^n}$ et tout $\mu \in \mathbb{F}_{2^m}$, nous avons

$$\begin{aligned} L(\mu x + y) &= \sum_{i=0}^{m/n-1} \ell_i (\mu x + y)^{2^{mi}} = \sum_{i=0}^{m/n-1} \ell_i \left((\mu x)^{2^{mi}} + y^{2^{mi}} \right) \\ &= \mu \left(\sum_{i=0}^{m/n-1} \ell_i x^{2^{mi}} \right) + \left(\sum_{i=0}^{m/n-1} \ell_i y^{2^{mi}} \right) \\ &= \mu L(x) + L(y). \end{aligned}$$

Rappelons que l'image d'une fonction (*i.e.* ensemble des valeurs prises par la fonction) linéaire est un sous-espace vectoriel de $\mathbb{F}_{2^n}$. De la même manière, l'image d'une fonction affine est le translaté d'un sous-espace de $\mathbb{F}_{2^n}$.

En particulier, nous remarquons que la fonction $\mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}$ est $\mathbb{F}_{2^m}$-linéaire sur $\mathbb{F}_{2^n}$.

Nous définissons une notion importante lorsque nous parlons de polynômes linéarisés.

Définition 1.3.14 (Fonction adjointe)**.** Soit un polynôme linéarisé sur $\mathbb{F}_2$ $L(x) = \sum_{i=0}^{n-1} \ell_i x^{2^i} \in \mathbb{F}_{2^n}[x]$. Nous définissons alors la fonction adjointe de la fonction L, notée L^* qui admet pour représentation polynomiale

$$L^*(x) = \sum_{i=0}^{n-1} \ell_{n-i}^{2^i} x^{2^i}.$$

La fonction adjointe d'une fonction linéaire possède entre autre une propriété intéressante lorsque nous composons avec la fonction trace.

Proposition 1.3.15. *Soit une fonction linéaire $L : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, et soit un élément $\lambda \in \mathbb{F}_{2^n}$. Alors, pour tout $x \in \mathbb{F}_{2^n}$, nous avons*

$$Tr(\lambda L(x)) = Tr(L^*(\lambda) x).$$

Notons que la fonction adjointe de L^* est la fonction $(L^*)^* = L$.

Forme trace des fonctions booléennes

À la section 1.2, nous avons évoqué l'existence d'une représentation des fonctions booléennes sous forme d'une trace.

Propriété 1.3.16. *Toute fonction booléenne à n variables, $f \in \mathcal{B}_n$, peut s'écrire*

$$\begin{array}{rccl} f: & \mathbb{F}_2^n & \to & \mathbb{F}_2 \\ & x & \mapsto & Tr\left(F\left(\varphi_{\mathcal{B}}^{-1}(x)\right)\right), \end{array}$$

pour une certaine fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et une certaine base, $\mathcal{B}$, de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$.

Cette forme n'est pas unique, c'est-à-dire qu'il existe plusieurs fonctions, disons F et G, telles que $f = \mathrm{Tr}(F) = \mathrm{Tr}(G)$.

Exemple 1.3.17. Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et soit $P : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ une fonction permutant l'hyperplan H_1 et laissant invariant le complémentaire de cet hyperplan, $\overline{H_1}$. Alors, les fonctions booléennes $x \mapsto \mathrm{Tr}(F(x))$ et $x \mapsto \mathrm{Tr}((P \circ F)(x))$ sont identiques. En effet, la fonction P est telle que pour tout $y \in \mathbb{F}_{2^n}$, $\mathrm{Tr}(y) = \mathrm{Tr}(P(y))$, donc

$$\mathrm{Tr}(F(x)) = \mathrm{Tr}(P \circ F(x)) \text{ pour tout } \quad x \in \mathbb{F}_{2^n}.$$

Avec cette représentation des fonctions booléennes, il est aisé de caractériser les fonctions composantes d'une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Il s'agit des fonctions $\tilde{f}_\lambda$ données par

$$\begin{array}{rccl} \tilde{f}_\lambda : & \mathbb{F}_{2^n} & \to & \mathbb{F}_2 \\ & x & \mapsto & \mathrm{Tr}\left(\lambda F(x)\right), \end{array}$$

pour tout $\lambda \in \mathbb{F}_{2^n}$.

Nous pouvons faire la comparaison entre cette forme et la définition que nous avions donnée des fonctions composantes à la définition 1.3.2. Si $\mathcal{B}$ est la base de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$, alors à la fonction $\tilde{f}_\lambda$, $\lambda \in \mathbb{F}_{2^n}$, définie comme ci-dessus correspond la fonction composante $f_{\varphi_{\mathcal{A}}^{-1}(\lambda)}$ dans la définition 1.3.2, où $\mathcal{A}$ est la base duale de la base $\mathcal{B}$.

Nous pouvons dorénavant caractériser l'équivalent des fonctions coordonnées dans $\mathbb{F}_{2^n}$ de la fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Ce sont les fonctions $x \mapsto \mathrm{Tr}(\alpha_i F(x))$, $0 \leq i < n$ où les $\alpha_i \in \mathbb{F}_{2^n}$ sont les éléments de la base duale $\mathcal{A}$.

Degré algébrique des fonctions sous forme polynomiale

Lorsque nous parlions de fonctions booléennes vectorielles, nous avions la notion de *degré algébrique*. Il est possible de définir le degré algébrique d'une fonction représentée sous forme polynomiale. A première vue, la relation entre le lemme suivant et la définition que nous avons donnée précédemment pour les fonctions booléennes vectorielles sur $\mathbb{F}_2^n$ n'est pas évidente. La preuve peut être trouvée dans [59] de Claude Carlet ou encore [71, Chapitre 5] de Gohar Kyureghyan.

Lemme 1.3.18 (Degré algébrique)**.** *Soit F une fonction de $\mathbb{F}_{2^n}$ dont la forme polynomiale est $F(x) = \sum_{i=0}^{2^n-1} c_i x^i \in \mathbb{F}_{2^n}[x]$. Le degré algébrique de la fonction F, noté* $\deg(F)$*, est défini par*

$$\deg(F) = \max_{0 \leq i \leq 2^n-1} \{wt(i) \mid c_i \neq 0\}.$$

Démonstration. Prenons $\{\beta_0, \ldots, \beta_{n-1}\}$ comme base de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$, ce qui nous permet de représenter chaque élément $x \in \mathbb{F}_{2^n}$ comme $x = x_0\beta_0 + \cdots + x_{n-1}\beta_{n-1}$, avec $x_i \in \mathbb{F}_2$. Nous pouvons ainsi représenter la fonction F dans l'anneau de polynôme multivarié $\mathbb{F}_{2^n}[x_0, \ldots, x_{n-1}]/(x_0^2 + x_0, \ldots, x_{n-1}^2 + x_{n-1})$:

$$\begin{aligned} F(x) &= \sum_{i=0}^{2^n-1} c_i x^i \\ &= \sum_{i=0}^{2^n-1} c_i (x_0\beta_0 + \cdots + x_{n-1}\beta_{n-1})^i \\ &= \sum_{i=0}^{2^n-1} c_i (x_0\beta_0 + \cdots + x_{n-1}\beta_{n-1})^{\sum_{k=0}^{n-1} i_k 2^k} \end{aligned}$$

$$= \sum_{i=0}^{2^n-1} c_i \prod_{k=0}^{n-1} (x_0\beta_0^{2^k} + \cdots + x_{n-1}\beta_{n-1}^{2^k})^{i_k}.$$

Cette dernière égalité montre que le degré algébrique de la fonction F, $\deg(F)$, ne peut excéder $\max_{0\leq i\leq 2^n-1}\{wt(i) \mid c_i \neq 0\}$, que nous notons w.

Nous remarquons maintenant que le nombre de fonctions booléennes à n variables de degré algébrique au plus w est $2^{\sum_{i=0}^{w}\binom{n}{i}}$, impliquant donc qu'il y a

$$2^{n\sum_{i=0}^{w}\binom{n}{i}},$$

fonctions vectorielles $\mathbb{F}_2^n \to \mathbb{F}_2^n$ de degré au plus w. Il existe aussi exactement le même nombre de polynômes univariés $F(x) = \sum_{i=0}^{2^n-1} c_i x^i \in \mathbb{F}_{2^n}[x]$ tels que pour tout $0 \leq i \leq 2^n - 1$ nous avons

$$c_i \neq 0 \Rightarrow wt(i) \leq w.$$

Comme, $\deg(F) \leq w$, qu'il y a le même nombre de fonctions $\mathbb{F}_2^n \to \mathbb{F}_2^n$ de degré algébrique w et de fonctions $\mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ dont le poids maximal des exposants est w, et que les espaces $\mathbb{F}_2^n$ et $\mathbb{F}_{2^n}$ sont isomorphes, nous en concluons que le degré algébrique de la fonction F est

$$\deg(F) = w.$$

□

Dans la suite de ce document, lorsque nous parlerons de degré, il s'agira essentiellement du degré algébrique de fonctions. Nous préciserons lorsque le besoin s'en fera ressentir s'il s'agit du degré algébrique de la fonction, ou du degré polynomial (univarié) du polynôme la représentant.

Avec l'ensemble des éléments de cette sous-section, nous sommes maintenant capables de parler indifféremment de fonctions booléennes vectorielles ou de fonctions sur $\mathbb{F}_{2^n}$ sous forme polynomiale. Dans le reste de ce document, le contexte clarifiera si nous sommes dans $\mathbb{F}_2^n$ ou dans $\mathbb{F}_{2^n}$ et nous abuserons parfois du passage de l'un à l'autre sans plus d'explications.

1.4. Bijectivité

Dans cette section, nous allons nous intéresser aux fonctions permutant le corps $\mathbb{F}_{2^n}$, c'est-à-dire aux fonctions

$$\begin{array}{rccc} F: & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & F(x), \end{array}$$

qui sont bijectives.

Définition 1.4.1 (Polynômes de permutation)**.** Un polynôme $F \in \mathbb{F}_{2^n}[x]$ est dit *polynôme de permutation* de $\mathbb{F}_{2^n}$ si la fonction qu'il représente est bijective.

Pour les lecteurs voulant approfondir leurs connaissances sur les polynômes de permutations, nous les renvoyons vers la thèse de Yann Laigle-Chapuy [138, Première Partie], par exemple, dont les premiers chapitres fournissent un bon état de l'art. Le chapitre dédié aux polynômes de permutation du livre de Rudolf Lidl et Harald Niederreiter [146, Chapitre 7] est aussi très complet. De nombreux chercheurs ont travaillé sur ce sujet.

Nous rappelons ici les résultats principaux, comme par exemple certaines constructions importantes, ou bien le problème de la recherche de l'inverse pour la composition. Puisque la fonction $x \mapsto x + \alpha$ est bijective sur $\mathbb{F}_{2^n}$ quel que soit $\alpha \in \mathbb{F}_{2^n}$, nous pouvons traiter sans perte de généralité le cas des fonctions F telles que $F(0) = 0$.

Il existe des critères plus ou moins triviaux pour savoir si un polynôme est un polynôme de permutation du corps $\mathbb{F}_{2^n}$, ou de manière équivalente si une fonction sur $\mathbb{F}_{2^n}$ est bijective. Parmi ceux là, nous pouvons citer la propriété suivante.

Propriété 1.4.2. *Le polynôme $F(x) \in \mathbb{F}_{2^n}[x]$ est un polynôme de permutation de $\mathbb{F}_{2^n}$ si et seulement si pour tout élément $\alpha \in \mathbb{F}_{2^n}$, le polynôme $F(x) + \alpha$ a une racine dans $\mathbb{F}_{2^n}$.*

En effet, si une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ a pour image $\mathbb{F}_{2^n}$ exactement, alors elle est bijective. Le théorème suivant est quant à lui un critère non trivial bien connu.

Théorème 1.4.3 (Critère d'Hermite)**.** *Le polynôme $F \in \mathbb{F}_{2^n}[x]$ est un polynôme de permutation de $\mathbb{F}_{2^n}$ si et seulement si les deux conditions suivantes sont vérifiées :*

1. *$F(x)$ a exactement une racine dans $\mathbb{F}_{2^n}$;*
2. *pour chaque entier impair $1 \leq t \leq 2^n - 2$, la réduction du polynôme $F(x)^t$ modulo $x^{2^n} - x$ est de degré $\leq 2^n - 2$.*

Corollaire 1.4.4. *Si l'entier $d > 1$ est un diviseur de $2^n - 1$, alors il n'existe pas de polynôme de permutation de $\mathbb{F}_{2^n}$ de degré univarié d.*

Il existe un autre critère pour déterminer si une fonction permute le corps $\mathbb{F}_{2^n}$, dont une preuve peut être trouvé dans [59].

Proposition 1.4.5. *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. La fonction F est bijective si et seulement si toutes ses fonctions composantes (non nulles) sont équilibrées.*

Ces deux critères (théorème 1.4.3 et proposition 1.4.5) sont calculatoirement difficiles à mettre en pratique lorsqu'il s'agit de vérifier si une fonction, ou une classe de fonctions, permute le corps $\mathbb{F}_{2^n}$. En effet, ils demandent un grand nombre d'opérations à effectuer. Il n'en reste pas moins que ce sont des outils relativement efficaces pour faire des preuves plus formelles.

Proposition 1.4.6. *Soit un élément $\alpha \in \mathbb{F}_{2^n}^*$. La fonction qui réalise la transposition (0α) dans $\mathbb{F}_{2^n}$ est donnée par*

$$x \mapsto \alpha^2 \left[\left((x + \alpha)^{-1} + \alpha^{-1} \right)^{-1} + \alpha \right]^{-1}.$$

Démonstration. Notons la fonction ci dessus T_α. Il est évident que $T_\alpha(0) = \alpha$ et $T_\alpha(\alpha) = 0$. Supposons maintenant que $y \in \mathbb{F}_{2^n} \backslash \{0, \alpha\}$. Nous avons alors

$$\begin{aligned}
T_\alpha(y) &= \alpha^2 \left[\left(y((y + \alpha)\alpha)^{-1} \right)^{-1} + \alpha \right]^{-1} \\
&= \alpha^2 \left[y^{-1} \left((y + \alpha)\alpha \right) + \alpha \right]^{-1} \\
&= \alpha^2 \left[y^{-1} \left((y + \alpha)\alpha \right) + y\alpha y^{-1} \right]^{-1} \\
&= \alpha^2 y \left[\alpha^2 \right]^{-1} \\
&= y.
\end{aligned}$$

□

Avec cette proposition, nous voyons que les permutations les plus simples peuvent avoir des représentations relativement complexes.

Ils existent aussi des polynômes qui sont de permutations pour une infinité de corps.

Définition 1.4.7 (Polynôme exceptionnel)**.** Un polynôme de permutation $F(x) \in \mathbb{F}_{2^n}[x]$ est appelé *exceptionnel* s'il permute un nombre infini d'extensions de $\mathbb{F}_{2^n}$.

Exemple 1.4.8. Le polynôme $x^3 \in \mathbb{F}_2[x]$ est exceptionnel puisqu'il permute tous les corps $\mathbb{F}_{2^n}$ avec n impair.

1.4.1. Polynômes linéarisés de permutation

Cette classe de polynômes, assez facile à représenter, pose tout de même certains problèmes.

Proposition 1.4.9. *Un polynôme linéarisé sur $\mathbb{F}_2$ est un polynôme de permutation de $\mathbb{F}_{2^n}$ si et seulement si 0 est son unique racine.*

Démonstration. Un polynôme linéarisé, $L(x) \in \mathbb{F}_{2^n}[x]$, représente une fonction linéaire. L'élément nul est toujours une racine de ce polynôme (puisqu'il ne contient pas de terme constant). Il s'ensuit que si L est de permutation sur $\mathbb{F}_{2^n}$ alors 0 est l'unique racine de L. Réciproquement, si 0 est la seule racine, nous avons

$$\forall x, y \in \mathbb{F}_{2^n}, \qquad L(x) = L(y) \Rightarrow L(x+y) = 0 \Rightarrow x = y.$$

□

Propriété 1.4.10. *L'inverse pour la composition d'une fonction linéaire bijective est une fonction linéaire.*

Démonstration. Soit une fonction bijective $L : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ linéaire sur $\mathbb{F}_{2^m}$. Pour tout $x, y \in \mathbb{F}_{2^n}$, il existe $x', y' \in \mathbb{F}_{2^n}$ tels que $x = L(x')$ et $y = L(y')$. Pour tout $\lambda \in \mathbb{F}_{2^m}$, nous avons donc

$$\begin{aligned} L^{-1}(x + \lambda y) &= L^{-1}(L(x') + \lambda L(y')) = L^{-1}(L(x' + \lambda y')) \\ &= x' + \lambda y' = L^{-1}(x) + \lambda L^{-1}(y). \end{aligned}$$

□

Propriété 1.4.11. *Soit une polynôme $\mathbb{F}_2$-linéaire $L(x) \in \mathbb{F}_{2^n}[x]$. Alors le polynôme L est de permutation si et seulement si L^* est de permutation.*

Démonstration. Puisque la fonction adjointe de L^* est L, il est suffisant de prouver la suffisance de l'équivalence. Supposons donc que le polynôme $L(x) \in \mathbb{F}_{2^n}$ est de permutation, alors pour tout élément $\lambda \in \mathbb{F}_{2^n}^*$, nous avons

$$\mathrm{Tr}(\lambda y) = \mathrm{Tr}\left(\lambda L(L^{-1}(y))\right) = \mathrm{Tr}\left(L^*(\lambda) L^{-1}(y)\right) = \mathrm{Tr}\left(L^{-1*}(L^*(\lambda)) y\right) \forall y \in \mathbb{F}_{2^n},$$

d'où $L^{-1*}(L(\lambda)) = \lambda$ pour tout $\lambda \in \mathbb{F}_{2^n}^*$. □

Malgré le critère bijectif simple (*i.e.* une unique racine), il est *a priori* difficile de déterminer des classes de polynômes linéarisés (affines) de permutation sur $\mathbb{F}_{2^n}$. Il est particulièrement difficile d'écrire formellement les inverses des polynômes linéarisés de permutation. Des auteurs tels que Pingzhi Yuan et Cunsheng Ding [215] et Baofeng Wu [208] se sont intéressés à l'étude de certaines classes de permutations linéaires sur $\mathbb{F}_{2^n}$. Au Chapitre 3, nous verrons comment construire des permutations à partir de trinômes affines, c'est-à-dire de polynômes de la forme $x^{2^i} + \alpha x^{2^j} + \gamma \in \mathbb{F}_{2^n}[x]$.

1.4.2. Permutations complètes

Certaines fonctions conservent leur bijectivité lorsque nous leur ajoutons la fonction identité.

Définition 1.4.12 (Permutations complètes)**.** Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. La fonction F est appelée une *permutation complète* si les fonctions $x \mapsto F(x)$ et $x \mapsto F(x)+x$ sont toutes deux bijectives sur $\mathbb{F}_{2^n}$.

Récemment, Gaofei Wu, Nian Li, Tor Helleseth et Yuqing Zhang [209] se sont intéressés à déterminer de nouvelles classes de permutations complètes à partir de fonctions monomiales.

La recherche de permutations complètes est en fait une instance à l'intersection de problèmes plus généraux. Comme par exemple :

Problème 1.4.13. Étant donnée une fonction de $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$:

- Pour quelles fonctions $L : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ linéaire sur un sous-corps $\mathbb{F}_{2^m}$ la fonction $F + L$ permute-t-elle $\mathbb{F}_{2^n}$?
- Si F permute le corps $\mathbb{F}_{2^n}$, quelles sont les fonctions bijectives $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telles que $F + G$ permute le corps $\mathbb{F}_{2^n}$?
- ...

Il est donc naturel que la communauté en mathématiques discrètes s'y intéresse, d'autant plus que les permutations complètes sont des objets remarquables qui trouvent des applications en combinatoire, théorie des graphes et en cryptographie [200]. Les permutations complètes sont parfois appelées *orthomorphismes*, suivant le contexte.

1.4.3. Inverses pour la composition

Nous ne pouvons étudier les fonctions bijectives sur $\mathbb{F}_{2^n}$ sans nous intéresser à leurs inverses. Dans un souci de clarté, nous allons tout d'abord définir la notion de composition de fonctions de manière générale.

Définition 1.4.14. Soient des ensembles non vides A, B et C. Soient des fonctions $F : A \to B$ et $G : C \to A$. Nous appelons la *composition* de la fonction F par la fonction G, la fonction

$$\begin{array}{rccl} F \circ G : & C & \to & B \\ & x & \mapsto & (F \circ G)(x) = F(G(x)). \end{array}$$

Définition 1.4.15. Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ bijective. Nous appelons *inverse pour la composition* (parfois inverse compositionnelle) de la fonction F, et notons F^{-1}, la fonction de $\mathbb{F}_{2^n}$ vérifiant

$$\forall x \in \mathbb{F}_{2^n}, \qquad F\left(F^{-1}(x)\right) = F^{-1}\left(F(x)\right) = x.$$

Trouver le polynôme représentant l'inverse d'une fonction bijective, de manière efficace, est en général un problème difficile qui fut énoncé par Gary Mullen [159]. Au chapitre suivant, nous nous intéressons à expliciter les inverses pour la composition des monômes de permutations. Au chapitre 3, nous donnons la représentation des inverses de certaines classes de polynômes de permutations creux (ayant peu de coefficients non nuls).

Dans [162], Amela Muratović-Ribić traite des coefficients des inverses des polynômes de permutations, répondant ainsi en partie au problème posé par Gary Mullen [159]. Nous allons rappeler ici certains de ses résultats dans le cas binaire. Qiang Wang [206] étudie aussi un peu plus précisément les coefficients de l'inverse des fonctions du type $x \mapsto x^r f(x^s)$ sur $\mathbb{F}_q$.

Nous rappelons d'abord une proposition importante en combinatoire.

Proposition 1.4.16 (Formule du multinôme de Newton)**.** *Soient $a_0, \ldots, a_{\ell-1}$, ℓ éléments d'un anneau commutatif, et soit un entier positif non nul m. Alors*

$$(a_0 + a_2 + \cdots + a_{\ell-1})^m = \sum_{k_0+\cdots+k_{\ell-1}=m} \frac{m!}{k_0! \ldots k_{\ell-1}!} \prod_{0 \le i < \ell} a_i^{k_i},$$

où la somme parcourt toutes les combinaisons des entiers positifs k_i, $0 \le i < \ell$.

Remarque 1.4.17. Nous remarquons que dans le cas où $\ell = 2$, la proposition 1.4.16 correspond évidemment à la formule du binôme de Newton.

Théorème 1.4.18 ([162])**.** *Soit un polynôme de permutation*

$$F(x) = a_0 + a_1 x + \cdots + a_{2^n-2} x^{2^n-2} \in \mathbb{F}_{2^n}[x].$$

Alors, son inverse pour la composition est donné par $F^{-1}(x) = b_0 + b_1 x + \cdots + b_{2^n-2} x^{2^n-2}$ avec

$$b_j = \sum \frac{(2^n - 1 - j)!}{k_0! k_1! \ldots k_{2^n-2}!} \prod_{0 \le i \le 2^n-2} a_i^{k_i}, \tag{1.5}$$

où la somme parcourt toutes les combinaisons des entiers positifs $k_0, \ldots, k_{2^n-2}$ telles que

$$k_0 + k_1 + \cdots + k_{2^n-2} = 2^n - 1 - j$$
$$k_1 + 2k_2 + \cdots + (2^n - 2)k_{2^n-2} \equiv 2^n - 2 \pmod{2^n - 1}.$$

Même si cette formule peut paraître compliquée et difficile à utiliser, elle s'avère beaucoup plus pratique lorsque nous considérons des polynômes creux. Il n'en reste pas moins qu'être capable d'écrire formellement les inverses de polynômes de permutation d'une certaine classe est un problème difficile qui mérite une étude au cas par cas.

Propriété 1.4.19. *Soit F une fonction permutant le corps $\mathbb{F}_{2^n}$ telle que $F(x) \in \mathbb{F}_{2^m}[x]$ où $\mathbb{F}_{2^m}$ est un sous-corps de $\mathbb{F}_{2^n}$. Alors la fonction F^{-1} est telle que $F^{-1}(x) \in \mathbb{F}_{2^m}[x]$.*

Ce résultat, en apprence non-trivial est une conséquence directe du théorème 1.4.18.

1.5. Critères cryptographiques

Dans cette section, nous allons voir les principaux critères que des fonctions à usage cryptographique doivent vérifier. Ces critères sont en relation directe avec les principales attaques connues à ce jour et correspondent à différentes formes de non-linéarité. Nous verrons ainsi comment une fonction jouant le rôle de fonction interne d'une primitive de cryptographie symétrique, peut résister aux principales attaques (par exemple différentielles, linéaires,...).

Dans cette section, nous ne parlerons que très peu du rôle important en cryptographie du degré algébrique. Nous noterons tout de même que dans son article novateur de 1949 [183], Claude Shannon explique que « casser un "bon" chiffrement, doit demander autant de travail que la résolution d'un système complexe d'équations en un grand nombre de variables ». Les *attaques algébriques*, introduites par Nicolas Courtois et Joseph Pieprzyk [77] pour tenter de cryptanalyser l'AES, utilisent des relations de faible degré algébrique entre les entrées et les sorties d'une fonction interne pour *linéariser*[4] le système d'équations d'un chiffrement. Dans [125], Lars R. Knudsen montre qu'un degré algébrique faible peut être exploité pour monter une attaque, qu'il appelle cryptanalyse différentielle d'ordre supérieur. Cette attaque est basée sur le concept des *dérivées d'ordre supérieur* (voir la définition de *dérivée* 1.5.1) généralement attribué à Xuejia Lai [134] même s'il était présent dans la thèse de John Dillon [82].

1.5.1. Uniformité différentielle

Définition 1.5.1 (Dérivée)**.** Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, et soit un élément $\alpha \in \mathbb{F}_{2^n}$. La *différentielle*, ou *dérivée discrète* de la fonction F en la direction α est la fonction, notée Δ_α, donnée par

$$\begin{array}{rccl} \Delta_\alpha F : & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & F(x) + F(x+\alpha). \end{array}$$

Notons que la dérivée en la direction 0 donne forcément la fonction nulle, quelle que soit la fonction que l'on dérive.

La propriété suivante explicite le parallèle qu'il existe entre la dérivée polynomiale "classique", et cette dérivée en une direction. Cette dernière, plutôt que de faire diminuer le degré univarié, fait diminuer le degré algébrique de la fonction que nous dérivons.

Propriété 1.5.2. *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, et soit un élément $\alpha \in \mathbb{F}_{2^n}$.*

$$\deg(F) > \deg(\Delta_\alpha F).$$

Définissons maintenant une notion essentielle en cryptographie symétrique.

Définition 1.5.3 (Uniformité différentielle [166])**.** Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Nous définissons pour tout α et $\beta \in \mathbb{F}_{2^n}$ la valeur $\delta_F(\alpha, \beta)$ par

$$\delta_F(\alpha, \beta) = |\{x \in \mathbb{F}_{2^n} \mid \Delta_\alpha F(x) = \beta\}|.$$

Le maximum de ces valeurs pour tout $\alpha \neq 0, \beta \in \mathbb{F}_{2^n}$, noté $\delta(F)$, est appelé *l'uniformité différentielle* de la fonction F,

$$\delta(F) = \max_{\alpha \neq 0, \beta \in \mathbb{F}_{2^n}} \delta_F(\alpha, \beta).$$

La fonction F est alors dite $\delta(F)$-différentiellement uniforme.

Puisque nous sommes en caractéristique 2, les valeurs $\delta_F(\alpha, \beta)$ sont forcément paires puisque si x est solution de l'équation $\Delta_\alpha F(x) = \beta$, $x + \alpha$ est une autre solution. La plus petite valeur que peut donc prendre $\delta(F)$ est 2.

[4] technique consistant à remplacer tous les monômes de degré > 1 par une nouvelle variable indépendante.

Définition 1.5.4 (Fonctions APN)**.** Une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que $\delta(F) = 2$ est appelée *presque*[5] *parfaitement non-linéaire*, ou plus brièvement APN (pour "Almost Perfect Nonlinear" en anglais).

Remarque 1.5.5. Calculer les valeurs δ_F permet évidemment de vérifier la bijectivité d'une fonction. En effet, prenons une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Il existe $\alpha \in \mathbb{F}_{2^n}$ tel que l'équation $\Delta_\alpha F(x) = 0$ a des solutions en x si et seulement si la fonction F n'est pas bijective sur $\mathbb{F}_{2^n}$.

En cryptographie symétrique, le concept de dérivée en une certaine direction d'une fonction permet d'étudier le comportement de la différence entre deux valeurs de sortie d'une boîte-S lorsque leurs antécédents ont une certaine différence connue. L'uniformité différentielle mesure donc la résistance d'une fonction aux attaques différentielles (voir chapitre 5). Il se trouve que les fonctions APN sont les fonctions qui résistent le mieux à ces attaques [166, 63]. En effet, en termes de probabilités, un potentiel attaquant est moins à même de savoir quelle sera la différence en sortie d'une boîte-S si aucune différence en sortie n'apparaît significativement plus souvent que les autres. L'exemple suivant va nous permettre de mieux comprendre.

Exemple 1.5.6. Considérons une fonction $S : \mathbb{F}_{2^4} \to \mathbb{F}_{2^4}$ qui servirait de boîte-S dans un chiffrement par blocs. La valeur

$$\frac{\delta_S(\alpha, \beta)}{2^4},$$

correspond à la probabilité d'obtenir une différence β en sortie de la boîte-S à partir d'une différence α en entrée. Si cette probabilité est trop élevée pour un certain couple (α, β), un attaquant pourrait s'en servir pour obtenir des informations sur les états internes d'une fonction de chiffrement lors de son utilisation. Nous verrons plus de détails concernant les attaques différentielles dans la seconde partie de ce document.

Remarque 1.5.7. Les fonctions linéaires de $\mathbb{F}_{2^n}$ sont 2^n-différentiellement uniformes, mais ce ne sont pas les seules !

Nous pouvons aussi caractériser les fonctions APN par leurs dérivées. Puisqu'une fonction est APN si et seulement si les équations $\Delta_\alpha F(x) = \beta$ ont zéro ou deux solutions quels que soient $\alpha \neq 0$ et $\beta \in \mathbb{F}_{2^n}$, cela signifie que l'image des fonctions $\Delta_\alpha F$, $\alpha \neq 0$, ne contient que la moitié des éléments du corps $\mathbb{F}_{2^n}$. De plus, chaque élément dans cette image possède exactement deux antécédents dans $\mathbb{F}_{2^n}$. Nous appellerons ces fonctions des *fonctions 2-to-1.*

Propriété 1.5.8. *Une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est APN si et seulement si toutes ses dérivées (non nulles) sont 2-to-1.*

Cette dernière propriété peut être améliorée. En effet, la vérification de toutes les dérivées n'est pas nécessaire pour montrer qu'une fonction est APN. Ce résultat est dû à Thomas Beth et Cunsheng Ding [15], mais l'énoncé que nous proposons est dû à Gohar Kyureghyan [71, Chapitre 5].

[5] en caractéristique impaire, il est possible d'avoir $\delta(F) = 1$. De telles fonctions sont appelées *planaires.* Aussi, les fonctions *parfaitement non-linéaire* sont les fonctions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^m}$, $n > m$, avec $\delta(F) = 2^{n-m}$. Elles coïncident avec les fonctions courbes.

Théorème 1.5.9 ([71, Chapitre 5, Théorème 2.1]). *Soit H un hyperplan de $\mathbb{F}_{2^n}$. Une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est APN si et seulement si les dérivées $\Delta_\alpha F$ sont 2-to-1 pour tout élément $\alpha \neq 0 \in H$.*

Notons que la plupart des fonctions APN connues à ce jour sont soit monomiales (voir chapitre suivant), binomiales [93] ou quadratiques (voir paragraphe dédié ci-dessous), à équivalence près (voir section 1.5.3). La détermination de nouvelles fonctions APN est un sujet de recherche rassemblant de nombreux chercheurs et amenant un nombre considérable de travaux depuis de nombreuses années. En effet, les fonctions APN sont des objets remarquables, apparaissant dans plusieurs domaines, et créant des liens entre ceux-là.

Nous venons de voir que les fonctions APN correspondent aux fonctions dont le biais des différences en sortie est le mieux réparti (voir exemple 1.5.6), d'où leur intérêt en cryptographie symétrique. Le lien unissant les fonctions APN à la théorie des codes est dû à Claude Carlet, Pascale Charpin et Victor Zinoviev [60].

Théorème 1.5.10 ([60, Théorème 5]). *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que $F(0) = 0$. Soit C_F le code binaire défini par sa matrice de parité de taille $2n \times (2^n - 1)$:*

$$H = \begin{pmatrix} \varphi_{\mathcal{B}}(1)^\top & \varphi_{\mathcal{B}}(\alpha)^\top & \varphi_{\mathcal{B}}(\alpha^2)^\top & \dots & \varphi_{\mathcal{B}}(\alpha^{2^n-2})^\top \\ \varphi_{\mathcal{B}}(F(1))^\top & \varphi_{\mathcal{B}}(F(\alpha))^\top & \varphi_{\mathcal{B}}(F(\alpha^2))^\top & \dots & \varphi_{\mathcal{B}}(F(\alpha^{2^n-2}))^\top \end{pmatrix},$$

où α est un élément primitif du corps $\mathbb{F}_{2^n}$, $\mathcal{B}$ est une base de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$ et $\varphi_{\mathcal{B}}$ est l'isomorphisme défini à l'équation (1.3).
Alors la fonction F est APN si et seulement si la distance minimale du code C_F est 5.

Permutations APN

En cryptographie symétrique, la bijectivé est un autre critère important. Cependant, il est difficile d'obtenir des fonctions ayant des propriétés différentielles intéressantes et qui soient bijectives. Lorsque n est impair, il existe très peu d'exemples de permutations APN qui ne soient pas monomiales.

Il a été longtemps conjecturé que les permutations APN n'existaient même pas sur $\mathbb{F}_{2^n}$ lorsque n est pair. Dans [14] (notamment le théorème 3), Thierry Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy font un état de l'art assez précis de ce qui est connu des permutations APN en dimension paire. Il n'existe par exemple pas de permutations APN dans $\mathbb{F}_{2^4}$. Ce cas a été traité exhaustivement, par Xiang-dong Hou [91].

En 2009 cependant, John F. Dillon et al. [45] dévoilent qu'il existe bel et bien une permutation APN dans $\mathbb{F}_{2^6}$. C'est à ce jour encore, à équivalence près (voir sous-section 1.5.3), le seul exemple que nous ayons de permutation APN dans $\mathbb{F}_{2^n}$ avec n pair. Elle est de la forme

$$\begin{aligned} D : x \mapsto\ & \alpha^{36}x^{60} + \alpha^{44}x^{58} + \alpha^{40}x^{57} + \alpha^{55}x^{56} + \alpha^{26}x^{54} + \alpha^{23}x^{53} + \alpha^{36}x^{52} + \alpha^{23}x^{51} \\ & + \alpha^{17}x^{50} + \alpha^{54}x^{49} + \alpha^{14}x^{48} + \alpha^{21}x^{46} + \alpha^{53}x^{45} + \alpha^{21}x^{44} + \alpha^{7}x^{43} + \alpha^{57}x^{42} \\ & + \alpha^{8}x^{41} + \alpha^{10}x^{40} + \alpha^{12}x^{39} + \alpha^{20}x^{38} + \alpha^{52}x^{37} + \alpha^{46}x^{36} + \alpha^{27}x^{35} + \alpha^{44}x^{34} \\ & + \alpha^{18}x^{33} + \alpha^{57}x^{32} + \alpha^{28}x^{30} + \alpha^{44}x^{29} + \alpha^{42}x^{28} + \alpha^{26}x^{27} + \alpha^{20}x^{26} + \alpha^{10}x^{25} \\ & + \alpha^{45}x^{24} + x^{23} + \alpha^{7}x^{22} + \alpha^{57}x^{21} + \alpha^{21}x^{20} + \alpha^{22}x^{19} + \alpha^{6}x^{17} + \alpha^{8}x^{16} + \alpha^{43}x^{15} \end{aligned}$$

$$+\alpha^{42}x^{13}+\alpha^{47}x^{12}+\alpha^{56}x^{11}+\alpha^{38}x^{10}+\alpha^{36}x^{8}+\alpha^{47}x^{7}+\alpha^{4}x^{6}+\alpha^{8}x^{5}+\alpha^{23}x^{4}$$
$$+\alpha^{39}x^{3}+\alpha^{52}x^{2}+\alpha^{59}x,$$

où α est un élément primitif de $\mathbb{F}_{2^6}$ vérifiant $\alpha^6=\alpha^4+\alpha^3+\alpha+1$. Le problème est maintenant le suivant :

Problème 1.5.11. Existe-t-il des permutations APN dans $\mathbb{F}_{2^n}$ pour tout $n>6$?

Fonctions quadratiques APN

Les fonctions quadratiques sont les fonctions de degré algébrique 2. La plupart des fonctions APN connues qui ne sont pas monomiales ni binomiales se trouvent être quadratiques[6]. Parmi les constructions remarquables de fonctions quadratiques APN, nous pouvons citer (de manière non exhaustive) Lilya Budaghyan, Claude Carlet et Gregor Leander [46, 47], Carl Bracken, Eimear Byrne, Nadya Markin et Gary McGuire [38, 39], Yuyin Yu, Mingsheng Wang et Yongqiang Li [214] et Guobiao Weng, Yin Tan et Guang Gong [207].

Proposition 1.5.12. *Une fonction $F:\mathbb{F}_{2^n}\to\mathbb{F}_{2^n}$ est quadratique si et seulement si ses dérivées en toutes les directions non nulles sont de degré algébrique au plus 1.*

Nous avons vu à la sous-section 1.3.2 que l'image d'une fonction linéaire (resp. affine) est un sous-espace de $\mathbb{F}_{2^n}$ (resp. le translaté d'un sous-espace). Dans le cas de fonctions 2-to-1 linéaires (resp. affines), cela signifie que l'image est un sous-espace de $\mathbb{F}_{2^n}$ (resp. translaté d'un sous-espace) de dimension $n-1$ sur $\mathbb{F}_2$. En d'autres termes, il s'agit d'un hyperplan (resp. complémentaire d'un hyperplan) de $\mathbb{F}_{2^n}$.

Cela nous amène à définir les fonctions dites *crooked* (recourbées en français) que l'on doit originellement à T. Bending et Dmitry Fon-der-Flass [13] (aussi définies par Erwin van Dam et Dmitry Fon-der-Flass [199]). Cependant, leur définition des fonctions crooked ne prend en compte que le cas où n est impair et lorsque les images des dérivées sont des complémentaires d'hyperplans, c'est-à-dire qu'ils supposent implicitement que les fonctions crooked sont bijectives. La définition que nous connaissons aujourd'hui des fonctions crooked, plus générale, est due à Gohar Kyureghyan [128].

Définition 1.5.13 (Fonctions crooked)**.** Soit une fonction $F:\mathbb{F}_{2^n}\to\mathbb{F}_{2^n}$. La fonction F est appelée *crooked* si l'image de chaque dérivée de la fonction F définit un hyperplan ou le complémentaire d'un hyperplan.

Nous venons de voir que les fonctions quadratiques APN sont forcément crooked. Le problème suivant a été énoncé par T. Bending et Dmitry Fon-Der-Flass [13] ou encore par Gohar Kyureghyan [128].

Problème 1.5.14. Existe-t-il des fonctions crooked de degré algébrique ≥ 2 ?

Bien que des avancées aient été faites pour caractériser un petit peu mieux les fonctions crooked (voir [128, 16]), il semble difficile de prendre parti quant à la solution de ce problème. La meilleure borne supérieure sur le degré algébrique de ces fonctions peut être trouvée dans [71, Chapitre 5] et utilise une propriété des fonctions composantes des dérivées.

[6] Il existe aussi des monômes et des binômes quadratiques.

Proposition 1.5.15 ([71, Chapitre 5, Proposition 4.7])**.** *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ crooked. Alors le degré algébrique de la fonction F vérifie*

$$\deg(F) \leq \lfloor \frac{n}{2} \rfloor.$$

Dans [57], Anne Canteaut et María Naya-Plasencia introduisent entre autre le concept de *fonctions crooked généralisées*. L'idée de cette généralisation est de relâcher la contrainte sur le fait que l'image de chaque dérivée non nulle doit être un hyperplan ou le complémentaire d'un hyperplan. Dans la définition qu'elles donnent, l'image de chaque dérivée doit être un sous-espace de $\mathbb{F}_{2^n}$ d'une certaine dimension (la même pour toutes les dérivées).

Au Chapitre 4, nous reviendrons sur une construction de fonctions quadratiques utilisant "l'intégration", dans $\mathbb{F}_{2^n}$, de fonctions affines.

Fonctions APN exceptionnelles

Nous avons vu à la section précédente qu'il existait des polynômes de permutations exceptionnels, c'est-à-dire qu'ils conservent leur bijectivité sur une infinité d'extensions du corps de leurs coefficients. Il est naturel de définir les fonctions APN exceptionnelles.

Définition 1.5.16. Soit un polynôme $F(x) \in \mathbb{F}_{2^n}[x]$. Le polynôme F est appelé APN exceptionnel s'il représente une fonction APN sur une infinité d'extensions de $\mathbb{F}_{2^n}$.

Même si nous devons la dénomination de *fonctions APN exceptionnelles* à John Dillon [83] en 2000, l'idée d'une telle classification est plus ancienne. En effet, en 1995, Heeralal Janwa, Gary McGuire et Richard M. Wilson [112] établissent une conjecture sur la distance minimale des codes cycliques, que nous pouvons désormais traduire en termes de fonctions APN : les seules fonctions monomiales qui sont APN sur un nombre infini d'extensions sont les fonctions de Gold, $x \mapsto x^{2^k+1}$ avec $\gcd(k, n) = 1$, et de Kasami, $x \mapsto x^{2^{2k}-2^k+1}$ avec $gcd(k, n) = 1$ (voir chapitre suivant). Cette conjecture fut prouvée par Fernando Hernando et Gary McGuire [109]. Cette preuve, en addition avec le travail de Claude Carlet, Pascale Charpin et Victor Zinoviev [60] établissant une relation précise entre la théorie des codes et les fonctions APN/AB, amène Yves Aubry, Gary McGuire et François Rodier à formuler une nouvelle conjecture [9], plus générale :

Conjecture 1.5.17. Un polynôme est APN exceptionnel si et seulement si la fonction qu'il représente est CCZ-équivalente (voir sous-section 1.5.3 ou bien [60] pour la définition de CCZ-équivalence) à une fonction monomiale de Gold ou de Kasami.

Les auteurs de cette conjecture donnent en plus des premières pistes quant à la résolution de ce problème. Florian Caullery [62] et François Rodier [179] travaillent notamment à prouver cette conjecture en réduisant petit à petit le corpus des fonctions à étudier.

1.5.2. Non-linéarité

Dans cette sous-section, nous allons introduire la seconde notion de *non-linéarité* des fonctions dans $\mathbb{F}_{2^n}$. Elle peut être rapprochée de l'étude des corrélations des séquences binaires.

Transformée de Walsh d'une fonction booléenne

Définition 1.5.18 (Transformée de Walsh)**.** La *transformée de Walsh* d'une fonction booléenne à n variables f, notée W_f est donnée par :

$$\begin{array}{rccc} W_f: & \mathbb{F}_2^n & \to & \mathbb{C} \\ & u & \mapsto & \displaystyle\sum_{x\in\mathbb{F}_2^n} (-1)^{f(x)+x\cdot u}. \end{array} \tag{1.6}$$

La valeur de W_f en un point u est appelé *coefficient de Walsh* en u et le multi-ensemble[7]

$$\{W_f(u) \mid u \in \mathbb{F}_2^n\},$$

est appelé le *spectre de Walsh*.

Il apparaît que la fonction W_f est la transformée discrète de Fourier (parfois appelée transformée discrète de Hadamard) de la fonction booléenne f. Nous avons donc les propositions suivantes.

Proposition 1.5.19 (Transformée de Walsh inverse)**.** *Pour tout $x \in \mathbb{F}_2^n$ et toutes fonctions booléennes à n variables f, nous avons*

$$\sum_{u\in\mathbb{F}_2^n} W_f(u)(-1)^{u\cdot x} = 2^n(-1)^{f(x)}.$$

Proposition 1.5.20 (Relation de Parseval)**.** *Pour toute fonction booléenne f à n variables, nous avons*

$$\sum_{u\in\mathbb{F}_2^n} W_f^2(u) = 2^{2n}.$$

Le coefficient de Walsh en $u \in \mathbb{F}_2^n$ d'une fonction booléenne f à n variables vérifie l'égalité :

$$W_f(u) = 2^{n-1} - wt(f + \phi_u),$$

où ϕ_u est, rappelons-le, la fonction affine $x \mapsto u \cdot x + c$. Les coefficients de Walsh évaluent donc la distance entre la fonction booléenne f et les fonctions affines. Plus précisément, cela correspond à la corrélation entre la fonction booléenne f et la fonction booléenne linéaire $x \mapsto u \cdot x$. Moins la fonction booléenne f est corrélée à une fonction linéaire, plus on dira que sa non-linéarité est haute.

Définition 1.5.21. Soit une fonction booléenne f à n variables. La linéarité de la fonction f, notée $\mathcal{L}(f)$ est la valeur maximale de la corrélation entre f et une fonction affine :

$$\mathcal{L}(f) = \max_{u\in\mathbb{F}_2^n} |W_f(u)|.$$

La non-linéarité de f, notée $\mathcal{NL}(f)$, est la distance minimale entre f et l'ensemble des fonctions booléennes affines :

$$\mathcal{NL}(f) = \min_{u\in\mathbb{F}_2^n, \beta\in\mathbb{F}_2} wt(f + \phi_u + \beta),$$

où ϕ_u est la fonction booléenne affine définie par u, ou de manière équivalente

$$\mathcal{NL}(f) = 2^{n-1} - \frac{1}{2}\mathcal{L}(f).$$

[7] ensemble de valeurs et de leurs multiplicités.

Nous devons à O. S. Rothaus [180] une borne inférieure sur la linéarité d'une fonction booléenne.

Théorème 1.5.22 ([180])**.** *Pour toute fonction booléenne f à n variables, nous avons*

$$\mathcal{L}(f) \geq 2^{\frac{n}{2}}.$$

L'égalité n'est possible que lorsque n est pair. De plus, les fonctions booléennes dont la linéarité est minimale sont appelées bent *("courbes" en français). Les fonctions booléennes bent ne sont pas équilibrées.*

Définition 1.5.23 (Fonction plateau [51])**.** Une fonction booléenne f à n variables est appelée *plateau* si son spectre de Walsh est $\{0, \pm\mathcal{L}(f)\}$. La valeur $\mathcal{L}(f)$ est alors appelée *l'amplitude* de la fonction f. De plus, dans ce cas là, $\mathcal{L}(f) = 2^s \geq 2^{n/2}$.

Les fonctions bent sur $\mathbb{F}_{2^n}$, n pair, sont donc des fonctions plateau d'amplitude minimale, $2^{n/2}$.

Transformée de Walsh d'une fonction sur $\mathbb{F}_{2^n}$

La notion de non-linéarité pour les fonctions de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_{2^n}$ est portée par leurs fonctions composantes.

Définition 1.5.24. La linéarité d'une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, notée $\mathcal{L}(F)$, est la valeur maximale de la corrélation entre une fonction composante de F et une fonction affine :

$$\mathcal{L}(F) = \max_{\lambda \neq 0 \in \mathbb{F}_{2^n}} \mathcal{L}\left(\mathrm{Tr}(\lambda F)\right).$$

La non-linéarité de la fonction F, notée $\mathcal{NL}(F)$, est donnée par

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

Cette fois-ci, la borne inférieure sur la linéarité d'une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est due à Florent Chabaud et Serge Vaudenay [63].

Théorème 1.5.25 ([63])**.** *Pour toute fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, nous avons*

$$\mathcal{L}(F) \geq 2^{\frac{n+1}{2}}.$$

L'égalité n'est possible que lorsque n est impair. De plus, les fonctions sur $\mathbb{F}_{2^n}$ dont la linéarité est minimale sont appelées almost bent *("presque courbes" en français), ou plus brièvement AB.*

Remarque 1.5.26. Lorsque n est pair, il existe des fonctions pour lesquelles la non-linéarité vaut

$$\mathcal{L}(F) = 2^{\frac{n+2}{2}},$$

mais rien ne prouve que cette valeur soit minimale.

L'intérêt en cryptographie des fonctions AB est qu'elles offrent la meilleure résistance possible à la cryptanalyse linéaire [193, 155]. En effet, un coefficient de Walsh $W_{\mathrm{Tr}(\lambda S)}(\mu)$ correspond à une corrélation linéaire entre des entrées et des sorties d'une fonction S. Moins cette corrélation est importante, moins le biais linéaire est exploitable par un attaquant.

Définition 1.5.27 (Fonction plateau)**.** Une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est dite *plateau* si toutes ses fonctions composantes sont plateaux.

Proposition 1.5.28 ([63])**.** *Toute fonction AB est APN. Plus précisément, toute fonction de $\mathbb{F}_{2^n}$ est AB si et seulement si elle est APN et toutes ses composantes sont plateau de même amplitude.*

Pour le lecteur voulant en apprendre plus sur les liens existant entre uniformité différentielle et non-linéarité, nous conseillons la lecture de [14] de Thierry Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy. D'un point de vue plus cryptographique, Florent Chabaud et Serge Vaudenay [63] et Kaisa Nyberg et Céline Blondeau [25] se sont intéressés aux liens entre cryptanalyse différentielle et cryptanalyse linéaire.

Dans [60], Claude Carlet, Pascale Charpin et Victor Zinoviev ont aussi montré le rôle joué par les fonctions AB en théorie des codes. Le théorème suivant est le pendant du théorème 1.5.10.

Théorème 1.5.29 ([60, Théorème 5])**.** *En réutilisant les mêmes notations qu'au théorème 1.5.10, la fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est AB si et seulement si le code $C_F^{\perp}$ (i.e. le code généré par la matrice H) ne possède que des mots de poids 0, $2^{n-1} - 2^{(n-1)/2}$, 2^{n-1} et $2^{n-1} + 2^{(n-1)/2}$.*

Dans [55], Anne Canteaut, Pascale Charpin et Hans Dobbertin explicitent aussi les liens qu'il existe entre les fonctions AB et la corrélation croisée des séquences de longueur maximale.

Un autre résultat important concerne le degré algébrique des fonctions AB.

Théorème 1.5.30 ([60, 55])**.** *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ AB. Alors son degré algébrique vérifie*

$$\deg(F) \leq \frac{n+1}{2}. \tag{1.7}$$

1.5.3. Invariance et équivalences

Nous avons vu dans les sections précédentes que les fonctions de $\mathbb{F}_{2^n}$ peuvent partager certaines propriétés intéressantes (APN, AB, bijectivité,...). Dans cette section, nous allons voir des relations d'équivalence qui laissent invariantes certaines de ces propriétés.

Inverse pour la composition

La non-linéarité et l'uniformité différentielle d'une fonction bijective sur $\mathbb{F}_{2^n}$ sont conservées lorsque nous considérons son inverse pour la composition de fonctions. En effet, pour une fonction bijective $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, quels que soient $\alpha \neq 0$ et $\beta \in \mathbb{F}_{2^n}$, nous pouvons écrire l'équation $\Delta_\alpha F(x) = \beta$ comme

$$\begin{aligned} & F(x+\alpha) + F(x) = \beta \\ \Leftrightarrow \qquad & F^{-1}(y) + F^{-1}(y+\beta) = \alpha, \end{aligned}$$

avec $y = F(x)$. Ainsi nous avons $\delta_F(\alpha, \beta) = \delta_{F^{-1}}(\beta, \alpha)$ qui implique

$$\delta(F) = \delta(F^{-1}).$$

De même pour la non-linéarité, pour tout $\lambda, \mu \in \mathbb{F}_{2^n}$, nous avons

$$W_{\mathrm{Tr}(\lambda F)}(\mu) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{\mathrm{Tr}(\lambda F(x) + \mu x)} = \sum_{y \in \mathbb{F}_{2^n}} (-1)^{\mathrm{Tr}\left(\lambda y + \mu F^{-1}(y)\right)} = W_{\mathrm{Tr}(\mu F^{-1})}(\lambda),$$

avec $y = F(x)$. D'où

$$\mathcal{NL}(F) = \mathcal{NL}(F^{-1}).$$

Cependant, le degré algébrique de l'inverse, lui n'est pas le même, sauf dans le cas des fonctions linéaires, dont l'inverse, rappelons-le, est aussi linéaire.

Exemple 1.5.31. La fonction inverse de la fonction $x \mapsto x^5$ dans $\mathbb{F}_{2^9}$ est $x \mapsto x^{409}$. Or

$$wt(5) = 2 \neq 5 = wt(409).$$

De même, pour une autre fonction quadratique $x \mapsto x^9$, son inverse est $x \mapsto x^{284}$ et

$$wt(9) = 2 \neq 4 = wt(284).$$

Équivalence affine étendue

Définition 1.5.32 (EA-équivalence)**.** Deux fonctions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ sont dites *EA-équivalentes* (pour "Extended Affine" en anglais) s'il existe des fonctions $\mathbb{F}_2$-affines bijectives sur $\mathbb{F}_{2^n}$, L_0 et L_1, et une fonction $\mathbb{F}_2$-affine de $\mathbb{F}_{2^n}$ quelconque A telles que

$$G = L_0 \circ F \circ L_1 + A.$$

Si A est la fonction nulle, on dit dans ce cas que les fonctions F et G sont *affines* équivalentes.

Il n'est pas difficile de montrer que cette relation est bien une relation d'équivalence.

Remarque 1.5.33. La fonction utilisée pour la boîte-S de l'AES est une fonction affine-équivalente à la fonction inverse $x \mapsto x^{-1}$ sur $\mathbb{F}_{2^8}$.

Deux fonctions EA-équivalentes partagent la même uniformité différentielle, la même non-linéarité et le même degré algébrique. En effet, en utilisant les notations de la définition ci-dessus, et pour tout $\alpha \neq 0$, $\beta \in \mathbb{F}_{2^n}$, nous avons

$$\delta_G(\alpha, \beta) = \delta_F\left(L_1(\alpha) + L_1(0), L_0^{-1}(\beta + A(\alpha) + A(0) + L_0(0))\right),$$

ce qui implique que $\delta(G) = \delta(F)$ puisque L_0 et L_1 sont de permutations.

Remarque 1.5.34. Nous voyons une nouvelle fois que le terme constant d'une fonction n'influence en rien ses dérivées puisque la fonction $x \mapsto x + c$ est bijective quelle que soit la constante c.

De même pour la linéarité, en considérant, sans perte de généralité, que les fonctions L_0, L_1 et A sont linéaires pour simplifier les notations. Pour tout $\lambda, \mu \in \mathbb{F}_{2^n}$, nous avons

$$\begin{aligned} W_{\mathrm{Tr}(\lambda G)}(\mu) &= \sum_{x \in \mathbb{F}_{2^n}} (-1)^{\mathrm{Tr}(\lambda(L_0(F(L_1(x))) + A(x)) + \mu x)} \\ &= \sum_{y \in \mathbb{F}_{2^n}} (-1)^{\mathrm{Tr}\left(L_0^*(\lambda)F(y) + (L_1^{-1})^*(A^*(\lambda) + \mu)y\right)} \end{aligned}$$

$$= W_{\mathrm{Tr}(L_0^*(\lambda)F)}\left((L_1^{-1})^*(A^*(\lambda)+\mu)\right),$$

avec $y = L_1(x)$. Ce qui implique que $\mathcal{NL}(G) = \mathcal{NL}(F)$ puisque L_0 et L_1 sont des permutations.

Pour montrer que le degré algébrique reste inchangé par EA-équivalence, il suffit de remarquer que les fonctions monomiales $x \mapsto x^d$ et $x \mapsto x^{2d}$ ont le même degré algébrique, ou de manière équivalente que les exposants d et $2d$ ont le même poids binaire.

Équivalence CCZ

Il existe une autre équivalence, qui laisse invariantes l'uniformité différentielle et la non-linéarité d'une fonction. Elle est due à Claude Carlet, Pascale Charpin et Victor Zinoviev [60], d'où son nom de *CCZ-équivalence*.

Définition 1.5.35 (CCZ-équivalence [60])**.** Deux fonctions F et $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ sont dites *CCZ-équivalentes* si leurs graphes $\left\{(x, F(x)) \in \mathbb{F}_{2^n}^2\right\}$, et $\left\{(x, G(x)) \in \mathbb{F}_{2^n}^2\right\}$ sont affines équivalents, c'est-à-dire s'il existe une permutation affine $L = (L_0, L_1)$ de $\mathbb{F}_{2^n}^2$ telle que l'on ait

$$y = F(x) \quad \Leftrightarrow \quad L_0(x, y) = G\left(L_1(x, y)\right),$$

pour tout $(x, y) \in \mathbb{F}_{2^n}^2$.

Dans [48], Lilya Budaghyan, Claude Carlet et Alexander Pott, à qui l'on doit l'appellation de cette équivalence, ont montré que la CCZ-équivalence est plus générale que l'EA-équivalence. C'est-à-dire que deux fonctions EA-équivalentes sont forcément CCZ-équivalentes, mais que la réciproque n'est en général pas vraie.

Il est remarquable qu'une permutation soit toujours CCZ-équivalente à son inverse. On peut en déduire, que l'équivalence CCZ ne laisse pas le degré algébrique invariant. Notons toutefois que Yoshiara [213] a montré que deux fonctions quadratiques étaient CCZ-équivalentes si et seulement si elles étaient EA-équivalentes.

Une relation entre l'équivalence CCZ et la théorie des codes a été mise en évidence par K. Browning, J. F. Dillon, R. E. Kibler et M. McQuistan [44]. Deux fonctions de $\mathbb{F}_{2^n}$ sont CCZ-équivalentes si et seulement si leurs codes respectifs (définis comme au théorème 1.5.10) sont équivalents. Par codes équivalents, nous entendons qu'ils sont égaux à une permutation près des coordonnées de leur mots de codes.

Dans [142], Gregor Leander et Axel Poschmann classifient l'ensemble des boîtes-S bijectives sur quatre bits, c'est-à-dire des permutations du corps $\mathbb{F}_{2^4}$. Leur travail s'inscrit dans le cadre de la recherche de boîtes-S optimales pour concevoir des primitives en cryptographie symétrique. Ils utilisent d'abord l'EA-équivalence pour classer ces fonctions. L'EA-équivalence, rappelons-le, laisse invariants la non-linéarité, l'uniformité différentielle et le degré algébrique. Ils arrivent au résultat qu'il existe seulement seize classes de boîtes-S qui sont optimales (*i.e.* meilleure uniformité différentielle et meilleure non-linéarité).

Dans un second temps, ils classent aussi ces boîtes-S pour l'équivalence CCZ. Ils trouvent alors qu'il existe seulement quatre classes de permutations de $\mathbb{F}_{2^4}$ qui sont CCZ-inéquivalentes.

1.5.4. Structures linéaires

Définition 1.5.36 (Structure linéaire)**.** Soit $\mathbb{F}_{2^m}$ un sous-corps de $\mathbb{F}_{2^n}$ et soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^m}$. Un élément $\alpha \in \mathbb{F}_{2^n}^*$ est appelé une β-structure linéaire, avec $\beta \in \mathbb{F}_{2^m}$, si

pour tout $x \in \mathbb{F}_{2^n}$ la relation suivante est vérifiée :

$$F(x) + F(x + \alpha) = \beta.$$

Si un élément α est une β-structure linéaire d'une fonction F, nous avons donc que

$$\beta = F(0) + F(\alpha).$$

Nous remarquons aussi que les fonctions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ admettant une structure linéaire sont les fonctions ayant la plus haute uniformité différentielle possible, c'est-à-dire $\delta(F) = 2^n$. Il n'est alors pas étonnant que de telles fonctions soient généralement à proscrire[8] dans la conception de primitive en cryptographie symétrique. Nous pouvons notamment citer [96], où Jan-Hendrick Evertse utilise des structures linéaires pour monter des cryptanalyses de chiffrements par blocs basés sur le DES (le chiffrement standard avant l'AES).

Les fonctions avec des structures linéaires ont aussi d'autres intérêts. Dans [130], Gohar Kyureghyan se sert de telles fonctions pour construire de larges classes de permutations sur $\mathbb{F}_{2^n}$. Dans [72, 73], Pascale Charpin et Sumanta Sarkar construisent des fonctions booléennes bent de type Maiorana-McFarland et établissent le lien entre ses dernières fonctions et les fonctions ayant une structure linéaire.

Au Chapitre 4, nous reviendrons sur les structures linéaires des fonctions, et donnerons notamment une caractérisation originale des fonctions ayant une 0-structure linéaire sur $\mathbb{F}_{2^n}$.

[8] sauf dans certains chiffrements à flot [101].

2. Les fonctions monomiales

DANS CE CHAPITRE, nous allons étudier les fonctions monomiales sur $\mathbb{F}_{2^n}$, c'est-à-dire les fonctions du type :

$$\begin{array}{ccc} \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ x & \mapsto & x^d, \end{array} \tag{2.1}$$

où d est un entier positif, appelé *l'exposant* de la fonction monomiale. Malgré le fait que les fonctions monomiales n'utilisent que la structure multiplicative des corps finis, elles sont largement utilisées en pratique (la boîte-S de l'AES est EA-équivalente à la fonction inverse $x \mapsto x^{-1}$). Cela est notamment dû à leur facilité d'implantation, tant au niveau matériel que logiciel.

La facilité d'écriture des monômes de permutations en font des objets largement étudiés en cryptographie [166, 23, 24] en théorie des codes correcteurs d'erreurs [60, 109, 83, 55, 143] et dans l'étude des séquences [54, 100, 163]. Cette liste de références n'est bien entendu pas exhaustive. Il existe des liens entre les fonctions monomiales et les codes cycliques [150, 120, 60, 23].

Les travaux présentés ici sont le fruit d'une collaboration avec Gohar Kyureghyan et ont donné lieux à des présentations dans des conférences internationales [132, 189] ainsi qu'à une publication journal [133].

À la section 2.1, nous introduisons les définitions essentielles ainsi que certaines propriétés inhérentes aux fonctions monomiales, et à leurs exposants. Nous voyons en particulier comment calculer l'uniformité différentielle des fonctions monomiales, quelles sont les fonctions monomiales APN connues à ce jour, comment nous pouvons classer les exposants suivant les classes cyclotomiques, et les notations utilisées pour décrire leurs expressions binaires.

À la section 2.2, nous donnons des résultats concernant les inverses des exposants de Dobbertin, des exposants de Niho et des exposants de Welch. Nous calculons les expressions binaires des exposants de ces inverses, ainsi que leur poids de Hamming binaire.

Enfin, à la section 2.3, nous donnons des résultats concernant les inverses d'un exposant d fixé modulo $2^n - 1$ pour tous les entiers positifs n qui conviennent, c'est-à-dire pour tous les n tel que $\gcd(d, 2^n - 1) = 1$. De ces résultats généraux, nous déduisons des résultats concernant les expressions binaires des inverses des exposants quadratiques, des exposants de Kasami et des exposants du type $2^k - 1$.

2.1. Les monômes de permutation

L'étude des fonctions monomiales ayant certaines propriétés est un travail ayant mobilisé et mobilisant toujours un grand nombre de chercheurs en mathématiques discrètes [109, 23, 142]. Nous pouvons notamment parler de la recherche de fonctions monomiales APN. Dans les tables 2.1 et 2.2 sont recensées les valeurs des exposants des fonctions monomiales

APN, que nous appellerons les *exposants APN*, lorsque n est impair et pair respectivement. Les compléter semble être une tâche ardue.

Nom	Valeur	Références
Gold †	2^k+1 avec $\gcd(k,n)=1$, $1 \leq k \leq t$	[100, 166]
Kasami †	$2^{2k}-2^k+1$ avec $\gcd(k,n)=1$ $2 \leq k \leq t$	[120]
inverse	$2^{2t}-1$	[166, 15]
Welch †	2^t+3	[87]
Niho †	$2^t+2^{\frac{t}{2}}-1$ si t est pair $2^t+2^{\frac{3t+1}{2}}-1$ si t est impair	[86]
Dobbertin	$2^{4k}+2^{3k}+2^{2k}+2^k-1$ si $n=5k$	[89]

TABLE 2.1. : Exposants APN sur $\mathbb{F}_{2^n}$ avec $n=2t+1$.
† aussi Almost Bent (AB).

Nom	Valeur	Références
Gold	2^k+1 avec $\gcd(k,n)=1$, $1 \leq i < t$	[100, 166]
Kasami	$2^{2k}-2^k+1$ avec $\gcd(k,n)=1$ $2 \leq i < t$	[120, 113]
Dobbertin	$2^{4k}+2^{3k}+2^{2k}+2^k-1$ si $n=5k$	[89]

TABLE 2.2. : Exposants APN sur $\mathbb{F}_{2^n}$ avec $n=2t$.

Nous rappelons que les exposants de Gold et de Kasami sont les seuls exposants exceptionnels [109].

2.1.1. Propriétés

Dans cette sous-section, nous allons voir quelques propriétés que partagent les fonctions monomiales. Tout d'abord, nous avons besoin de la définition suivante.

Définition 2.1.1 (Classe Cyclotomique)**.** Soient d et n des entiers positifs. La *classe cyclotomique* (aussi appelé *coset*) de d modulo 2^n-1, notée $\mathcal{C}_d$ est définie comme suit :

$$\mathcal{C}_d = \{d, 2d, \dots 2^{n_d-1}d\},$$

où les valeurs sont calculées modulo 2^n-1 et où n_d est le plus petit entier positif tel que $d \equiv 2^{n_d}d \pmod{2^n-1}$. La valeur $\min \mathcal{C}_d$ est appelée le *représentant de la classe cyclotomique* (*"coset leader"* en anglais) de $\mathcal{C}_d$.

Exemple 2.1.2. Les classes cyclotomiques modulo 15 sont :

$$\mathcal{C}_0 = \{0\},\ \mathcal{C}_1 = \{1, 2, 4, 8\},\ \mathcal{C}_3 = \{3, 6, 12, 9\},\ \mathcal{C}_5 = \{5, 10\},\ \mathcal{C}_7 = \{7, 14, 13, 11\}.$$

$0, 1, 3, 5$ et 7 sont les représentants des classes cyclotomiques modulo 15.

Conformément à la définition 2.1.1, dire que deux exposants sont dans la même classe cyclotomique, revient à dire que les deux fonctions monomiales associées sont affines équivalentes (définition 1.5.32) entre elles par des monômes linéaires. Ainsi, puisque des propriétés cryptographiques telles que le degré algébrique, l'uniformité différentielle et la non-linéarité sont conservées, l'étude du représentant d'une classe cyclotomique est généralement suffisante pour étudier un exposant.

Nous avons déjà mentionné que grâce à la structure uniquement multiplicative des fonctions monomiales, l'étude pouvait être un petit peu plus aisée. Par exemple, le calcul de l'uniformité différentielle d'une fonction monomiale ne nécessite pas de vérifier l'image des dérivées par rapport à tous les points d'un hyperplan comme dans le cas général (cf Section 1), mais uniquement l'image de la dérivée en 1, comme le montre la propriété suivante

Propriété 2.1.3. *En utilisant les mêmes notations qu'à la définition 1.5.3, soit une fonction monomiale $F : x \mapsto x^d$ de $\mathbb{F}_{2^n}$ pour un entier positif d, alors nous avons*

$$\delta(F) = \max_{\beta \in \mathbb{F}_{2^n}} \delta_F(1, \beta).$$

Démonstration. Pour tout $\alpha, \beta \in \mathbb{F}_{2^n}$, l'équation $F(x) + F(x + \alpha) = \beta$ peut s'écrire

$$\alpha^d \left(\left(\frac{x}{\alpha} + 1 \right)^d + \left(\frac{x}{\alpha} \right)^d \right) = \beta,$$

impliquant que $\delta_F(\alpha, \beta) = \delta_F(1, \beta b/\alpha^d)$. □

Nous voyons maintenant une propriété arithmétique bien connue, qui nous sera utile dans la suite de ce chapitre.

Propriété 2.1.4. *Soient des entiers $n, k \geq 1$. Alors*

$$\gcd(2^k - 1, 2^n - 1) = 2^{\gcd(k,n)} - 1.$$

Démonstration. Lorsque $k = n$, le résultat est évident. On peut supposer sans perte de généralité que $k < n$ et faire une récurrence sur n. Notons que

$$(2^n - 1) - 2^{n-k}(2^k - 1) = 2^{n-k} - 1.$$

D'où

$$\begin{aligned} \gcd(2^k - 1, 2^n - 1) &= \gcd(2^k - 1, 2^{n-k} - 1) \\ &= 2^{\gcd(k,n-k)} - 1 \qquad \text{(par hypothèse de récurrence)} \\ &= 2^{\gcd(k,n)} - 1. \end{aligned}$$

□

Propriété 2.1.5. *Soit un entier positif d. La fonction $x \mapsto x^d$ permute le corps $\mathbb{F}_{2^n}$ si et seulement si* $\gcd(d, 2^n - 1) = 1$.

Concernant les exposants APN, nous avons déjà le résultat suivant :

Lemme 2.1.6 ([14, Proposition 3])**.** *Soit d un exposant APN. Alors,*

$$\gcd(d, 2^n - 1) = \begin{cases} 1 & \text{si } n \text{ est impair,} \\ 3 & \text{si } n \text{ est pair.} \end{cases}$$

Trouver l'inverse pour la composition d'une fonction monomiale bijective revient en fait à trouver l'inverse multiplicatif de son exposant modulo $2^n - 1$. En effet, soient d et e des entiers positifs tels que $x \mapsto x^d$ et $x \mapsto x^e$ sont des fonctions inverses (pour la composition) l'une de l'autre. Alors

$$\begin{aligned} & (x^d)^e \equiv x \pmod{x^{2^n-1}+1} \\ \Leftrightarrow \quad & de \equiv 1 \pmod{2^n - 1}. \end{aligned}$$

Dans la suite de ce document, nous désignerons le *plus petit reste positif* modulo $2^n - 1$ d'un entier d, l'entier $0 \leq e < 2^n - 1$ tel que $d^{-1} \equiv e \pmod{2^n - 1}$.

Le moyen le plus évident pour calculer l'inverse multiplicatif d'un entier modulo un autre entier est d'utiliser l'algorithme 1, connu sous le nom d'*Algorithme d'Euclide Étendu* (EEA). Originellement, cet algorithme permet de calculer le *pgcd* de deux entiers et de connaître leurs coefficients de Bézout. Il se trouve que lorsque d et $2^n - 1$ sont premiers entre eux, le coefficient de Bézout de l'entier d est égal à l'inverse de d modulo $2^n - 1$:

$$1 = (2^n - 1)q + de,$$

pour un certain entier q.

Algorithme 1 Algorithme d'Euclide Étendu (EEA)

Entrée :
des entiers a et b.
Sortie :
des entiers relatifs r, u et v tels que $r = \gcd(a, b)$ et $r = au + bv$.

Fonction EEA(a, b)
1: $r \leftarrow a$; $r' \leftarrow b$; $u \leftarrow 1$; $v \leftarrow 0$; $u' \leftarrow 0$; $v' \leftarrow 1$;
2: **Tant que** $r' \neq 0$
3: $q \leftarrow$ quotient de r/r'
4: $rs \leftarrow r$; $us \leftarrow u$; $vs \leftarrow v$;
5: $r \leftarrow r'$; $u \leftarrow u'$; $v \leftarrow v'$;
6: $r' \leftarrow rs - q \times r'$; $u' \leftarrow us - q \times u'$; $v' \leftarrow vs - q \times v'$;
7: **Retourner** r, u, v.

Cet algorithme ne permet qu'un calcul ponctuel. Il ne donne pas d'expression formelle, qui nous est utile pour pouvoir obtenir plus d'informations concernant l'inverse de d modulo $2^n - 1$. Par plus d'informations, nous entendons par exemple la représentation binaire de ces exposants, ou bien le poids de Hamming binaire qui correspond au degré algébrique de l'inverse pour la composition.

Il est important de connaître ces informations. En effet, les exposants de la table 2.1 sont inversibles modulo $2^n - 1$ (*i.e.* les fonctions monomiales associées permutent le corps $\mathbb{F}_{2^n}$), mais leurs inverses, qui sont aussi des exposants APN ne sont malheureusement pas répertoriées dans cette table. Très peu de choses sont connues sur ces exposants.

De plus, dans [163, 164], Yassir Nawaz, Guang Gong et Kishan Chand Gupta utilisent la représentation binaire des exposants APN pour en déduire une borne supérieure sur l'immunité algébrique des fonctions monomiales correspondantes.

Seuls les exposants de Gold [166] et les exposants de Niho [175] ont été le sujet d'une étude de leurs inverses.

Nous adopterons la convention suivante : le plus petit reste positif d'un entier quelconque modulo 1 vaudra 1.

2.1.2. Représentation binaire des exposants

Dans cette sous-section, nous allons introduire les notations dont nous nous servirons dans les sections suivantes pour nous permettre d'écrire la décomposition binaire des exposants et de leurs inverses.

Notation 2.1.7. Soient a et n des entiers positifs non nuls. Nous notons $[\boldsymbol{a}]_n$ la *séquence binaire* sur n bits représentant l'entier a modulo $2^n - 1$, et $\ .\ |\ .\ $ l'opérateur de *concaténation* entre deux *séquences binaires*. C'est-à-dire si

$$a \equiv \sum_{i=0}^{n-1} a_i 2^i \pmod{2^n - 1}, \qquad a_i \in \mathbb{F}_2,$$

alors

$$[\boldsymbol{a}]_n = a_{n-1}|a_{n-2}|\dots|a_1|a_0.$$

De plus, nous noterons $\overline{[\boldsymbol{a}]}_n$ le *complémentaire* binaire de la séquence $[\boldsymbol{a}]_n$. C'est-à-dire la séquence binaire représentant l'entier $2^n - 1 - (a \pmod{2^n - 1}) = 2^n - 1 - \sum_{i=0}^{n-1} a_i 2^i$.

Voyons un exemple pour mieux comprendre ces notations.

Exemple 2.1.8. Prenons $a = 13$, alors

- $[\boldsymbol{a}]_4 = 1101$;
- $[\boldsymbol{a}]_3 = 110$ car $13 \equiv 6 \pmod 7$;
- $[\boldsymbol{a}]_5 = 01101$;
- $[\boldsymbol{a}]_4 \mid [\boldsymbol{a}]_4 = 1101|1101 = 11011101 = [\mathbf{221}]_8$ car $221 = 13 \times 16 + 13$;
- $\overline{[\boldsymbol{a}]}_5 = 10010$.

Notons au passage que $[\mathbf{7}]_3 = 000$.

Nous utiliserons le poids de Hamming sur les séquences binaires de la façon suivante :

$$wt([\boldsymbol{a}]_n) = wt(a \pmod{2^n - 1}).$$

Propriété 2.1.9. *Soient des entiers $s \geq 2$ et $a \geq 0$. Alors*

$$wt([\boldsymbol{a}]_s \mid \overline{[\boldsymbol{a}]}_s) = wt([\boldsymbol{a} \times \boldsymbol{2^s} + \boldsymbol{(2^s - 1 - a)}]_{2s}) = wt([\boldsymbol{a}]_s) + wt(\overline{[\boldsymbol{a}]}_s) = s.$$

Lemme 2.1.10. *Soient des entiers $s \geq 2$ et $0 < u < 2^s$. Alors*

$$[\boldsymbol{(2^s - 1)u}]_{2s} = [\boldsymbol{u - 1}]_s \mid \overline{[\boldsymbol{u - 1}]}_s = [\boldsymbol{u - 1}]_s \mid [\boldsymbol{2^s - u}]_s.$$

De plus, $wt([\boldsymbol{(2^s - 1)u}]_{2s}) = s$.

Démonstration. L'établissement de ce lemme provient du fait que $(2^s - 1)u = (u - 1)2^s + (2^s - u)$. De plus, la représentation binaire de longueur s de $2^s - u$ est le complémentaire de celle de $u - 1$, car $(2^s - u) + (u - 1) = 2^s - 1$. □

2.2. Calcul des inverses des exposants APN

Dans cette section, nous donnons les formules explicites des inverses modulo $2^n - 1$ des exposants APN (voir table 2.1) qui dépendent de n lorsque ce dernier est impair. C'est à dire que nous donnons le *plus petit reste positif* des inverses des exposants de Dobbertin, Niho et Welch modulo $2^n - 1$ ainsi que leur représentation binaire et leur poids de Hamming binaire.

Nous ne traiterons pas les exposants de Gold et de Kasami dans cette section. Nous leurs consacrerons la section 2.3. En effet, étant donné que ces exposants décrivent des fonctions monomiales qui sont APN (et donc bijectives 2.1.6 lorsque n est impair) sur une infinité de corps, nous montrerons alors des résultats très généraux concernant les inverses d'un entier fixé modulo $2^n - 1$, pour tous les entiers positifs n qui conviennent. Les exposants de Gold et de Kasami deviendront alors des cas particuliers.

Nous ne traiterons pas non plus la fonction inverse, qui est involutive :

$$(2^n - 2) \times (2^n - 2) = (-1) \times (-1) \equiv 1 \pmod{2^n - 1},$$

avec $[\boldsymbol{2^n - 2}]_n = 111\ldots1110$ et donc $wt(2^n - 2) = n - 1$.

2.2.1. Exposants de Dobbertin

Rappelons que les exposants de Dobbertin sont de la forme suivante :

$$d = 2^{4k} + 2^{3k} + 2^{2k} + 2^k - 1, \qquad \text{avec } n = 5k. \tag{2.2}$$

Conformément au lemme 2.1.6, nous savons que ces exposants n'induisent des permutations de $\mathbb{F}_{2^n}$ que lorsque k est impair. Le théorème suivant nous donne l'expression de l'inverse d'un exposant de Dobbertin.

Théorème 2.2.1. *Soit k un entier strictement positif impair. Le* plus petit reste positif *de l'inverse de $d = 2^{4k} + 2^{3k} + 2^{2k} + 2^k - 1$ modulo $2^{5k} - 1$ est*

$$t = \frac{1}{2}\left(\frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} - 1\right). \tag{2.3}$$

Démonstration. Tout d'abord, notons que t est bien un entier positif vérifiant $t < 2^{5k} - 1$. En effet, d'après la propriété 2.1.4, $2^k - 1$ divise $2^{5k} - 1$ et comme k est impair, $3 = 2^2 - 1$ divise $2^{k+1} - 1$. De plus,

$$\frac{2^{5k} - 1}{2^k - 1} = 2^{4k} + 2^{3k} + 2^{2k} + 2^k + 1,$$

et

$$\frac{2^{k+1} - 1}{3} = 2^{k-1} + 2^{k-3} + \cdots + 1,$$

donc

$$\frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} < 2^{4k+k} = 2^{5k}.$$

Comme $\frac{2^{5k}-1}{2^k-1} \cdot \frac{2^{k+1}-1}{3}$ est impair, nous avons

$$\frac{1}{2}\left(\frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} - 1\right) < 2^{5k} - 1.$$

On remarque ensuite que $d = \frac{2^{5k}-1}{2^k-1} - 2$. Nous constatons aussi la chose suivante :

$$\begin{aligned} 2^k \cdot \frac{2^{5k} - 1}{2^k - 1} &= 2^k(2^{4k} + 2^{3k} + 2^{2k} + 2^k + 1) \\ &= (2^{5k} + 2^{4k} + 2^{3k} + 2^{2k} + 2^k) \\ &\equiv (2^{4k} + 2^{3k} + 2^{2k} + 2^k + 1) \pmod{2^{5k} - 1} \\ &\equiv \frac{2^{5k} - 1}{2^k - 1} \pmod{2^{5k} - 1}. \end{aligned}$$

Cela implique donc que

$$d \cdot \frac{2^{5k} - 1}{2^k - 1} \equiv 3 \cdot \frac{2^{5k} - 1}{2^k - 1} \pmod{2^{5k} - 1}.$$

D'où

$$\begin{aligned} t \cdot d &= \frac{1}{2}\left(\frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} - 1\right) d \\ &= \frac{1}{2}\left(d \cdot \frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} - d\right) \\ &\equiv \frac{1}{2}\left(3 \cdot \frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} - d\right) \pmod{2^{5k} - 1} \\ &\equiv \frac{1}{2}\left(\frac{2^{5k} - 1}{2^k - 1}(2^{k+1} - 1) - \frac{2^{5k} - 1}{2^k - 1} + 2\right) \pmod{2^{5k} - 1} \\ &\equiv 1 \pmod{2^{5k} - 1}. \end{aligned}$$

□

Nous pouvons écrire de manière légèrement différente l'expression (2.3) :

$$2t = \frac{2^{5k} - 1}{2^k - 1} \cdot \frac{2^{k+1} - 1}{3} - 1 = \left(\sum_{i=0}^{4} \sum_{j=0}^{\frac{k-1}{2}} 2^{(i+1)k-2j-1}\right) - 1. \tag{2.4}$$

En effet,

$$\frac{2^{5k}-1}{2^k-1} = \left(2^{4k}+2^{3k}+2^{2k}+2^k+1\right) = \sum_{i=0}^{4} 2^{ik},$$

et

$$\frac{2^{k+1}-1}{3} = \left(2^{k-1}+2^{k-3}+\cdots+2^2+1\right) = \sum_{j=0}^{\frac{k-1}{2}} 2^{k-2j-1}.$$

Exemple 2.2.2. Prenons $k=3$. Nous avons alors que $n = 5\times 3 = 15$ et

$$d = 2^{12}+2^9+2^6+2^3-1 = 4679,$$

ou bien plus visuellement, l'expression binaire de d est

$$[\boldsymbol{d}]_{15} = 001001001000111.$$

En utilisant l'équation (2.3) du théorème 2.2.1, on obtient que l'inverse de $d=4679$ modulo $2^{15}-1$ vaut

$$\begin{aligned} t &= \frac{1}{2}\left(\frac{2^{15}-1}{2^3-1}\cdot\frac{2^4-1}{3}-1\right) \\ &= \frac{1}{2}\left(4681\times 5-1\right) \\ &= 11702. \end{aligned}$$

Nous pouvons obtenir la représentation binaire de façon plus simple en utilisant la relation (2.4) :

$$[\boldsymbol{t}]_{15} = 010110110110110.$$

A partir de la relation (2.4), "plus visuelle", nous pouvons extraire le résultat suivant sur le poids de Hamming des inverses des exposants de Dobbertin.

Corollaire 2.2.3. *Soit $d = 2^{4k}+2^{3k}+2^{2k}+2^k-1$ un exposant de Dobbertin et soit t le plus petit reste positif de son inverse modulo $2^{5k}-1$. Nous avons*

$$wt(t) = \frac{n+3}{2}. \tag{2.5}$$

Démonstration. Pour démontrer ce résultat, nous allons utiliser la formule (2.4) utilisant une double somme. Nous avons donc

$$\begin{aligned} 2t &= \left(\sum_{i=0}^{4}\sum_{j=0}^{\frac{k-1}{2}} 2^{(i+1)k-2j-1}\right)-1 \\ &= (2^{4k}+2^{3k}+2^{2k}+2^k+1)(2^{k-1}+2^{k-3}+\cdots+1)-1 \\ &= 2^{5k-1}+2^{5k-3}+\cdots+2^{4k}+2^{4k-1}+2^{4k-3}+\cdots+2^{3k}+\cdots+1-1 \end{aligned}$$

impliquant

$$wt(2t) = wt(t) = 5\times\frac{k+1}{2}-1 = \frac{5k+3}{2}.$$

□

Le résultat sous-jacent du corollaire 2.2.3 est que l'inverse d'un exposant de Dobbertin définit une fonction APN de degré algébrique dépassant $\frac{n+1}{2}$. À notre connaissance, il s'agit là du seul exemple que nous ayons d'une telle fonction, avec celui de la fonction inverse qui est de degré algébrique $n-1$.

De plus, si l'on combine cette observation sur le haut degré algébrique des inverses des exposants de Dobbertin avec le résultat donné au théorème 1.5.30, nous constatons que nous donnons une nouvelle démonstration du fait que les exposants de Dobbertin ne sont pas AB. Ce résultat fut montré pour la première fois par Anne Canteaut, Pascale Charpin et Hans Dobbertin [55]. Ils utilisèrent alors les propriétés de divisibilité des codes correspondants aux exposants de Dobbertin.

2.2.2. Exposants de Niho

Les inverses des exposants de Niho furent déterminés par Marius Portmann et Marc Rennhard [175]. Il est cependant relativement difficile de se procurer le rapport dans lequel apparaissent ces résultats, c'est pourquoi nous les rappelons dans cette sous-section.

Commençons tout d'abord par rappeler la forme des exposants de Niho pour n impair. Posons $n = 2m+1$, les exposants de Niho sont les entiers

$$d = \begin{cases} 2^m + 2^{\frac{m}{2}} - 1 & \text{si } m \text{ est pair,} \\ 2^m + 2^{\frac{3m+1}{2}} - 1 & \text{si } m \text{ est impair.} \end{cases}$$

Ces exposants peuvent aussi s'écrire de manière équivalente sous la forme suivante :

$$d = \begin{cases} 2^{2k} + 2^k - 1 & \text{si } m = 2k & (n = 4k+1), \\ 2^{2k+1} + 2^{3k+2} - 1 & \text{si } m = 2k+1 & (n = 4k+3), \end{cases} \qquad \text{avec } k \geq 1.$$

Cette représentation des exposants de Niho va s'avérer plus utile par la suite. En effet, comme le montre le théorème suivant, l'expression des inverses dépend de n modulo 8.

Théorème 2.2.4 ([175, Théorèmes 12 et 14])**.** *Soit d un exposant de Niho et soit t le* plus petit reste positif *de l'inverse de d modulo $2^n - 1$.*

(a) *Soit $n = 4k+1$ et soit $d = 2^{2k} + 2^k - 1$. Alors*

- *si $n \equiv 1 \pmod 8$ (i.e. k est pair),*

$$t = \frac{2^k - 1}{3}(2^{3k+1} + 2^{k+1} + 1) + 2^k + 2^{3k+1}. \tag{2.6}$$

- *si $n \equiv 5 \pmod 8$ (i.e. k est impair),*

$$t = \frac{2^{k-1} - 1}{3}(2^{3k+2} + 2^{2k+2} + 1) + 2^{3k+1} + 2^{2k+1} + 2^{k-1}. \tag{2.7}$$

(b) *Soit $n = 4k+3$ et soit $d = 2^{3k+2} + 2^{k+2} + 2$. Alors*

- *si $n \equiv 3 \pmod 8$ (i.e. k est pair),*

$$t = \frac{2^k - 1}{3}(2^{3k+4} + 2^{k+2} + 2) + 2^{3k+3} + 2^{k+1}. \tag{2.8}$$

– *si* $n \equiv 7 \pmod 8$ *(i.e. k est impair),*

$$t = \frac{2^{k+1}-1}{3}(2^{3k+3} + 2^{2k+3} + 2) + 2^{2k+2}. \tag{2.9}$$

De plus,

$$wt(t) = \begin{cases} \frac{3n+5}{8} & si\ n \equiv 1 \pmod 8, \\ \frac{3n+9}{8} & si\ n \equiv 5 \pmod 8, \\ \frac{3n+7}{8} & si\ n \equiv 3 \pmod 8, \\ \frac{3n+11}{8} & si\ n \equiv 7 \pmod 8. \end{cases}$$

Démonstration. Puisque la démonstration est du domaine calculatoire, nous ne démontrerons qu'un seul des quatre cas exposés par ce théorème. Le lecteur averti pourra alors compléter la preuve avec des méthodes similaires à celles présentées dans ce qui suit.

Prenons $n = 4k+1$ avec k pair. Nous considérons alors l'exposant de Niho $d = 2^{2k}+2^k-1$. Pour commencer, nous avons bien que t est un entier positif tel que $t < 2^n - 1$. Ensuite nous avons

$$\begin{aligned} d \cdot t &= (2^{2k} + 2^k - 1)\left(\frac{2^k-1}{3}\left(2^{3k+1} + 2^{k+1} + 1\right) + 2^k + 2^{3k+1}\right) \\ &\equiv \frac{2^k-1}{3}\left(2^{5k+1} + 2^{4k+1} - 2^{3k+1} + 2^{3k+1} + 2^{2k+1} - 2^{k+1} + 2^{2k} + 2^k - 1\right) \\ &\quad + 2^{3k} + 2^{2k} - 2^k + 2^{5k+1} + 2^{4k+1} - 2^{3k+1} \\ &\equiv \frac{2^k-1}{3}\left(2^{2k+1} + 2^{2k}\right) - 2^{3k} + 2^{2k} + 1 \\ &\equiv (2^k-1)2^{2k} - 2^{3k} + 2^{2k} + 1 \\ &\equiv 1 \pmod{2^{4k+1}-1}. \end{aligned}$$

Notons maintenant que le développement de t est comme suit :

$$t = \frac{2^k-1}{3} + 2^k + 2^{k+1}\frac{2^k-1}{3} + 2^{3k+2} + 2^{3k+1}\left(\frac{2^k-1}{3} - 1\right),$$

et que

$$\frac{2^k-1}{3} = \sum_{i=0}^{\frac{k}{2}-1} 2^{2i}.$$

Ainsi $wt(t) = wt(\frac{2^k-1}{3})+1+wt(\frac{2^k-1}{3})+1+wt(\frac{2^k-1}{3})-1 = 1+3\frac{k}{2} = 1+3\frac{n-1}{8} = \frac{3n+5}{8}$. □

Exemple 2.2.5. Prenons un exposant de Niho pour $k = 4$. C'est à dire que nous regardons t, l'inverse de $d = 2^8 + 2^4 - 1 = 271$ modulo $2^{17} - 1$:

$$t = 5(2^{13} + 2^5 + 1) + 2^4 + 2^{13} = 49333.$$

De plus,

$$[\boldsymbol{t}]_{17} = 01100000010110101,$$

et $wt(t) = \frac{3\times17+5}{8} = 7$.

2.2.3. Exposants de Welch

Les exposants de Welch sont de la forme suivante :

$$d = 2^k + 3, \qquad \text{avec } n = 2k + 1. \tag{2.10}$$

Les exposants de Welch sont APN et inversibles modulo $2^n - 1$ pour tous les n impairs. Leurs inverses dépendent de k modulo 8, et sont décrits dans le théorème suivant.

Théorème 2.2.6. *Soit $n = 2k + 1$. Le* plus petit reste positif t *de l'inverse de l'exposant de Welch $d = 2^k + 3$ modulo $2^{2k+1} - 1$ est défini par :*

- *Si* $k \equiv 0 \pmod 8$ *alors*

$$t = 2^k + \frac{2^k - 1}{17}\left(13 \cdot 2^{k+1} + 7\right)$$

  et $wt(t) = k + 1$.

- *Si* $k \equiv 1 \pmod 8$ *alors*

$$t = 2^{k-1} + 2^k + \frac{2^{k-1} - 1}{17}\left(7 \cdot 2^{k+2} + 1\right)$$

  et $wt(t) = k + 1$.

- *Si* $k \equiv 2 \pmod 8$ *alors*

$$t = 1 + 2^{k+1} + \frac{2^{k-2} - 1}{17}\left(5 \cdot 2^{k+3} + 16\right)$$

  et $wt(t) = k$.

- *Si* $k \equiv 3 \pmod 8$ *alors*

$$t = 2^k + 2^{k+2} + 2^{k+3} + \frac{2^{k-3} - 1}{17}\left(7 \cdot 2^{k+5} + 8\right)$$

  et $wt(t) = k$.

- *Si* $k \equiv 4 \pmod 8$ *alors*

$$t = 2^{k-4} + 2^{k-2} + 2^{k-1} + 2^{k+4} + \frac{2^{k-4} - 1}{17}\left(9 \cdot 2^{k+5} + 3\right)$$

  et $wt(t) = k$.

- *Si* $k \equiv 5 \pmod 8$ *alors*

$$t = 1 + 2^{k-3} + 2^{k-1} + 2^k + 2^{k+1} + \frac{2^{k-5} - 1}{17}\left(2^{k+6} + 12\right)$$

  et $wt(t) = k$.

- *Si* $k \equiv 6 \pmod 8$ *alors*

$$t = 2^{k-5} + 2^{k-4} + 2^{k-2} + 2^{k+3} + 2^{k+4} + 2^{k+5} + 2^{k+6} + \frac{2^{k-6} - 1}{17}\left(16 \cdot 2^{k+7} + 10\right)$$

  et $wt(t) = k + 1$.

- *Si* $k \equiv 7 \pmod 8$ *alors*

$$t = 2^{k-5} + 2^{k-4} + 2^{k-3} + 2^{k-2} + 2^{k+1} + 2^{k+2} + 2^{k+4} + 2^{k+7} + \frac{2^{k-7}-1}{17}\left(10 \cdot 2^{k+8} + 4\right)$$

et $wt(t) = k + 1$.

Démonstration. Notons tout d'abord que les t listés ci-dessus sont des entiers positifs tels que $t < 2^{2k+1} - 1$. Nous montrerons que $d \cdot t \equiv 1 \pmod{2^n - 1}$ uniquement dans le cas où $k \equiv 4 \pmod 8$ puisque les autres cas sont similaires, et peuvent être traités de la même manière.

Posons donc $k \equiv 4 \pmod 8$, et considérons le produit suivant :

$$(2^k + 3) \cdot \left(2^{k-4} + 2^{k-2} + 2^{k-1} + 2^{k+4} + \frac{2^{k-4}-1}{17}\left(9 \cdot 2^{k+5} + 3\right)\right) \pmod{2^{2k+1} - 1}. \tag{2.11}$$

Nous observons que

$$(2^k + 3) \cdot \left(9 \cdot 2^{k+5} + 3\right) \equiv 17 \cdot (9 + 3 \cdot 17 \cdot 2^k) \pmod{2^{2k+1} - 1},$$

et que l'équation (2.11) peut être réduite en

$$\begin{aligned}
& (2^k + 3) \cdot \left(2^{k-4} + 2^{k-2} + 2^{k-1} + 2^{k+4}\right) + (2^{k-4} - 1) \cdot (9 + 3 \cdot 17 \cdot 2^k) \\
= \;& (2^k + 2 + 1)\left(2^{k-4} + 2^{k-2} + 2^{k-1} + 2^{k+4}\right) \\
& \quad + (2^{k-4} - 1)(2^3 + 1 + 2^{k+5} + 2^{k+4} + 2^{k+1} + 2^k) \\
\equiv \;& 2^{2k-4} + 2^{2k-2} + 2^{2k-1} + 2^3 + 2^{k-3} + 2^{k-1} + 2^k + 2^{k+5} + 2^{k-4} + 2^{k-2} + 2^{k-1} \\
& + 2^{k+4} + 2^{k-1} + 2^{k-4} + 1 + 2^{2k} + 2^{2k-3} + 2^{2k-4} \\
& \quad - 2^3 - 1 - 2^{k+5} - 2^{k+4} - 2^{k+1} - 2^k \qquad \pmod{2^{2k+1} - 1} \\
\equiv \;& 2^{2k-4} + 2^{2k-2} + 2^{2k-1} + 2^{k-3} + 2^{k-1} - 2^k + 2^{k-4} + 2^{k-2} + 2^{k-4} \\
& \quad + 2^{2k} + 2^{2k-3} + 2^{2k-4} \qquad \pmod{2^{2k+1} - 1} \\
\equiv \;& 2^{2k} + 2^{2k-1} + 2^{2k-2} + 2^{2k-3} + 2 \cdot 2^{2k-4} - 2^k \\
& \quad + 2^{k-1} + 2^{k-2} + 2^{k-3} + 2 \cdot 2^{k-4} \qquad \pmod{2^{2k+1} - 1} \\
\equiv \;& 1 \pmod{2^{2k+1} - 1}.
\end{aligned}$$

Il ne reste plus qu'à prouver que le poids de $t = 2^{k-4} + 2^{k-2} + 2^{k-1} + 2^{k+4} + \frac{2^{k-4}-1}{17}\left(9 \cdot 2^{k+5} + 3\right)$ est $wt(t) = k$ pour finir la démonstration.

Calculons tout d'abord le poids de $\frac{2^{k-4}-1}{17}\left(9 \cdot 2^{k+5} + 3\right)$.

$$\begin{aligned}
\frac{2^{k-4}-1}{17}\left(9 \cdot 2^{k+5} + 3\right) &= \frac{2^{k-4}-1}{2^8 - 1}(2^4 - 1)\left(9 \cdot 2^{k+5} + 3\right) \\
&= (2^4 - 1)\left(9 \cdot 2^{k+5} + 3\right) \sum_{j=0}^{\frac{k-4}{8}-1} 2^{8j}
\end{aligned}$$

$$= \quad 3(2^4-1)\sum_{j=0}^{\frac{k-4}{8}-1} 2^{8j} + 9(2^4-1)\sum_{l=\frac{k+4}{8}}^{\frac{k}{4}-1} 2^{8l+1}.$$

Puisque les entiers 3 et 9 sont inférieurs à 2^4, cela implique par le lemme 2.1.10 que les entiers $3(2^4-1)\sum_{j=0}^{\frac{k-4}{8}-1} 2^{8j}$ et $9(2^4-1)\sum_{l=\frac{k+4}{8}}^{\frac{k}{4}-1} 2^{8l+1}$ ont le même poids binaire, qui est :

$$wt\left(3(2^4-1)\sum_{j=0}^{\frac{k-4}{8}-1} 2^{8j}\right) = wt\left(9(2^4-1)\sum_{l=\frac{k+4}{8}}^{\frac{k}{4}-1} 2^{8l+1}\right) = 4\frac{k-4}{8} = \frac{k-4}{2}.$$

Nous en concluons que

$$wt(t) = 4 + 2\frac{k-4}{2} = k.$$

□

Exemple 2.2.7. Posons $k = 10$. Nous regardons t, l'inverse de $d = 2^{10} + 3 = 1027$ modulo $2^{21} - 1$:

$$t = 1 + 2^{11} + \frac{2^8 - 1}{17}(5 \times 2^{13} + 16) = 616689.$$

De plus,

$$[\boldsymbol{t}]_{21} = 010010110100011110001,$$

et $wt(t) = 10$.

2.3. Inversion modulo $2^n - 1$

Dans la section précédente, nous avons présenté les inverses modulo $2^n - 1$ des exposants APN qui dépendent de n. Dans cette section, nous nous intéressons à un problème plus général :

Problème 2.3.1. Soit un entier positif $d \geq 1$ fixé. Pour tout entier positif n tel que $\gcd(d, 2^n - 1) = 1$, que peut on dire des inverses de d modulo $2^n - 1$?

Nous aimerions notamment savoir quelles sont leur représentation binaire, leur poids de Hamming binaire.

2.3.1. Cas général

La plupart des résultats de cette sous-section peuvent se généraliser au cas où le modulo est $p^n - 1$, pour p un nombre premier quelconque.

Définition 2.3.2. Soit d un entier positif *fixé*, et soit le sous-ensemble des entiers naturels $\mathbb{N}_d = \{n \in \mathbb{N} \mid \gcd(d, 2^n - 1) = 1\}$. Définissons la fonction

$$\mathrm{Inv}_d \ : \ \mathbb{N}_d \to \mathbb{N}$$

dont les images sont les *plus petits restes positifs* des inverses de d modulo $2^n - 1$.

Plus précisément, pour chaque $n \in \mathbb{N}_d$, l'entier $\mathrm{Inv}_d(n)$ est tel que

$$0 < \mathrm{Inv}_d(n) < 2^n - 1 \qquad \text{et} \qquad d \cdot \mathrm{Inv}_d(n) \equiv 1 \pmod{2^n - 1}.$$

Nous passons maintenant à l'étude de cette fonction, Inv_d. Nous pouvons d'ores et déjà supposer que lors de cette étude, l'entier d sera impair. En effet, les inverses de deux entiers d'une même classe cyclotomique sont eux aussi dans une même classe cyclotomique modulo $2^n - 1$, comme le montre la propriété suivante.

Propriété 2.3.3. *Soit un entier positif d et soit $n \in \mathbb{N}_d$. Alors,*

$$\mathrm{Inv}_{2d}(n) \equiv 2^{n-1}\mathrm{Inv}_d(n) \pmod{2^n - 1}.$$

De fait, durant toute cette sous-section 2.3.1, d désignera un *entier positif impair*. La définition suivante introduit un élément essentiel à l'étude des inverses modulo $2^n - 1$.

Définition 2.3.4. Soit d un nombre entier impair positif. L'ordre de 2 modulo d, noté θ_d, est le plus petit entier positif o tel que $2^o \equiv 1 \pmod{d}$,

$$\theta_d = \min\{o \in \mathbb{N}^* \mid 2^o - 1 \equiv 0 \pmod{d}\}. \tag{2.12}$$

Nous définissons la convention : $\theta_1 = 0$.

La quantité θ_d joue un rôle crucial dans la suite de cette section sur l'étude de la fonction Inv_d. En effet, nous verrons que seulement $\lfloor \theta_d/2 \rfloor$ valeurs de la fonction Inv_d déterminent complètement cette dernière. De plus, cette valeur, θ_d, dépend uniquement de l'entier d dont nous voulons étudier les inverses modulo $2^n - 1$ pour tous les $n \in \mathbb{N}_d$.

Pour comprendre cela, nous allons tout d'abord voir dans la proposition suivante que la connaissance de $\mathrm{Inv}_d(r)$, pour un entier $1 \leq r \leq \theta_d - 1$ convenable (*i.e.* $\gcd(d, 2^r - 1) = 1$), nous donne la valeur $\mathrm{Inv}_d(\theta_d - r)$. Avant cela, nous devons montrer le lemme suivant.

Lemme 2.3.5. *Soit d un entier positif non nul impair, et soit un entier $1 \leq r \leq \theta_d - 1$. Alors,*

$$\gcd(d, 2^r - 1) \quad = \quad \gcd(d, 2^{\theta_d - r} - 1).$$

Entre autre,

$$r \in \mathbb{N}_d \quad \Leftrightarrow \quad \theta_d - r \in \mathbb{N}_d.$$

Démonstration. Nous avons que d divise $2^{\theta_d} - 1$ d'après la définition 2.3.4, d'où

$$\begin{aligned}\gcd(d, 2^{\theta_d - r} - 1) &= \gcd(d, 2^r(2^{\theta_d - r} - 1)) = \gcd(d, 2^r(2^{\theta_d - r} - 1) - (2^{\theta_d} - 1)) \\ &= \gcd(d, -(2^r - 1)) = \gcd(d, 2^r - 1).\end{aligned}$$

□

Proposition 2.3.6. *Soit $1 \leq r \leq \theta_d - 1$ tel que $r \in \mathbb{N}_d$. Alors,*

$$\mathrm{Inv}_d(\theta_d - r) = \frac{\left(d + 1 - \frac{\mathrm{Inv}_d(r) \cdot d - 1}{2^r - 1}\right)(2^{\theta_d - r} - 1) + 1}{d}. \tag{2.13}$$

Démonstration. Posons t le nombre rationnel représentant la partie droite de l'équation (2.13), c'est-à-dire

$$t = \frac{\left(d + 1 - \frac{\mathrm{Inv}_d(r) \cdot d - 1}{2^r - 1}\right)(2^{\theta_d - r} - 1) + 1}{d}.$$

Premièrement, nous remarquons que

$$\begin{aligned}&(2^r-1)\left(\left(1-\frac{\mathrm{Inv}_d(r)\cdot d-1}{2^r-1}\right)(2^{\theta_d-r}-1)+1\right) \qquad (2.14)\\ &\equiv (2^r-1)\left(2^{\theta_d-r}-1-\frac{(\mathrm{Inv}_d(r)\cdot d-1)(2^{\theta_d-r}-1)}{2^r-1}+1\right)\\ &\equiv (2^r-1)2^{\theta_d-r}-(\mathrm{Inv}_d(r)\cdot d-1)(2^{\theta_d-r}-1)\equiv 0 \pmod d.\end{aligned}$$

La partie gauche de cette congruence (2.14) est

$$2^{\theta_d}-1-\mathrm{Inv}_d(r)\cdot d\cdot\left(2^{\theta_d-r}-1\right),$$

qui est divisible par d, puisque $2^{\theta_d}-1\equiv 0 \pmod d$ par la définition 2.3.4. Comme $\gcd(d,2^r-1)=1$, *i.e.* $r\in\mathbb{N}_d$, nous venons de montrer que le numérateur de t est divisible par d, impliquant que t est un entier.

Finalement, nous pouvons vérifier aisément que $t\cdot d\equiv 1 \pmod{2^{\theta_d-r}-1}$ ainsi que $1\le t\le 2^{\theta_d-r}-1$. En effet, l'entier t est borné par

$$t<\frac{(d+1-1)(2^{\theta_d-r}-1)+1}{d}=2^{\theta_d-r}-1+\frac{1}{d}.$$

□

De cette proposition, nous pouvons déduire l'équivalence suivante, qui est obtenue directement de l'équation (2.13).

Corollaire 2.3.7. *Soit $1\le r\le\theta_d-1$ tel que $r\in\mathbb{N}_d$. Alors*

$$\frac{d\cdot\mathrm{Inv}_d(\theta_d-r)-1}{2^{\theta_d-r}-1}+\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1}=d+1. \qquad (2.15)$$

Le théorème suivant est un résultat montrant comment la valeur de $\mathrm{Inv}_d(n)$ peut être calculée à l'aide de $\mathrm{Inv}_d(r)$ où r est le plus petit reste positif de n modulo θ_d.

Ce théorème comporte plusieurs expressions de la même dépendance (entre $\mathrm{Inv}_d(n)$ et $\mathrm{Inv}_d(r)$). Nous verrons dans la suite de cette section, que chacune d'entre elles s'avérera plus utile/pratique à manipuler selon la situation.

Théorème 2.3.8. *Soit un entier positif $n\in\mathbb{N}_d$ et soit $1\le r\le\theta_d-1$ tel que $n\equiv r \pmod{\theta_d}$. Alors*

(a)

$$\mathrm{Inv}_d(n)=\mathrm{Inv}_d(r)\cdot 2^{n-r}+\left(\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1}-1\right)\cdot\frac{2^{n-r}-1}{d}. \qquad (2.16)$$

(b)

$$\mathrm{Inv}_d(n)=\mathrm{Inv}_d(r)\cdot\sum_{i=0}^{m}2^{\theta_d\cdot i}+\left(2^{\theta_d-r}-1-\mathrm{Inv}_d(\theta_d-r)\right)2^r\sum_{i=0}^{m-1}2^{\theta_d\cdot i}, \qquad (2.17)$$

où $m=\frac{n-r}{\theta_d}$. De manière équivalente,

$$\mathrm{Inv}_d(n) \;=\; \mathrm{Inv}_d(r)\cdot\frac{2^{\theta_d\cdot(m+1)}-1}{2^{\theta_d}-1}+\left(2^{\theta_d-r}-1-\mathrm{Inv}_d(\theta_d-r)\right)2^r\cdot\frac{2^{\theta_d\cdot m}-1}{2^{\theta_d}-1}.$$

(c)

$$\mathrm{Inv}_d(n) = \frac{\mathrm{Inv}_d(r)(2^n-1) - \frac{2^n-2^r}{d}}{2^r-1}. \tag{2.18}$$

Démonstration. **(a)** Posons

$$t = \mathrm{Inv}_d(r)\cdot 2^{n-r} + \left(\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} - 1\right)\cdot\frac{2^{n-r}-1}{d}.$$

Puisque $d\cdot\mathrm{Inv}_d(r)-1 \equiv 0 \pmod{2^r-1}$ et que $2^{n-r}-1 \equiv 2^{\theta_d}-1 \equiv 0 \pmod{d}$, il est clair que t est un entier positif. Nous montrons ensuite que $t < 2^n-1$. Notons d'abord que $\mathrm{Inv}_d(r) < 2^r-2$, car si $\mathrm{Inv}_d(r) = 2^r-2$ alors l'exposant $d = 2^r-2$ définit la fonction inverse (modulo 2^r-1) qui est involutive, et cela contredirait l'hypothèse initiale que d est impair. D'où

$$\begin{aligned}
& & \mathrm{Inv}_d(r) &< 2^r-2\\
\Rightarrow & & d\cdot\mathrm{Inv}_d(r) &< d\cdot(2^r-2)\\
\Rightarrow & & d\cdot\mathrm{Inv}_d(r)-1 &< d\cdot(2^r-1)-(d+1)\\
\Rightarrow & & \tfrac{(d\cdot\mathrm{Inv}_d(r)-1)(2^n-1)}{2^r-1} &< (2^n-1)\cdot d - \tfrac{(d+1)(2^n-1)}{2^r-1}\\
\Rightarrow & & \tfrac{(d\cdot\mathrm{Inv}_d(r)-1)(2^n-1)}{2^r-1}+1 &< (2^n-1)\cdot d - \tfrac{(d+1)(2^n-1)}{2^r-1}+1\\
\Rightarrow & & \tfrac{(d\cdot\mathrm{Inv}_d(r)-1)(2^n-1)}{2^r-1}+1 &< (2^n-1)\cdot d - (d+1)+1\\
\Rightarrow & (2^n-2^r)\cdot\tfrac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} + (d\cdot\mathrm{Inv}_d(r)-1)+1 &< (2^n-1)\cdot d.
\end{aligned}$$

Nous observons que la partie gauche de la dernière inégalité vaut

$$\begin{aligned}
& (2^n-2^r)\cdot\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} + (d\cdot\mathrm{Inv}_d(r)-1)+1\\
= \;& \left(2^{n-r}-1\right)\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} + 2^{n-r}(d\cdot\mathrm{Inv}_d(r)-1)+1\\
= \;& 2^{n-r}\cdot d\cdot\mathrm{Inv}_d(r) + \left(2^{n-r}-1\right)\left(\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1}-1\right)\\
= \;& t\cdot d,
\end{aligned}$$

montrant que nous avons bien $t < 2^n-1$.

Pour finir cette preuve, nous devons montrer que t est bien l'inverse de d modulo 2^n-1. Nous le déduisons de l'égalité ci-dessus :

$$\begin{aligned}
t\cdot d &= (2^n-2^r)\cdot\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} + (d\cdot\mathrm{Inv}_d(r)-1)+1\\
&= (2^n-1)\cdot\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} - (2^r-1)\cdot\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1} + (d\cdot\mathrm{Inv}_d(r)-1)+1\\
&\equiv 1 \pmod{2^n-1}.
\end{aligned}$$

(b) Pour montrer cette forme, nous posons la notation suivante :

$$\mathrm{S}_d(n) = \frac{d\cdot\mathrm{Inv}_d(n)-1}{2^n-1}.$$

Le corollaire 2.3.7 peut alors être résumé à

$$\mathrm{S}_d(r) + \mathrm{S}_d(\theta_d - r) = d + 1. \tag{2.19}$$

En multipliant l'équation (2.17) par d, nous obtenons

$$\begin{aligned}
&d\cdot\left(\mathrm{Inv}_d(r)\sum_{i=0}^{m}2^{\theta_d\cdot i}+\left(2^{\theta_d-r}-1-\mathrm{Inv}_d(\theta_d-r)\right)2^r\cdot\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\right)\\
&\quad=(\mathrm{S}_d(r)(2^r-1)+1)\sum_{i=0}^{m}2^{\theta_d\cdot i}\\
&\qquad+2^r d(2^{\theta_d-r}-1)\sum_{i=0}^{m-1}2^{\theta_d\cdot i}-2^r\left(\mathrm{S}_d(\theta_d-r)(2^{\theta_d-r}-1)+1\right)\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\\
&\quad=\mathrm{S}_d(r)\left(2^r\left(2^{n-r}+\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\right)-\left(1+2^{\theta_d}\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\right)\right)+\sum_{i=0}^{m}2^{\theta_d\cdot i}\\
&\qquad+\left(2^r(2^{\theta_d-r}-1)(d-\mathrm{S}_d(\theta_d-r))-2^r\right)\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\\
&\quad=\mathrm{S}_d(r)\left((2^n-1)+(2^r-2^{\theta_d})\cdot\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\right)+\left(1+2^{\theta_d}\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\right)\\
&\qquad+\left(2^r(2^{\theta_d-r}-1)(d-\mathrm{S}_d(\theta_d-r))-2^r\right)\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\\
&\quad=\mathrm{S}_d(r)(2^n-1)+1+2^{\theta_d}\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\\
&\qquad+\left(2^r(2^{\theta_d-r}-1)(d-\mathrm{S}_d(\theta_d-r)-\mathrm{S}_d(r))-2^r\right)\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\\
&\quad=1+\mathrm{S}_d(r)(2^n-1)+2^r(2^{\theta_d-r}-1)(d-\mathrm{S}_d(\theta_d-r)-\mathrm{S}_d(r)+1)\sum_{i=0}^{m-1}2^{\theta_d\cdot i}\\
&\quad\equiv 1\pmod{2^n-1},
\end{aligned}$$

Nous pouvons appliquer la relation (2.19) ici, pour obtenir la congruence désirée modulo $2^n - 1$.

(c) Cette égalité provient de l'équation (2.16) de la partie **(a)** de ce théorème.

$$\begin{aligned}
\mathrm{Inv}_d(n) &= \mathrm{Inv}_d(r)\cdot 2^{n-r}+\left(\frac{d\cdot\mathrm{Inv}_d(r)-1}{2^r-1}-1\right)\cdot\frac{2^{n-r}-1}{d}\\
&=\frac{d\cdot\mathrm{Inv}_d(r)\cdot(2^n-2^{n-r})+(d\cdot\mathrm{Inv}_d(r)-1)(2^{n-r}-1)-(2^{n-r}-1)(2^r-1)}{d\cdot(2^r-1)}\\
&=\frac{d\cdot\mathrm{Inv}_d(r)(2^n-1)-(2^{n-r}-1)-(2^{n-r}-1)(2^r-1)}{d\cdot(2^r-1)}\\
&=\frac{\mathrm{Inv}_d(r)(2^n-1)-\frac{2^n-2^r}{d}}{2^r-1}.
\end{aligned}$$

□

Nous remarquons que l'équation (2.17) permet de mieux décrire et visualiser la représentation binaire des inverses de d, Inv_d, comme le montre le corollaire suivant.

Corollaire 2.3.9. *Soient $n > 1$ et $1 \leq r \leq \theta_d - 1$ tels que $\gcd(d, 2^n - 1) = 1$ et $n \equiv r \pmod{\theta_d}$. Posons*

- $\boldsymbol{t} = \mathrm{Inv}_d(n)$;
- $\boldsymbol{a} = \mathrm{Inv}_d(r)$;
- $\boldsymbol{b} = \mathrm{Inv}_d(\theta_d - r)$;
- $\boldsymbol{u} = \left(\dfrac{d \times \boldsymbol{a} - 1}{2^r - 1} - 1\right) \dfrac{2^{\theta_d} - 1}{d} = \boldsymbol{a} + 2^r(2^{\theta_d - r} - 1 - \boldsymbol{b})$.

Alors $[\boldsymbol{t}]_n$ est obtenu en concaténant $[\boldsymbol{a}]_r$, $\overline{[\boldsymbol{b}]}_{\theta_d - r}$ et $[\boldsymbol{u}]_{\theta_d} = \overline{[\boldsymbol{b}]}_{\theta_d - r} \mid [\boldsymbol{a}]_r$ comme suit :

$$[\boldsymbol{t}]_n \;=\; [\boldsymbol{a}]_r \mid \underbrace{[\boldsymbol{u}]_{\theta_d} \mid [\boldsymbol{u}]_{\theta_d} \mid \cdots \mid [\boldsymbol{u}]_{\theta_d}}_{m} \;=\; [\boldsymbol{a}]_r \mid \overline{[\boldsymbol{b}]}_{\theta_d - r} \mid [\boldsymbol{a}]_r \mid \cdots \mid \overline{[\boldsymbol{b}]}_{\theta_d - r} \mid [\boldsymbol{a}]_r,$$

où $m = \dfrac{n - r}{\theta_d}$.

Démonstration. Nous montrons que

$$\left(\frac{d \times \mathrm{Inv}_d(r) - 1}{2^r - 1} - 1\right) \frac{2^{\theta_d} - 1}{d} = \mathrm{Inv}_d(r) + 2^r(2^{\theta_d - r} - 1 - \mathrm{Inv}_d(\theta_d - r)).$$

Pour cela, nous utilisons tout d'abord l'égalité (2.13) :

$$\begin{aligned}
&\mathrm{Inv}_d(r) + 2^r(2^{\theta_d - r} - 1 - \mathrm{Inv}_d(\theta_d - r)) \\
&= \mathrm{Inv}_d(r) + 2^r \left(2^{\theta_d - r} - 1 - \frac{\left(d + 1 - \frac{\mathrm{Inv}_d(r) \cdot d - 1}{2^r - 1}\right)(2^{\theta_d - r} - 1) + 1}{d}\right) \\
&= \frac{d\mathrm{Inv}_d(r) - \left(1 - \frac{\mathrm{Inv}_d(r) \cdot d - 1}{2^r - 1}\right)(2^{\theta_d} - 2^r) - 2^r}{d} \\
&= \frac{d\mathrm{Inv}_d(r)(2^{\theta_d} - 1) - 2^{\theta_d}(2^r - 1) - (2^{\theta_d} - 2^r)}{d(2^r - 1)} \\
&= \frac{d\mathrm{Inv}_d(r)(2^{\theta_d} - 1) - 2^r(2^{\theta_d} - 1)}{d(2^r - 1)} \\
&= \frac{(d\mathrm{Inv}_d(r) - 2^r)(2^{\theta_d} - 1)}{d(2^r - 1)} \\
&= \left(\frac{d \times \mathrm{Inv}_d(r) - 1}{2^r - 1} - 1\right) \frac{2^{\theta_d} - 1}{d}.
\end{aligned}$$

□

Nous pouvons maintenant définir la classe d'équivalence des inverses d'un entier fixé.

Définition 2.3.10 (Classe d'équivalence des inverses d'un entier fixé)**.** Soit d un entier positif fixé (impair) et soient $n, n' \in \mathbb{N}_d$. Nous définissons la relation d'équivalence $\sim_{\mathbb{N}_d}$ entre les entiers $\mathrm{Inv}_d(n)$ et $\mathrm{Inv}_d(n')$ comme :

$$\mathrm{Inv}_d(n) \sim_{\mathbb{N}_d} \mathrm{Inv}_d(n') \qquad \Leftrightarrow \qquad n \equiv n' \pmod{\theta_d}.$$

Il n'est pas difficile de vérifier que la relation $\sim_{\mathbb{N}_d}$ est une relation d'équivalence. Le corollaire précédent permet de se rendre compte de l'importance des *représentants* des classes d'équivalence définies par la relation $\sim_{\mathbb{N}_d}$ pour définir complètement la fonction Inv_d sur $\mathbb{N}_d$. En d'autres termes, la connaissance des valeurs $\mathrm{Inv}_d(r)$ pour $1 \leq r < \theta_d$ tels que $r \in \mathbb{N}_d$ permet de déterminer toutes les valeurs $\mathrm{Inv}_d(n)$ quel que soit $n \in \mathbb{N}_d$. Prenons $n \in \mathbb{N}_d$ et $r \equiv n \pmod{\theta_d}$. En utilisant les notations du corollaire 2.3.9, et en posant $\mathbf{t}' = \mathrm{Inv}_d(n + \theta_d)$, nous avons

$$[\boldsymbol{t}]_n \;=\; [\boldsymbol{a}]_r \mid \underbrace{[\boldsymbol{u}]_{\theta_d} \mid [\boldsymbol{u}]_{\theta_d} \mid \cdots \mid [\boldsymbol{u}]_{\theta_d}}_{m},$$

et comme $\mathrm{Inv}_d(n) \sim_{\mathbb{N}_d} \mathrm{Inv}_d(n + \theta_d)$:

$$[\boldsymbol{t'}]_{n+\theta_d} \;=\; [\boldsymbol{a}]_r \mid \underbrace{[\boldsymbol{u}]_{\theta_d} \mid [\boldsymbol{u}]_{\theta_d} \mid \cdots \mid [\boldsymbol{u}]_{\theta_d}}_{m+1} \;=\; [\boldsymbol{t}]_n \mid \underbrace{\overline{[\boldsymbol{b}]}_{\theta_d - r} \mid [\boldsymbol{a}]_r}_{=[\boldsymbol{u}]_{\theta_d}}.$$

De plus l'égalité (2.13) nous permet de trouver $\boldsymbol{b} = \mathrm{Inv}_d(\theta_d - r)$ à partir de $\boldsymbol{a} = \mathrm{Inv}_d(r)$. En conclusion, au plus $\lfloor \theta_d / 2 \rfloor$ valeurs de la fonction Inv_d suffisent pour entièrement la déterminer.

Exemple 2.3.11. Prenons $d = 13$, nous avons alors que $\theta_{13} = 12$. Nous avons aussi que $\mathrm{Inv}_{13}(3) = 6$, donc $\mathrm{Inv}_{13}(12 - 3) = 118$ en utilisant la relation (2.13). Les expressions binaires sont les suivantes :

$$[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(3)}]_3 = [\mathbf{6}]_3 = 110 \qquad \text{et} \qquad [\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(9)}]_9 = [\mathbf{118}]_9 = 001110110.$$

Les expressions binaires des entiers dans la classe d'équivalence de $\mathrm{Inv}_{13}(3)$ sont donc :

$$\begin{aligned}
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(3)}]_3 &= 110 \\
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(3+12)}]_{15} &= 110 \mid 110001001 \mid 110 \\
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(3+2\times 12)}]_{27} &= 110 \mid 110001001 \mid 110 \mid 110001001 \mid 110 \\
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(3+3\times 12)}]_{39} &= 110 \mid 110001001 \mid 110 \mid 110001001 \mid 110 \mid 110001001 \mid 110 \\
&\vdots
\end{aligned}$$

Par la même occasion, nous avons les expressions binaires des entiers dans la classe d'équivalence de $\mathrm{Inv}_{13}(9)$:

$$\begin{aligned}
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(9)}]_9 &= 001110110 \\
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(9+12)}]_{21} &= 001110110 \mid 001 \mid 001110110 \\
[\mathrm{Inv}_{\mathbf{13}}\,\mathbf{(9+2\times 12)}]_{33} &= 001110110 \mid 001 \mid 001110110 \mid 001 \mid 001110110 \\
&\vdots
\end{aligned}$$

Nous voyons bien qu'avec la détermination de $\mathrm{Inv}_{13}(3)$, nous pouvons entièrement déterminer deux classes d'équivalence des inverses de $d = 13$.

Nous avons aussi le résultat suivant concernant le poids de Hamming binaire des valeurs de Inv_d.

Lemme 2.3.12. *Soit $n \geq 1$ tel que $\gcd(d, 2^n - 1) = 1$. Supposons que θ_d est pair tel que d divise $2^{\theta_d/2} + 1$ et soit $1 \leq r \leq \theta_d - 1$ tel que $r \equiv n \pmod{\theta_d}$. Alors les propriétés suivantes se vérifient :*

(a)
$$wt(\mathrm{Inv}_d(n)) = wt(\mathrm{Inv}_d(r)) + \frac{n-r}{2}.$$

(b)
$$wt(\mathrm{Inv}_d(\theta_d - r)) = wt(\mathrm{Inv}_d(r)) + \frac{\theta_d}{2} - r. \tag{2.20}$$

Démonstration. Par le corollaire 2.3.9, le poids binaire de $\mathrm{Inv}_d(n)$ est

$$wt(\mathrm{Inv}_d(n)) = wt(\mathrm{Inv}_d(r)) + \frac{n-r}{\theta_d} wt(u),$$

où

$$u = \left(\frac{d \cdot \mathrm{Inv}_d(r) - 1}{2^r - 1} - 1\right) \cdot \frac{2^{\theta_d} - 1}{d} = \left(\frac{d \cdot \mathrm{Inv}_d(r) - 1}{2^r - 1} - 1\right) \cdot \frac{2^{\theta_d/2} + 1}{d} \cdot (2^{\theta_d/2} - 1).$$

Par le lemme 2.1.10 et comme

$$u = \mathrm{Inv}_d(r) + 2^r(2^{\theta_d - r} - 1 - \mathrm{Inv}_d(\theta_d - r)) < 2^{\theta_d} - 1,$$

le poids binaire de u est $\theta_d/2$, ce qui complète la preuve de **(a)**. L'établissement de la relation **(b)** découle directement du fait que

$$[\boldsymbol{u}]_{\theta_d} = \overline{[\boldsymbol{b}]}_{\theta_d - r} \mid [\boldsymbol{a}]_r$$

et donc que

$$\frac{\theta_d}{2} = wt([\boldsymbol{u}]_{\theta_d}) = \theta_d - r - wt(\mathrm{Inv}_d(\theta_d - r)) + wt(\mathrm{Inv}_d(r)).$$

□

Exemple 2.3.13. Prenons $d = 7$. Nous avons donc que $\theta_7 = 3$ et quel que soit n un entier positif non divisible par 3, $r = 1, 2 \equiv n \pmod 3$. Nous savons que $\mathrm{Inv}_7(1) = \mathrm{Inv}_7(2) = 1$, donc d'après le théorème 2.3.8, nous en déduisons

$$\mathrm{Inv}_7(n) = 2^{n-r} + \left(\frac{6}{2^r - 1} - 1\right) \frac{2^{n-r} - 1}{7}.$$

Supposons que $r = 1$, alors la représentation binaire de $\mathrm{Inv}_7(n)$ est :

$$[\mathrm{Inv}_{\mathbf{7}}(\boldsymbol{n})]_n = 1 \mid [\boldsymbol{u}]_3 \mid [\boldsymbol{u}]_3 \mid \cdots \mid [\boldsymbol{u}]_3 = 1 \mid \underbrace{10 \mid 1}_{=\boldsymbol{u}} \mid \underbrace{10 \mid 1}_{=\boldsymbol{u}} \mid \cdots \mid \underbrace{10 \mid 1}_{=\boldsymbol{u}},$$

où $\boldsymbol{u} = \left(\frac{7 \times 1 - 1}{2^1 - 1} - 1\right) \frac{2^3 - 1}{7} = 5$. De plus

$$wt(\mathrm{Inv}_7(n)) = wt(\mathrm{Inv}_7(1)) + \frac{n-1}{3} wt(\boldsymbol{u}) = 1 + \frac{n-1}{3} 2 = \frac{2n+1}{3}.$$

Supposons maintenant que $r = 2$, alors la représentation binaire de $\mathrm{Inv}_7(n)$ est :

$$[\mathrm{Inv}_{\mathbf{7}}(\boldsymbol{n})]_n = 01 \mid [\boldsymbol{u}]_3 \mid [\boldsymbol{u}]_3 \mid \cdots \mid [\boldsymbol{u}]_3 = 01 \mid \underbrace{0 \mid 01}_{=\boldsymbol{u}} \mid \underbrace{0 \mid 01}_{=\boldsymbol{u}} \mid \cdots \mid \underbrace{0 \mid 01}_{=\boldsymbol{u}},$$

où $\boldsymbol{u} = \left(\frac{7 \times 1 - 1}{2^2 - 1} - 1\right) \frac{2^3 - 1}{7} = 1$. De plus,

$$wt(\mathrm{Inv}_7(n)) = wt(\mathrm{Inv}_7(2)) + \frac{n-2}{3} wt(\boldsymbol{u}) = 1 + \frac{n-2}{3} = \frac{n+1}{3}.$$

Nous pouvons déduire du théorème 2.3.8 et de la proposition 2.3.6 une méthode récursive pour calculer Inv_d. En effet, pour certaines classes d'entiers, le calcul de $\mathrm{Inv}_d(n)$ pour un entier d fixé et un certain entier positif n peut être réduit à celui de $\mathrm{Inv}_d(n')$, avec $n' < n$. De plus, dans certains cas la réduction pourra continuer puisqu'il est envisageable que $d \equiv d' \pmod{2^{n'} - 1}$ avec $d' < d$. Cette méthode nous permet d'écrire l'algorithme suivant (Algorithme 2). Cependant, dans certains cas, la réduction pourrait ne plus être possible (ni sur n, ni sur d). Dans cette situation, nous aurons recours à un algorithme "classique" (Algorithme 1, par exemple) pour calculer l'inverse d'un entier modulo $2^n - 1$. Nous appellerons cet algorithme "classique" : `Inverse`.

Le caractère récursif de cet algorithme est dû au fait que le calcul de $\mathrm{Inv}_d(n)$, aux étapes (8), (11) et (16), est effectué par ce même algorithme. Notons qu'à l'étape (9), la formule (2.18) du théorème 2.3.8 aurait pu être utilisée indifféremment à la place de la formule (2.16). Seule la formule (2.17) serait moins efficace, puisque faisant intervenir à la fois $\mathrm{Inv}_d(r)$ et $\mathrm{Inv}_d(\theta_d - r)$.

L'algorithme 2 réduit en fait le calcul de l'inverse de d modulo $2^n - 1$, à celui de d modulo $2^r - 1$, avec $r \equiv d \pmod{\theta_d}$, $1 \leq r \leq \theta_d$.

Cet algorithme est particulièrement efficace quand il s'agit d'entiers d avec θ_d beaucoup plus petit que n ou bien encore lorsque d comporte de petits facteurs.

Le principal avantage de cet algorithme par rapport à l'algorithme d'Euclide étendu est qu'il permet de mieux comprendre la structure binaire. Cependant, en pratique, le calcul de l'ordre de 2 modulo un entier fixé, θ_d, peut être très coûteux.

Nous allons voir dans les sous-sections suivantes, que les observations faites dans cette sous-section ainsi que l'algorithme 2 peuvent être adaptés plus spécifiquement à certaines classes d'entiers, tels que les exposants quadratiques, de Kasami, ou bien encore les entiers du type $2^k - 1$. Dans ces cas précis, la valeur θ_d peut être déterminée simplement.

Tout d'abord, voyons des exécutions de cet algorithme avec les deux exemples simples suivants.

Exemple 2.3.14. Prenons $n = 97$ et $d = 2^{11} + 1 = 2049$. Nous allons calculer l'inverse de 2049 modulo $2^{97} - 1$:

Algorithme 2 Algorithme récursif d'inversion dans $\mathbb{Z}_{2^n-1}$

Entrée :
des entiers positifs d et n tels que $\gcd(d, 2^n - 1) = 1$.
Sortie :
$\mathrm{Inv}_d(n)$, l'inverse de d modulo $2^n - 1$.

Fonction $\mathrm{Inv}_d(n)$

1: **Si** $n = 1$ ou $d = 1$
2: **Retourner** 1
3: **Si** d est pair
4: **Retourner** $2^{n-1}\mathrm{Inv}_{d/2}(n) \pmod{2^n - 1}$
5: $\theta_d \leftarrow$ l'ordre de 2 modulo d
6: $r \leftarrow n \pmod{\theta_d}$
7: **Si** $r \neq n$
8: $A \leftarrow \mathrm{Inv}_d(r)$
9: **Retourner** $A \cdot 2^{n-r} + \left(\dfrac{d \cdot A - 1}{2^r - 1} - 1\right) \cdot \dfrac{2^{n-r} - 1}{d}$
10: **Sinon**
11: $d' \leftarrow d \pmod{2^n - 1}$
12: **Si** $d' \neq d$
13: **Retourner** $\mathrm{Inv}_{d'}(n)$
14: **Sinon**
15: **Si** $n > \theta_d/2$
16: $B \leftarrow \mathrm{Inv}_d(\theta_d - n)$
17: **Retourner** $\dfrac{\left(d + 1 - \frac{d \cdot B - 1}{2^{\theta_d - n} - 1}\right)(2^n - 1) + 1}{d}$ ▷ $(= \mathrm{Inv}_d(n))$
18: **Sinon**
19: Calculer $\mathrm{Inv}_d(n)$ en utilisant `Inverse`

1 : $\theta_{2049} \leftarrow 22$

2 : $r \leftarrow 9 \equiv 97 \pmod{22}$

3 : $9 \neq 97$ alors retourner $\mathrm{Inv}_{2049}(9) \cdot 2^{88} + \left(\dfrac{2049 \cdot \mathrm{Inv}_{2049}(9) - 1}{511} - 1\right) \cdot \dfrac{2^{88} - 1}{2049}$

3.1 : maintenant $n = 9$ et $d = 2049$

3.2 : $\theta_{2049} \leftarrow 22$

3.3 : $r \leftarrow 9 \pmod{22}$ alors

3.4 : $d' \leftarrow 5 \equiv 2049 \pmod{2^9 - 1}$

3.5 : retourner $\mathrm{Inv}_5(9)$

3.5.1 : maintenant $n = 9$ et $d = 5$

3.5.2 : $\theta_5 \leftarrow 4$

3.5.3 : $r \leftarrow 1 \equiv 9 \pmod{4}$

3.5.4 : $1 \neq 9$ alors retourner $2^8 + 3 \cdot \dfrac{2^8 - 1}{5}$

4 : $\mathrm{Inv}_{2049}(97) = \left(2^8 + 3 \cdot \dfrac{2^8 - 1}{5}\right) \cdot 2^{88} + \left(\dfrac{2049 \cdot \left(2^8 + 3 \cdot \frac{2^8-1}{5}\right) - 1}{511} - 1\right) \cdot \dfrac{2^{88} - 1}{2049}$

Remarquons que dans cet exemple, nous n'avons pas besoin de faire appel à l'algorithme "classique" `Inverse`

Exemple 2.3.15. Prenons $n = 101$ et $d = 13$. Nous allons calculer l'inverse de 13 modulo $2^{101} - 1$:

1 : $\theta_{13} \leftarrow 12$

2 : $r \leftarrow 5 \equiv 101 \pmod{12}$

3 : En utilisant `Inverse`, nous calculons que l'inverse de 13 modulo $2^5 - 1$ est 12

4 : $\mathrm{Inv}_{13}(101) = 12 \cdot 2^{96} + \left(\dfrac{155}{31} - 1\right) \cdot \dfrac{2^{96} - 1}{13}$

Remarquons que dans cette exemple, même si nous avons dû faire appel à l'algorithme `Inverse`, le calcul de l'inverse de 13 modulo $2^{101} - 1$ a tout de même été réduit à celui de 13 modulo $2^5 - 1$.

2.3.2. Exposants quadratiques

Dans cette sous-section, nous allons traiter l'étude de l'inverse des exposants quadratiques. Nous noterons $1 \leq d \leq 2^n - 2$ ces entiers dans tout le reste de cette sous-section.

Rappelons que les exposants quadratiques sont de la forme $d = 2^k + 1$. Lorsque n et d sont premiers en eux, ils sont généralement appelés exposants de Gold. Comme nous l'avons vu dans les tableaux 2.2 et 2.1, la condition $\gcd(d, n) = 1$ est nécessaire et suffisante pour que la fonction monomiale $x \mapsto x^d$ soit APN sur $\mathbb{F}_{2^n}$. De manière plus générale, il est possible de connaître l'uniformité différentielle des fonctions monomiales quadratiques :

Lemme 2.3.16. *Soient des entiers positifs k et n et soit $d = 2^k + 1$. Alors, le nombre de solutions de l'équation $\Delta_1(x \mapsto x^d) = x^d + (x+1)^d = b$ est*

$$\delta_{x \mapsto x^d}(1, b) \in \{0, 2^{\gcd(k,n)}\},$$

pour tout $b \in \mathbb{F}_{2^n}$. De plus, $\delta(x \mapsto x^d) = 2^{\gcd(k,n)}$.

Démonstration. Nous remarquons que d'après la propriété 2.1.3, il est en effet suffisant de considérer uniquement $\delta_{x \mapsto x^d}(1, b)$ avec $b \in \mathbb{F}_{2^n}$. Nous avons

$$\begin{aligned} \Delta_1(x \mapsto x^d) &= x^d + (x+1)^d \\ &= x^{2^k+1} + x^{2^k+1} + x^{2^k} + x + 1 \\ &= x^{2^k} + x + 1. \end{aligned}$$

Puisque $\Delta_1(x \mapsto x^d)$ est affine, le nombre de solutions de $\Delta_1(x \mapsto x^d) = b$ est soit 0, soit le même que le nombre de racines de la partie linéaire. C'est-à-dire le nombre de solutions de l'équation

$$\begin{aligned} \Delta_1(x \mapsto x^d) + 1 &= 0 \\ x^{2^k} + x &= 0. \end{aligned} \tag{2.21}$$

Les éléments x vérifiant l'équation (2.21) sont les éléments de $\mathbb{F}_{2^k}$. Puisque nous regardons les solutions dans $\mathbb{F}_{2^n}$, les solutions de (2.21) sont les éléments

$$x \in \mathbb{F}_{2^n} \cap \mathbb{F}_{2^k} \simeq \mathbb{F}_{2^{\gcd(k,n)}},$$

qui sont au nombre de $2^{\gcd(k,n)}$. □

Dans cette sous-section, nous nous intéresserons aux exposants quadratiques en général, incluant les exposants de Gold. Le lemme suivant, que l'on peut notamment trouver dans [156], nous donne une condition nécessaire et suffisante sur k et n pour que $\gcd(d, 2^n-1) = 1$ (c.à.d $x \mapsto x^d$ permute $\mathbb{F}_{2^n}$). Nous rajoutons la preuve pour plus de clarté.

Lemme 2.3.17 ([156, Lemme 11.1])**.** *Soient k et n des entiers positifs. Alors*

$$\gcd(2^k+1, 2^n-1) = \begin{cases} 1 & si\ \gcd(2k,n) = \gcd(k,n), \\ 2^{\gcd(k,n)}+1 & si\ \gcd(2k,n) = 2\gcd(k,n). \end{cases}$$

Démonstration. Puisque $(2^k+1)(2^k-1) = 2^{2k}-1$, nous avons que

$$\gcd(2^k+1, 2^n-1) \mid \gcd(2^{2k}-1, 2^n-1) = 2^{\gcd(2k,n)}-1,$$

en utilisant la propriété 2.1.4. Si $\gcd(2k,n) = \gcd(k,n)$, cela implique que

$$\gcd(2^k+1, 2^n-1) \mid 2^{\gcd(k,n)}-1 \mid 2^k-1.$$

Cependant, comme 2^k+1 et 2^k-1 sont premiers entre eux, cela signifie que $\gcd(2^k+1, 2^n-1) = 1$ dans ce cas là.

Considérons maintenant le cas où $\gcd(2k,n) = 2\gcd(k,n)$. Alors dans ce cas là, nous avons que

$$\gcd(2^k+1, 2^n-1) \mid (2^{\gcd(k,n)}+1)(2^{\gcd(k,n)}-1).$$

Mais puisque 2^k+1 et $2^{\gcd(k,n)}-1$ sont premiers entre eux, d'après le premier cas de ce lemme que nous venons de montrer, il s'en suit que

$$\gcd(2^k+1, 2^n-1) \mid 2^{\gcd(k,n)}-1.$$

Mais comme

$$2^{\gcd(k,n)}+1 \mid 2^{\gcd(2k,n)}-1 \mid 2^n-1, \quad \text{et} \quad 2^{\gcd(k,n)}+1 \mid 2^k+1,$$

puisque $k/\gcd(k,n)$ est impair par hypothèse, nous en concluons que

$$\gcd(2^k+1, 2^n-1) = 2^{\gcd(k,n)}+1.$$

□

Remarque 2.3.18. La condition $\gcd(2k,n) = \gcd(k,n)$ est équivalente à ce que $\frac{n}{\gcd(k,n)}$ soit impair.
De même, dire $\gcd(2k,n) = 2\gcd(k,n)$ est équivalent à dire que $\frac{n}{\gcd(k,n)}$ est pair.

Les inverses des exposants de Gold (*i.e.* quadratiques APN) ont déjà été trouvés par Kaisa Nyberg [166].

Proposition 2.3.19 ([166, Proposition 5])**.** *Soit n un entier positif impair, tel que k et n soient premiers entre eux. Alors*

$$\mathrm{Inv}_{2^k+1}(n) \equiv \frac{2^{k(n+1)}-1}{2^{2k}-1} \equiv \sum_{i=0}^{\frac{n-1}{2}} 2^{2ik} \pmod{2^n-1}.$$

De plus, $wt(\mathrm{Inv}_{2^k+1}(n)) = \frac{n+1}{2}$.

Il est à noter que même si cette formule offre une bonne lisibilité de la structure binaire de Inv_{2^k+1}, l'entier $\frac{2^{k(n+1)}-1}{2^{2k}-1}$ est égal au plus petit reste positif de l'inverse de 2^k+1 modulo 2^n-1 si et seulement si $k=1$. De plus, cette formule ne s'applique qu'aux exposants quadratiques APN (*i.e.* exposants de Gold).

Dans la suite de cette sous-section, nous allons voir qu'il est possible d'adapter de manière concrète les résultats de la sous-section précédente à tous les exposants quadratiques, généralisant de fait le résultat de Nyberg de la proposition 2.3.19.

Lemme 2.3.20. *Soit un entier $k \geq 1$. L'ordre de 2 modulo 2^k+1 est $\theta_{2^k+1} = 2k$.*

Démonstration. Il est clair que k est le plus petit entier positif vérifiant

$$2^k \equiv -1 \pmod{2^k+1},$$

ce qui implique que $\theta_{2^k+1} = 2k$. □

Nous savons d'après le théorème 2.3.8 et le lemme 2.3.20 qu'il est suffisant pour inverser les exposants quadratiques 2^k+1 modulo tous les 2^n-1 ($n \geq 1$) qui conviennent, de ne considérer que les inverses $\mathrm{Inv}_{2^k+1}(r)$ pour $1 \leq r < 2k$. Dans certains cas durant cette sous-section nous ne considérerons que $1 \leq r \leq k$ puisque nous pouvons observer la propriété suivante :

Propriété 2.3.21. *Soit $0 \leq r \leq k$. Alors $2^k+1 \equiv 2^k(2^r+1) \pmod{2^{k+r}-1}$. En particulier, $wt(\mathrm{Inv}_{2^r+1}(k+r)) = wt(\mathrm{Inv}_{2^k+1}(k+r))$.*

Le théorème suivant résume nos résultats concernant les inverses des exposants quadratiques.

Théorème 2.3.22. *Soient n et $k \geq 1$ des entiers positifs tels que $n/\gcd(k,n)$ soit impair. Supposons que r est le plus petit reste positif de n modulo $2k$. Alors*

(a)

$$\mathrm{Inv}_{2^k+1}(n) = 2^{n-r} \cdot \mathrm{Inv}_{2^k+1}(r) + \left(\frac{(2^k+1)\mathrm{Inv}_{2^k+1}(r)-1}{2^r-1} - 1\right)(2^k-1)\frac{2^{n-r}-1}{2^{2k}-1}.$$

(b) *Soit $\boldsymbol{w} = \frac{(2^k+1)\cdot \mathrm{Inv}_{2^k+1}(r)-1}{2^r-1} - 2$. Alors*

$$[\mathrm{Inv}_{\boldsymbol{2^k+1}}(\boldsymbol{n})]_n = [\mathrm{Inv}_{\boldsymbol{2^k+1}}(\boldsymbol{r})]_r \mid [\boldsymbol{w}]_k \mid \overline{[\boldsymbol{w}]}_k \mid \cdots \mid [\boldsymbol{w}]_k \mid \overline{[\boldsymbol{w}]}_k.$$

En particulier, $wt(\mathrm{Inv}_{2^k+1}(n)) = wt(\mathrm{Inv}_{2^k+1}(r)) + \frac{n-r}{2}$.

(c) *Le poids de Hamming binaire de* $\mathrm{Inv}_{2^k+1}(n)$ *est*

$$wt(\mathrm{Inv}_{2^k+1}(n)) = \frac{n-\gcd(k,n)}{2} + 1 = \frac{n-\gcd(k,n)+2}{2}. \tag{2.22}$$

Démonstration. L'égalité **(a)** provient directement de l'application de l'égalité **(a)** du théorème 2.3.8. Pour prouver **(b)**, il suffit de remarquer que

$$\begin{aligned} 2^k - (\boldsymbol{w}+1) &= 2^k + 1 - \frac{(2^k+1)\cdot \mathrm{Inv}_{2^k+1}(r) - 1}{2^r-1} \\ &= \frac{(2^k+1)\left(2^r - 1 - \mathrm{Inv}_{2^k+1}(r)\right) + 1}{2^r-1} > 0. \end{aligned}$$

Nous pouvons donc appliquer le lemme 2.1.10, puisque $\boldsymbol{w}+1 \leq 2^k$, conjointement avec la partie **(a)** de ce théorème pour conclure la preuve de **(b)**.

Nous allons prouver **(c)** par récurrence sur k_1, où $k_1 = k/\gcd(k,n)$. Si $k_1 = 1$ (c.à.d $k = \gcd(k,n)$, ou encore que k divise n), alors $n \equiv k \pmod{2k}$ puisque $n/\gcd(k,n) = n/k$ est impair par hypothèse. En conséquence, $r = k$. Nous observons que $\mathrm{Inv}_{2^k+1}(k) = 2^{k-1}$. D'après **(b)**, nous avons que

$$wt(\mathrm{Inv}_{2^k+1}(n)) = wt(\mathrm{Inv}_{2^k+1}(k)) + \frac{n-\gcd(k,n)}{2} = 1 + \frac{n-\gcd(k,n)}{2}.$$

Supposons maintenant que l'hypothèse de récurrence (2.22) est vraie pour tout $k_1 < \ell$ et prenons $k_1 = \ell$ (c.à.d $k = \gcd(k,n)\ell$). Posons $1 \leq r < 2k$ tel que $r \equiv n \pmod{2k}$. D'après **(b)**, le problème se réduit à trouver le poids de Hamming binaire de l'inverse de 2^k+1 modulo 2^r-1. Notons que dans le cas où $r < k$, alors il existe $\ell' < \ell$ tel que

$$2^k + 1 = 2^{\gcd(k,n)\ell} + 1 \equiv 2^{\gcd(k,n)\ell'} + 1 \pmod{2^r - 1}.$$

On en déduit que l'inverse de 2^k+1 modulo 2^r-1 est égal à celui de $2^{\gcd(k,n)\ell'}+1$, et en utilisant l'hypothèse de récurrence, nous obtenons

$$\begin{aligned} wt(\mathrm{Inv}_{2^k+1}(n)) &= wt(\mathrm{Inv}_{2^k+1}(r)) + \frac{n-r}{2} \\ &= wt(\mathrm{Inv}_{2^{\gcd(k,n)\ell'}+1}(r)) + \frac{n-r}{2} \\ &= 1 + \frac{r-\gcd(k,n)}{2} + \frac{n-r}{2} \\ &= 1 + \frac{n-\gcd(k,n)}{2}. \end{aligned}$$

Pour terminer la preuve, nous devons aussi montrer que l'hypothèse de récurrence est vraie dans le cas où $r \geq \gcd(k,n)\ell$. Posons $r = \gcd(k,n)\ell + r' = k + r'$, alors $r' = s\ell'$ pour un certain $\ell' < \ell$. Nous concluons en utilisant la proposition 2.3.21 : le poids de Hamming binaire des inverses de 2^k+1 et $2^{r'}+1$ modulo $2^{k+r'}-1$ sont égaux. □

Pour établir un lien plus explicite avec le corollaire 2.3.9 nous constatons que la valeur $\boldsymbol{w} = \frac{(2^k+1)\cdot \mathrm{Inv}_{2^k+1}(r)-1}{2^r-1} - 2$ du théorème précédent vérifie l'égalité :

$$\boldsymbol{w}2^k + (2^k - 1 - \boldsymbol{w}) = \left(\frac{(2^k+1)\mathrm{Inv}_{2^k+1}(r) - 1}{2^r-1} - 1\right)(2^{2k}-1).$$

Cela signifie en représentation binaire :

$$[\boldsymbol{w}]_k \mid \overline{[\boldsymbol{w}]}_k = [\mathrm{Inv}_{\boldsymbol{2^k+1}}(\boldsymbol{r})]_r \mid \overline{[\mathrm{Inv}_{\boldsymbol{2^k+1}}(\boldsymbol{\theta_{2^k+1}} - \boldsymbol{r})]}_{\theta_{2^k+1}-r}.$$

Remarque 2.3.23. Dans [23, Section 5.2], les auteurs montrent que l'uniformité différentielle des exposants quadratiques augmente avec $\gcd(k, n)$, comme nous l'avons rappelé en début de sous-section. Avec le théorème 2.3.22, nous constatons que le degré algébrique des inverses des fonctions monomiales quadratiques diminue avec ce même gcd.

Il est à noter que le calcul du poids de Hamming binaire des inverses des exposants quadratiques (**(c)** théorème 2.3.22) est notamment dû au fait que l'algorithme 2 ne fait pas appel à l'algorithme `Inverse`.

En effet, nous avons vu dans la preuve précédente et dans la proposition 2.3.21 que le calcul des inverses des quadratiques se fait par réductions modulaire successives (si $k < (n \pmod{\theta_{2^k+1}})$) en échangeant éventuellement le rôle de $\mathrm{Inv}_{2^k+1}(r)$ et $\mathrm{Inv}_{2^k+1}(\theta_{2^k+1} - r)$. Cela permet à l'algorithme récursif 2 de "terminer" à l'étape (2) plutôt qu'à l'étape (19).

Cette terminaison de l'algorithme 2 dans le cas des exposants quadratiques est démontrable. Il n'en reste pas moins un cas particulier. Dans les sous-sections suivantes, nous verrons les exposants de Kasami, dans lequel l'utilisation de l'algorithme `Inverse` sera généralement nécessaire. Nous verrons aussi le cas des exposants du type $2^k - 1$ pour lequel l'algorithme 2 pourra être grandement simplifié puisque bien moins de possibilités se présenteront.

2.3.3. Exposants de Kasami

Les exposants de Kasami sont des entiers de la forme $d = 2^{2k} - 2^k + 1$, $k \in \mathbb{N}^*$. D'après le lemme 2.1.6, nous savons d'ores et déjà que des fonctions monomiales créées à partir de tels exposants sont bijectives sur $\mathbb{F}_{2^n}$ lorsque n est impair et que $\gcd(k, n) = 1$ (en tant que fonctions monomiales APN en dimension impair).

Cependant, à notre connaissance, aucun résultat ne donne de conditions nécessaires et suffisantes pour qu'un exposant de Kasami décrive une permutation de $\mathbb{F}_{2^n}$ pour des entiers k et n arbitraires. Dans cette section nous donnons donc une condition nécessaire et suffisante pour qu'un exposant de Kasami induise une fonction monomiale bijective sur une extension de $\mathbb{F}_2$.

Lemme 2.3.24. *Soient k et n deux entiers positifs. Alors* $\gcd(2^{2k} - 2^k + 1, 2^n - 1) = 1$ *si et seulement si une des deux conditions suivantes est satisfaite :*

- **(a)** $\frac{n}{\gcd(k,n)}$ *est impair, c.à.d* $\gcd(2k, n) = \gcd(k, n)$ *;*
- **(b)** $\frac{n}{\gcd(k,n)}$ *et k sont pairs et* $\gcd(3k, n) = \gcd(k, n)$.

Dit autrement, $\gcd(2^{2k} - 2^k + 1, 2^n - 1) = 1$ *si et seulement si une des trois conditions suivantes est satisfaite :*

- *n est impair et $k \geq 1$;*
- *$n = 2^r a$ et $k = 2^r b$, où a est impair, $b \geq 1$ et $r \geq 1$;*
- *$n = 2^r 3^u a$ et $k = 2^s 3^v b$, où $a \geq 1$, b est impair, $r > s \geq 1$ et $v \geq u \geq 1$.*

Démonstration. Posons $d = 2^{2k} - 2^k + 1$. Dans cette preuve, nous utilisons de manière répétée le lemme 2.3.17. Tout d'abord nous remarquons que

$$\begin{aligned}\gcd(d, 2^k + 1) &= \gcd(2^{2k} - 2^k + 1, 2^k + 1) \\ &= \gcd(2^{k+1}(2^{k-1} - 1), 2^k + 1) \\ &= \gcd(2^{k-1} - 1, 2^k + 1) \\ &= \begin{cases} 1 & \text{si } k \text{ est pair,} \\ 2^{\gcd(k,k-1)} + 1 = 3 & \text{sinon.} \end{cases}\end{aligned}$$

Puisque l'entier $\frac{n}{\gcd(k,n)}$ est soit impair soit pair, nous n'avons donc que deux cas possibles.

(a) Supposons que $\frac{n}{\gcd(k,n)}$ est impair. Alors $\gcd(2^{rk} + 1, 2^n - 1) = 1$ pour tous les entiers $r > 1$ impairs. Dans ce cas nous avons que $\gcd(d, 2^n - 1) = 1$ puisque $(2^k + 1)d = 2^{3k} + 1$.

(b) Supposons que $\frac{n}{\gcd(k,n)}$ est pair. Alors $\gcd(2^{rk} + 1, 2^n - 1) = 2^{\gcd(rk,n)} + 1$ pour tous les entiers $r > 1$ impairs. Si k est impair, alors $3 = \gcd(d, 2^k + 1) | \gcd(d, 2^n - 1)$ car $3|2^n - 1$ dès lors que n est pair. Si k est pair, cela implique que $\gcd(d, 2^k + 1) = 1$ et donc que

$$\gcd(d(2^k + 1), 2^n - 1) = \gcd(d, 2^n - 1) \gcd(2^k + 1, 2^n - 1).$$

Cela signifie que $2^{\gcd(3k,n)} + 1 = \gcd(d, 2^n - 1)(2^{\gcd(k,n)} + 1)$ et donc que la condition $\gcd(3k, n) = \gcd(k, n)$ devient nécessaire et suffisante pour obtenir $\gcd(d, 2^n - 1)$.

□

Nous donnons aussi un résultat sur l'uniformité différentielle des fonctions monomiales avec un exposant de Kasami dû originellement à Doreen Hertel et Alexander Pott [110]. Une preuve du théorème suivant est donnée par Céline Blondeau, Anne Canteaut et Pascale Charpin [23].

Théorème 2.3.25. *Soient les entiers positifs $d = 2^{2k} - 2^k + 1$, $n \neq 3k$ et $s = \gcd(n, k)$ avec n/s impair. Soit la fonction $x \mapsto x^d$ sur $\mathbb{F}_{2^n}$. Alors*

$$\delta_{x \mapsto x^d}(1, \beta) \in \{0, 2^s\}, \qquad \text{pour tout } \beta \in \mathbb{F}_{2^n}.$$

En conséquence, $\delta(x \mapsto x^d) = 2^s$.

Proposition 2.3.26. *Soit un entier $k > 1$. L'ordre de 2 modulo $2^{2k} - 2^k + 1$ est $\theta_{2^{2k}-2^k+1} = 6k$.*

Démonstration. Nous allons montrer que $3k$ est le plus petit entier positif satisfaisant la congruence $2^{3k} \equiv -1 \pmod{2^{2k} - 2^k + 1}$. Il est clair que cette congruence est vérifiée puisque $2^{2k} - 2^k + 1$ divise $2^{3k} + 1$. Soit un entier $0 < s < 3k$ tel que $2^s \equiv -1 \pmod{2^{2k} - 2^k + 1}$. Alors $s \geq 2k - 1$ puisque $1 < 2^l < 2^{2k} - 2^k$ pour $0 < l \leq 2k - 1$. On peut donc supposer que $s = 2k + i$ avec $0 \leq i \leq k$. Notons que

$$2^{2k+i} \equiv 2^i(2^k - 1) \pmod{2^{2k} - 2^k + 1}.$$

Pour compléter la preuve, il suffit d'observer que $2^i(2^k - 1) < 2^{2k} - 2^k$ si $0 \leq i < k$. □

L'adaptation du théorème 2.3.8 aux exposants de Kasami est résumée dans le théorème suivant.

Théorème 2.3.27. *Soient des entiers $n, k \geq 1$ tels que* $\gcd(2^{2k} - 2^k + 1, 2^n - 1) = 1$. *Soit r le plus petit reste positif de n modulo $6k$. Alors*

(a)

$$\mathrm{Inv}_{2^{2k}-2^k+1}(n) = \frac{\mathrm{Inv}_{2^{2k}-2^k+1}(r)(2^n-1) - \frac{2^n-2^r}{2^{2k}-2^k+1}}{2^r-1}. \tag{2.23}$$

(b) *Posons* $\boldsymbol{w} = \left(\frac{(2^{2k}-2^k+1)\cdot \mathrm{Inv}_{2^{2k}-2^k+1}(r)-1}{2^r-1} - 1\right)(2^k+1) - 1$. *Alors*

$$[\mathrm{Inv}_{\mathbf{2^{2k}-2^k+1}}(\boldsymbol{n})]_n = [\mathrm{Inv}_{\mathbf{2^k+1}}(\boldsymbol{r})]_r \mid [\boldsymbol{w}]_{3k} \mid \overline{[\boldsymbol{w}]}_{3k} \mid \cdots \mid [\boldsymbol{w}]_{3k} \mid \overline{[\boldsymbol{w}]}_{3k}.$$

En particulier, $wt(\mathrm{Inv}_{2^{2k}-2^k+1}(n)) = wt(\mathrm{Inv}_{2^{2k}-2^k+1}(r)) + \frac{n-r}{2}$.

Démonstration. La preuve découle directement du théorème 2.3.8 et du lemme 2.1.10, et est similaire à celle du théorème 2.3.22. □

Pour établir un lien plus explicite avec le corollaire 2.3.9 nous constatons que la valeur $\boldsymbol{w} = \left(\frac{(2^{2k}-2^k+1)\cdot \mathrm{Inv}_{2^{2k}-2^k+1}(r)-1}{2^r-1} - 1\right)(2^k+1) - 1$ du théorème précédent vérifie l'égalité :

$$\boldsymbol{w}2^{3k} + (2^{3k} - 1 - \boldsymbol{w}) = \left(\frac{(2^{2k} - 2^k + 1)\mathrm{Inv}_{2^{2k}-2^k+1}(r) - 1}{2^r - 1} - 1\right)(2^{6k} - 1).$$

Cela signifie en représentation binaire :

$$[\boldsymbol{w}]_k \mid \overline{[\boldsymbol{w}]}_k = [\mathrm{Inv}_{\mathbf{2^{2k}-2^k+1}}(\boldsymbol{r})]_r \mid \overline{[\mathrm{Inv}_{\mathbf{2^{2k}-2^k+1}}(\mathbf{6k-r})]}_{6k-r}.$$

Nous remarquons qu'avec les résultats précédents, il est difficile d'obtenir une expression du poids de Hamming binaire des inverses des exposants de Kasami $2^{2k} - 2^k + 1$ modulo $2^n - 1$, qui ne dépende que de k et n.

Dans l'exemple qui suit, nous allons porter notre attention sur l'exposant de Kasami $13 = 2^4 - 2^2 + 1$.

Exemple 2.3.28. Considérons l'exposant de Kasami $13 = 2^4 - 2^2 + 1$. Le lemme 2.3.24 nous apprend que

$$\gcd(13, 2^n - 1) = 1 \Leftrightarrow n \not\equiv 0 \pmod{12}.$$

En utilisant le théorème 2.3.27, on obtient :

- Si $n \equiv 1 \pmod{12}$ alors

  $$\mathrm{Inv}_{13}(n) = 2^n - 1 - \frac{2^n - 2}{13},$$

  car $\mathrm{Inv}_{13}(1) = 1$. De plus dans ce cas, $wt(\mathrm{Inv}_{13}(n)) = (n+1)/2$.

- Si $n \equiv 2 \pmod{12}$ alors

  $$\mathrm{Inv}_{13}(n) = \frac{2^n - 1}{3} - \frac{2^n - 2^2}{3 \cdot 13},$$

  car $\mathrm{Inv}_{13}(2) = 1$. De plus dans ce cas, $wt(\mathrm{Inv}_{13}(n)) = n/2$.

- Si $n \equiv 3 \pmod{12}$ alors

$$\mathrm{Inv}_{13}(n) = \frac{6(2^n-1)}{7} - \frac{2^n - 2^3}{7 \cdot 13},$$

car $\mathrm{Inv}_{13}(3) = 6$. De plus dans ce cas, $wt(\mathrm{Inv}_{13}(n)) = (n+1)/2$.

- Si $n \equiv 4 \pmod{12}$ alors

$$\mathrm{Inv}_{13}(n) = \frac{7(2^n-1)}{15} - \frac{2^n - 2^4}{15 \cdot 13},$$

car $\mathrm{Inv}_{13}(4) = 7$. De plus dans ce cas, $wt(\mathrm{Inv}_{13}(n)) = (n+2)/2$.

- Si $n \equiv 5 \pmod{12}$ alors

$$\mathrm{Inv}_{13}(n) = \frac{12(2^n-1) - \frac{2^n - 2^5}{13}}{31},$$

car $\mathrm{Inv}_{13}(5) = 12$. De plus dans ce cas, $wt(\mathrm{Inv}_{13}(n)) = (n-1)/2$.

- Si $n \equiv 6 \pmod{12}$ alors

$$\mathrm{Inv}_{13}(n) = \frac{2^n-1}{63} - \frac{2^n - 2^6}{63 \cdot 13},$$

car $\mathrm{Inv}_{13}(6) = 34$. De plus dans ce cas, $wt(\mathrm{Inv}_{13}(n)) = (n-2)/2$.

Nous pouvons utiliser maintenant l'égalité (2.15) pour obtenir les autres cas :

- Si $n \equiv 7 \pmod{12}$ alors $wt(\mathrm{Inv}_{13}(n)) = (n-1)/2$ car $wt(\mathrm{Inv}_{13}(7)) = 3$.
- Si $n \equiv 8 \pmod{12}$ alors $wt(\mathrm{Inv}_{13}(n)) = (n+2)/2$ car $wt(\mathrm{Inv}_{13}(8)) = 5$.
- Si $n \equiv 9 \pmod{12}$ alors $wt(\mathrm{Inv}_{13}(n)) = (n+1)/2$ car $wt(\mathrm{Inv}_{13}(9)) = 5$.
- Si $n \equiv 10 \pmod{12}$ alors $wt(\mathrm{Inv}_{13}(n)) = n/2$ car $wt(\mathrm{Inv}_{13}(10)) = 5$.
- Si $n \equiv 11 \pmod{12}$ alors $wt(\mathrm{Inv}_{13}(n)) = (n+1)/2$ car $wt(\mathrm{Inv}_{13}(11)) = 6$.

Puisqu'il semble difficile d'obtenir des résultats plus généraux concernant les poids de Hamming binaire des inverses des exposants de Kasami, nous résumons dans les Tables 2.3, 2.4 et 2.5 ces valeurs lorsque $k = 3$, 4 et 5 respectivement.

r	1	3	5	7	9	11	13	15	17
$wt(\mathrm{Inv}_d(r))$	1	1	2	4	2	6	6	7	9

TABLE 2.3. : Poids de Hamming binaire des inverses de $d = 2^6 - 2^3 + 1$ modulo $2^r - 1$, $1 \leq r \leq 17$

Dans la proposition 2.3.29, nous rassemblons quelques cas particuliers notables concernant la forme des inverses des exposants de Kasami.

r	1	2	3	4	5	6	7	8	9	10	11	12
$wt(\mathrm{Inv}_d(r))$	1	1	2	1	3	2	4	5	5	3	4	2
r	13	14	15	16	17	18	19	20	21	22	23	
$wt(\mathrm{Inv}_d(r))$	5	5	8	9	9	8	10	9	11	11	12	

TABLE 2.4. : Poids de Hamming binaire des inverses de $d = 2^8 - 2^4 + 1$ modulo $2^r - 1$, $1 \leq r \leq 23$

r	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29
$wt(\mathrm{Inv}_d(r))$	1	2	1	3	5	5	7	2	9	9	11	11	11	14	15

TABLE 2.5. : Poids de Hamming binaire des inverses de $d = 2^{10} - 2^5 + 1$ modulo $2^r - 1$, $1 \leq r \leq 29$

Proposition 2.3.29. *Soient des entiers $k, n \geq 1$ tels que* $\gcd(2^{2k} - 2^k + 1, 2^n - 1) = 1$. *Alors*

1. *Si $n \equiv b \pmod{6k}$, où $1 \leq b < 6k$ est un diviseur de k, alors* $\mathrm{Inv}_{2^{2k}-2^k+1}(k/b) = 1$ *et*
$$\mathrm{Inv}_{2^{2k}-2^k+1}(n) = 2^{n-k/b} + \left(\frac{2^k(2^k-1)}{2^{k/b}-1} - 1\right) \cdot \frac{2^{n-k/b}-1}{2^{2k}-2^k+1}.$$
2. $\mathrm{Inv}_{2^{2k}-2^k+1}(k-1) = \frac{2^k-1}{3}$.
3. $\mathrm{Inv}_{2^{2k}-2^k+1}(k+1) = \frac{2^{k+2}-1}{3} + 1$.
4. $\mathrm{Inv}_{2^{2k}-2^k+1}(2k) = (2^k+2) \cdot \frac{2^k-1}{3} + 1$.
5. *Si $n \equiv 3k/b \pmod{6k}$, où $1 \leq b < 6k$ est un diviseur de k avec* $\gcd(b,3) = 1$. *Soit b' le plus petit reste positif de b modulo 3, alors*
$$\mathrm{Inv}_{2^{2k}-2^k+1}(3k/b) = 2^{3k/b-1} + 2^{b' \cdot k/b-1}.$$
6. $\mathrm{Inv}_{2^{2k}-2^k+1}(4k) = (1 + 2^{k+1} + 2^{2k} + 2^{3k+1}) \cdot \frac{2^k-1}{3} + 2^k$.
7. $\mathrm{Inv}_{2^{2k}-2^k+1}(5k) = 2^{5k} - 2^{4k} + 2^{2k} + 2^k - 1 \equiv 2^{2k}(2^{4k} - 2^{2k} + 1) \pmod{2^{5k} - 1}$.
8. $\mathrm{Inv}_{2^{2k}-2^k+1}(6k-1) = (2^{3k}-1)(2^k+1)$.

Notons que dans les cas (2) à (8) de cette proposition, n dépend de k, et nous nous retrouvons donc dans la même situation que dans la Section 2.2.

Nous conjecturons aussi que, pour tout entier positif b divisant k et premier avec 5, nous avons
$$\mathrm{Inv}_{2^{2k}-2^k+1}(5k/b) \equiv 2^u(2^{2v} - 2^v + 1) \pmod{2^{5k/b} - 1},$$
pour des entiers positifs u et v. Plus concrètement, cela signifie que pour ces choix d'entiers b et k, l'inverse d'un exposant de Kasami modulo $2^{5k/b} - 1$ est dans la classe cyclotomique d'un exposant de Kasami.

Pour finir avec les exposants de Kasami, nous remarquons que dans certains cas, le calcul de l'inverse d'exposants de Kasami peut se réduire à la recherche d'inverse d'exposants quadratiques.

Propriété 2.3.30. *Soient des entiers positifs k et n tels que $\frac{n}{\gcd(k,n)}$ soit impair. Alors les entiers 2^k+1 et $2^{3k}+1$ sont tous les deux premiers avec 2^n-1, et nous avons donc que*

$$\mathrm{Inv}_{2^{2k}-2^k+1}(n) \equiv (2^k+1)\mathrm{Inv}_{2^{3k}+1}(n) \pmod{2^n-1}.$$

Démonstration. Nous savons que

$$2^{2k}-2^k+1 = \frac{2^{3k}+1}{2^k+1}.$$

Et comme $\frac{n}{\gcd(k,n)}$ est impair, les entiers 2^k+1 et $2^{3k}+1$ sont inversibles modulo 2^n-1. □

2.3.4. Exposants 2^k-1

Dans [24], l'étude des exposants du type 2^k-1 montre qu'ils possèdent d'intéressantes propriétés cryptographiques bien que n'étant pas APN. La propriété de bijection de ces exposants est un fait bien connu que nous avons rappelé à la propriété 2.1.4.

Puisqu'un même exposant 2^k-1, k fixé, est inversible modulo 2^n-1 pour une infinité de n, nous nous intéressons dans cette sous-section à l'étude des inverses de ces exposants par les résultats des sous-sections précédentes.

Théorème 2.3.31. *Soient des entiers $n, k \geq 1$ premiers entre eux. Alors*

$$\mathrm{Inv}_{2^k-1}(n) \equiv \frac{2^{k\cdot s}-1}{2^k-1} \pmod{2^n-1},$$

où s est n'importe quel entier positif satisfaisant $sk \equiv 1 \pmod n$.

Plus précisément, si k_n^{-1} est le plus petit reste positif de l'inverse de k modulo n, alors

$$\mathrm{Inv}_{2^k-1}(n) = \sum_{i=0}^{k_n^{-1}-1} 2^{ki} \pmod n. \tag{2.24}$$

De plus, $wt(\mathrm{Inv}_{2^k-1}(n)) = k_n^{-1}$.

Démonstration. Posons $sk-1 = nm$. Alors,

$$(2^k-1)\frac{2^{ks}-1}{2^k-1} = 2^{ks}-1 = 2^{nm+1}-1 \equiv 1 \pmod{2^n-1}.$$

Il suffit de prendre $s = k_n^{-1}$ pour obtenir l'équation (2.24). Le fait que $wt(\mathrm{Inv}_{2^k-1}(n)) = k_n^{-1}$ provient du fait que k est premier avec n et donc tous les $ki \pmod n$, $0 \leq i < k_n^{-1}$, sont tous différents dans la somme de l'équation (2.24). □

Nous pouvons faire le constat qu'inverser les exposants 2^k-1 modulo 2^n-1 revient au problème d'inverser k modulo n. Nous pouvons imaginer l'apparition du problème récursivement :

Exemple 2.3.32. Prenons $n = 2^5 - 1$ et $k = 2^3 - 1$. Ainsi nous cherchons à calculer l'inverse de $2^k - 1 = 2^7 - 1 = 127$ modulo $2^n - 1 = 2^{31} - 1 = 2147483647$. Nous pouvons utiliser l'équation (2.24) et nous rendre compte qu'il suffit de calculer l'inverse de $k = 2^3 - 1 = 7$ modulo $n = 2^5 - 1 = 31$ qui une nouvelle fois peut être réduit à calculer l'inverse de 3 modulo 5 qui vaut 2. Ainsi, l'inverse de $k = 7$ modulo $n = 31$ vaut

$$\mathrm{Inv}_7(5) = \sum_{i=0}^{2-1} 2^{3i} \pmod 5 = 1 + 2^3 = 9,$$

en utilisant l'équation (2.24). Nous pouvons donc en conclure, en utilisant une nouvelle fois l'équation (2.24), que l'inverse de 127 modulo 2147483647 vaut :

$$\begin{aligned}\mathrm{Inv}_{127}(31) &= \sum_{i=0}^{9-1} 2^{7i} \pmod{31} \\ &= 1 + 2^7 + 2^{14} + 2^{21} + 2^{28} + 2^4 + 2^{11} + 2^{18} + 2^{25} \\ &= 304367761,\end{aligned}$$

et

$$[\mathbf{304367761}]_{2^{31}-1} = 00100100010010001001000010010001.$$

Nous remarquons aussi que l'algorithme 2 peut être grandement simplifié. En effet, la proposition suivante permet d'éliminer des possibilités par rapport à l'algorithme 2 lors du calcul de l'inverse de $2^k - 1$ modulo $2^n - 1$.

Proposition 2.3.33. *Soit un entier $k \geq 1$. L'ordre de 2 modulo $2^k - 1$ est $\theta_{2^k-1} = k$.*

Algorithme 3 Algorithme 2 adapté aux exposants $2^k - 1$

Entrée :

des entiers $n, k \geq 1$ premiers entre eux.

Sortie :

$\mathrm{Inv}_{2^k-1}(n)$, l'inverse de $2^k - 1$ modulo $2^n - 1$.

Fonction $\mathrm{Inv}_{2^k-1}(n)$

1: **Si** $n = 1$ ou $k = 1$
2: **Retourner** 1
3: $r \leftarrow n \pmod k$
4: **Si** $r \neq n$ ▷ $n > k$
5: **Retourner** $\frac{\mathrm{Inv}_{2^k-1}(r)(2^n-1) - \frac{2^n-2^r}{2^k-1}}{2^r-1}$
6: **Sinon**
7: $k' \leftarrow k \pmod n$
8: **Retourner** $\mathrm{Inv}_{2^{k'}-1}(n)$

Exemple 2.3.34. Reprenons les paramètres de l'exemple précédent, à savoir $k = 7$ et $n = 31$. Nous cherchons à calculer l'inverse de 127 modulo $2^{31} - 1 = 2147483647$:

1 : $r \leftarrow 3 \equiv 31 \pmod 7$

2 : $3 \neq 31$ alors retourner $\dfrac{\mathrm{Inv}_{2^7-1}(3)(2^{31}-1) - \frac{2^{31}-2^3}{2^7-1}}{2^3-1}$

2.1 : maintenant $n = 3$ et $k = 7$

2.2 : $k' \leftarrow 1 \equiv 7 \pmod 3$

2.3 : retourner $\mathrm{Inv}_{2^1-1}(3)$

2.3.1 : maintenant $n = 3$ et $k = 1$

2.3.2 : retourner 1

3 : $\mathrm{Inv}_{127}(31) = \dfrac{(2^{31}-1) - \frac{2^{31}-2^3}{2^7-1}}{2^3-1} = 304367761$

De plus, nous pouvons en déduire la représentation binaire de l'inverse de 127 modulo $2^{31} - 1 = 2147483647$:

$$\begin{aligned}[\mathbf{304367761}]_{31} &= [\mathbf{1}]_3 \mid \overline{[\mathbf{13}]}_4 \mid [\mathbf{1}]_3 \mid \overline{[\mathbf{13}]}_4 \mid [\mathbf{1}]_3 \mid \overline{[\mathbf{13}]}_4 \mid [\mathbf{1}]_3 \mid \overline{[\mathbf{13}]}_4 \mid [\mathbf{1}]_3 \\ &= 001 \mid 0010 \mid 001 \mid 0010 \mid 001 \mid 0010 \mid 001 \mid 0010 \mid 001,\end{aligned}$$

où 13 est obtenu en utilisant l'équation (2.15) du corollaire 2.3.7.

3. Polynômes de permutation creux avec une uniformité différentielle faible

En cryptographie symétrique, il est important de choisir attentivement les fonctions utilisées dans les boîtes-S. La conception de telles fonctions comporte de multiples problèmes puisqu'elles doivent posséder de bonnes propriétés cryptographiques, être évaluables efficacement (logiciel et matériel) et, lorsque nécessaire, elles doivent aussi être bijectives (voir chapitre 1).

Dans ce chapitre, nous présentons et étudions de nouvelles classes de permutations sur $\mathbb{F}_{2^n}$ ayant une bonne résistance à la cryptanalyse différentielle (uniformité différentielle basse) et qui sont faciles à mettre en œuvre puisqu'elles sont construites à partir de peu de monômes. Nous appelons les polynômes ayant peu de coefficients non nuls, des *polynômes creux*.

Une idée naturelle pour construire de nouvelles fonctions ayant certaines propriétés intéressantes, est de *déformer* légèrement, *modifier* des fonctions connues, en espérant conserver ces propriétés. Un moyen efficace de réaliser cela, et qui donne en général des résultats satisfaisants, est d'ajouter à la fonction connue, une fonction particulière. Cette fonction particulière peut être de plusieurs natures, mais rajouter une fonction booléenne (*i.e.* la trace absolue d'une fonction) s'est montré fructueux par le passé. Entre 2008 et 2010, Pascale Charpin et Gohar Kyureghyan [65, 66, 67] ont montré qu'il était possible de construire par cette méthode de nouvelles classes de polynômes de permutation à partir de polynômes de permutation connus ou de polynômes linéarisés. En 2009, Lilya Budaghyan, Claude Carlet et Gregor Leander [47] ont construit de nouvelles fonctions APN en déformant des fonctions APN connues.

Rajouter une fonction trace à une fonction connue pour essayer d'en garder les propriétés est en fait une méthode qui consiste à modifier, remplacer, certaines des fonctions coordonnées de la fonction vectorielle. Yves Edel et Alexander Pott ont livré un travail approfondi qui vise à étudier précisément la construction de fonction APN à l'aide de cette méthode [94].

Dans un premier temps, nous effectuons des rappels concernant la technique de remplacement de fonctions coordonnées. Nous introduisons aussi la formule de l'inverse pour la composition de permutations construites à partir de cette méthode.

Nous étudions ensuite intensivement le cas des fonctions monomiales *déformées* par des fonctions booléennes monomiales elles aussi. C'est à dire que nous considérons les fonctions du type :

$$\begin{array}{ccc} \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ x & \mapsto & x^s + \gamma \mathrm{Tr}(x^t). \end{array}$$

Nous décrivons entre autre l'ensemble de toutes ces fonctions qui sont bijectives, ainsi que l'expression explicite de leur inverse pour la composition des fonctions. Nous montrons que lorsque n est impair, si la fonction $x \mapsto x^s$ est AB, alors les fonctions décrites ci-dessus sont des fonctions 2-to-1, c'est à dire que chaque élément dans l'image possède exactement

deux antécédents. Lorsque n est impair, nous montrons qu'il est possible d'obtenir des permutations dont l'uniformité différentielle est 4 et qui ont un degré algébrique élevé.

Ensuite, nous considérons des classes particulières de fonctions de ce type, pour lesquels nous déterminons la taille de leur image, leur degré algébrique et leur uniformité différentielle. Nous étudions notamment le cas où $x \mapsto x^s$ est la fonction inverse. Nous montrons qu'en remplaçant une fonction coordonnée de cette fonction, l'uniformité différentielle est soit 4 soit 6, et nous construisons des fonctions bijectives ayant une uniformité différentielle 6. Nous étudions aussi les fonctions construites à partir de deux monômes quadratiques.

Les travaux présentés dans ce chapitre ont été effectués en commun avec Pascale Charpin et Gohar Kyureghyan [69].

3.1. Constructions par modification de coordonnées d'une fonction vectorielle

L'idée générale de cette construction est de remplacer certaines fonctions coordonnées d'une fonction vectorielle F. En particulier, soit la fonction vectorielle suivante,

$$\begin{array}{rccc} F: & \mathbb{F}_2^n & \to & \mathbb{F}_2^n \\ & x & \mapsto & (f_0(x), \ldots, f_{n-1}(x)), \end{array}$$

et soit une fonction booléenne $g : \mathbb{F}_2^n \to \mathbb{F}_2$. Nous pouvons "créer" une nouvelle fonction vectorielle en remplaçant juste une fonction coordonnée f_i par la fonction booléenne g. Nous pouvons par exemple obtenir la fonction F' donnée par

$$\begin{array}{rccc} F': & \mathbb{F}_2^n & \to & \mathbb{F}_2^n \\ & x & \mapsto & (\ldots, f_{j-1}(x), g(x), f_{j+1}(x), \ldots). \end{array}$$

D'un point de vue univarié, les fonctions booléennes coordonnées f_i de la fonction F peuvent s'écrire

$$f_i : x \mapsto \mathrm{Tr}(\beta_i F(x)),$$

pour $\mathcal{B} = \{\beta_0, \ldots, \beta_{n-1}\}$ une base de $\mathbb{F}_{2^n}$ sur $\mathbb{F}_2$. Supposons que la base duale de la base $\mathcal{B}$ soit la base $\mathcal{A} = \{\alpha_0, \ldots, \alpha_{n-1}\}$. La fonction F' peut alors s'écrire

$$F' : x \mapsto F(x) + \alpha_j(g(x) + \mathrm{Tr}(\beta_j F(x))).$$

En effet, les fonctions coordonnées de F' sont données par

$$\begin{aligned} \mathrm{Tr}(\beta_i F'(x)) &= \mathrm{Tr}(\beta_i F(x)) + \mathrm{Tr}(\alpha_j \beta_i)\,(g(x) + \mathrm{Tr}(\beta_j F(x))) \\ &= \begin{cases} g(x) & \text{si } i = j, \\ \mathrm{Tr}(\beta_i F(x)) & \text{sinon.} \end{cases} \end{aligned}$$

Par contre, remplacer une fonction coordonnée par une fonction booléenne revient à modifier la moitié de l'ensemble des fonctions composantes d'une fonction vectorielle :

$$\mathrm{Tr}(\lambda F'(x)) = \mathrm{Tr}(\lambda F(x)) \quad \Leftrightarrow \quad \mathrm{Tr}(\lambda \alpha_j) = 0.$$

Cette méthode s'est révélée très pratique pour construire des permutations lorsque nous considérons des fonctions de $\mathbb{F}_{2^n}$, ainsi que des fonctions APN/AB. Dans les sous-sections qui suivent, nous verrons quelques uns de ces exemples.

3.1.1. Constructions de permutations

La modification des fonctions coordonnées permet de construire des fonctions bijectives. Dans cette section nous prenons l'exemple de la construction de nouvelles fonctions bijectives à partir de fonctions bijectives connues. Nous donnons aussi l'expression de l'inverse de polynômes de permutations obtenus par cette méthode.

Théorème 3.1.1 (Théorème 2, [65])**.** *Soit G une permutation de $\mathbb{F}_{2^n}$ et soit f une fonction booléenne quelconque à n variables. Alors la fonction F donnée par*

$$\begin{array}{rccl} F: & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & G(x)+\gamma f(x),\ \gamma \in \mathbb{F}_{2^n}, \end{array} \tag{3.1}$$

est une permutation de $\mathbb{F}_{2^n}$ si et seulement si γ est une 0-structure linéaire de $f \circ G^{-1}$.

Remarque 3.1.2. Soit F une fonction définie par (3.1). Pour tout élément $y = F(x)$ de $\mathbb{F}_{2^n}$, nous avons

$$G^{-1}(G(x)) = x = G^{-1}(y + \gamma f(x)).$$

Comme f est une fonction booléenne, il existe au plus deux antécédents à y par F.

Connaissant l'inverse de la permutation G, nous pouvons décrire les inverses des permutations définies au théorème précédent. Nous avons tout d'abord un résultat qui est une instance d'un résultat de Gohar Kyureghyan [130, Théorème 3].

Lemme 3.1.3. *Soit $Q : x \mapsto x + \gamma g(x)$ une fonction sur $\mathbb{F}_{2^n}$ où $\gamma \in \mathbb{F}_{2^n}^*$ est une 0-structure linéaire de la fonction booléenne à n variables g. Alors la fonction Q est involutive, c'est-à-dire $Q \circ Q : x \mapsto x$.*

Démonstration. Nous avons par hypothèse que $g(x) + g(x + \gamma) = 0$ pour tout $x \in \mathbb{F}_{2^n}$. Nous avons aussi que

$$Q(Q(x)) = (x + \gamma g(x)) + \gamma g\left(x + \gamma g(x)\right),$$

où clairement $Q(Q(x)) = x$ lorsque $g(x) = 0$. Maintenant lorsque $g(x) = 1$, nous avons

$$Q(Q(x)) = x + \gamma + \gamma g(x + \gamma) = x + \gamma + \gamma g(x) = x.$$

□

Théorème 3.1.4. *Soit $F : x \mapsto G(x) + \gamma f(x)$ une permutation de $\mathbb{F}_{2^n}$ comme définie au théorème 3.1.1. Alors son inverse pour la composition est*

$$F^{-1} : x \mapsto \left(G^{-1} \circ Q\right)(x) \qquad \text{où} \qquad Q(x) = x + \gamma\left(f \circ G^{-1}\right)(x).$$

Démonstration. D'après le théorème 3.1.1, γ est une 0-structure linéaire de la fonction Booléenne $f \circ G^{-1}$. Nous pouvons donc appliquer le lemme 3.1.3 : la fonction Q est involutive. Nous avons donc que pour tout $x \in \mathbb{F}_{2^n}$

$$\begin{aligned} F(G^{-1} \circ Q)(x) &= G\left((G^{-1} \circ Q)(x)\right) + \gamma f\left((G^{-1} \circ Q)(x)\right) \\ &= \left(G \circ G^{-1} \circ Q\right)(x) + \gamma\left(f \circ G^{-1} \circ Q\right)(x) \\ &= Q(x) + \gamma\left(f \circ G^{-1}\right)(Q(x)) \\ &= (Q \circ Q)(x) = x. \end{aligned}$$

□

Notons qu'il est possible de calculer une borne supérieure sur l'uniformité différentielle des fonctions du type $F : x \mapsto G(x) + \gamma \mathrm{Tr}(H(x))$.

Proposition 3.1.5 ([47] et [67, Proposition 3]). *Soient G et H des fonctions sur $\mathbb{F}_{2^n}$, et soit $\delta(G) = \rho$. Alors la fonction $F : x \to G(x) + \gamma\, Tr(H(x))$ vérifie $\delta(F) \leq 2\rho$ quel que soit $\gamma \in \mathbb{F}_{2^n}^*$.*

3.1.2. Construction de fonctions APN/AB

Dans [47], Lilya Budaghyan, Claude Carlet et Gregor Leander déforment des fonctions quadratiques pour tenter d'obtenir des fonctions APN qui sont inéquivalentes à des fonctions puissances. Ils arrivent ainsi à obtenir la fonction notable $x \mapsto x^3 + \mathrm{Tr}(x^9)$, qui est APN sur $\mathbb{F}_{2^n}$ pour n'importe quel n.

Leurs résultats peuvent être vus différemment. En effet, la propriété APN d'une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ peut aussi être établie par rapport à ses fonctions coordonnées $f_0, \ldots, f_{n-1}$. La fonction F est APN si et seulement si pour tout $\alpha \in \mathbb{F}_2^{n*}$ et tout $\beta_0, \ldots, \beta_{n-1} \in \mathbb{F}_2$, le système

$$\begin{cases} f_0(x) + f_0(x+\alpha) &= \beta_0 \\ f_1(x) + f_1(x+\alpha) &= \beta_1 \\ & \vdots \\ f_{n-1}(x) + f_{n-1}(x+\alpha) &= \beta_{n-1} \end{cases} \tag{3.2}$$

admet au plus deux solutions. Cela signifie que pour construire une nouvelle fonction APN à partir de la fonction F, il suffit de trouver une fonction booléenne g à substituer à l'une des fonctions f_i pour que le système d'équations (3.2) ait encore au plus deux solutions.

Pour en apprendre plus sur les constructions de fonctions APN en remplaçant les fonctions coordonnées (ou plus généralement composantes), nous recommandons au lecteur les articles de Yves Edel et Alexander Pott [94], ainsi que l'état de l'art de Gohar Kyureghyan [71, Chapitre 5].

3.2. Polynômes creux

Dans cette section, nous étudions les fonctions sur $\mathbb{F}_{2^n}$ ainsi définies :

Définition 3.2.1. Nous définissons la classe de fonction $F_{s,t,\gamma}$ sur $\mathbb{F}_{2^n}$ donnée par

$$\begin{array}{rccl} F_{s,t,\gamma} : & \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^s + \gamma \mathrm{Tr}(x^t), \end{array} \tag{3.3}$$

où γ est un élément non nul fixé de $\mathbb{F}_{2^n}$ et $1 \leq s, t \leq 2^n - 2$ sont des entiers fixés.

Le fait que ces fonctions soient construites avec deux monômes nous assurent une implémentation efficace.

Nous décrivons l'ensemble de ces fonctions qui sont bijectives sur le corps $\mathbb{F}_{2^n}$ ainsi que leurs inverses pour la composition de fonction. Nous considérons aussi quelques classes spécifiques pour lesquels nous donnons leur degré algébrique et leur uniformité différentielle. Cette étude fait suite à un travail initié par Pascale Charpin et Gohar Kyureghyan [65, 66].

3.2.1. La sous-classe des permutations

Nous allons tout d'abord caractériser les entiers s et t ainsi que l'élément $\gamma \in \mathbb{F}_{2^n}^*$ tels que la fonction $F_{s,t,\gamma}$ est une permutation de $\mathbb{F}_{2^n}$. Notons tout d'abord que l'entier s doit être premier avec $2^n - 1$:

Lemme 3.2.2. *Si* $\gcd(s, 2^n - 1) > 1$ *alors la fonction* $F_{s,t,\gamma}$ *n'est ni une permutation ni une fonction 2-to-1 sur* $\mathbb{F}_{2^n}$.

Démonstration. Soit $\ell = \gcd(s, 2^n - 1)$ avec $\ell \geq 3$. Alors il existe exactement ℓ éléments distincts $x_0, \ldots, x_{\ell-1} \in \mathbb{F}_{2^n}$ tels que $x_i^s = \omega$ pour un certain $\omega \in \mathbb{F}_{2^n}$ et $0 \leq i < \ell$. Nous avons donc

$$F_{s,t,\gamma}(x_i) \in \{\omega, \omega + \gamma\}, \qquad \text{pour tout} \quad 0 \leq i < \ell.$$

La fonction $F_{s,t,\gamma}$ n'est donc pas bijective. De plus, si $\ell \geq 5$, $F_{s,t,\gamma}$ n'est évidemment pas une fonction 2-to-1 non plus.

Supposons que $\ell = 3$ et que $F_{s,t,\gamma}$ est une fonction 2-to-1. Alors, il existe un élément $z \in \mathbb{F}_{2^n}$, $z \neq x_i$ pour tout $0 \leq i < \ell$, tel que $F_{s,t,\gamma}(z) \in \{\omega, \omega + \gamma\}$. Cela implique que $z^s = \omega + \gamma$, et il y a donc deux autres éléments z_0, z_1 tels que $z_i^s = z$ impliquant que $F_{s,t,\gamma}(z_i) \in \{\omega, \omega + \gamma\}$, qui est une contradiction avec le fait que la fonction $F_{s,t,\gamma}$ est 2-to-1. □

Le théorème suivant est une instance du théorème 7 de [67] dû à Pascale Charpin et Gohar Kyureghyan.

Théorème 3.2.3. *Soit une fonction* $F_{s,t,\gamma} : x \mapsto x^s + \gamma Tr(x^t)$ *sur* $\mathbb{F}_{2^n}$ *avec* $\gamma \in \mathbb{F}_{2^n}$. *Alors la fonction* $F_{s,t,\gamma}$ *est bijective sur* $\mathbb{F}_{2^n}$ *si et seulement si* $\gcd(s, 2^n - 1) = 1$,

$$t \equiv 2^j(2^i + 1)s \pmod{2^n - 1}, \text{ pour des entiers } 0 \leq i, j \leq n - 1, \ i \neq n/2,$$

et soit la condition (a) *soit la condition* (b) *est satisfaite :*

(a) $i = 0$ *et* $Tr(\gamma) = 0$;

(b) $i > 0$ *et* $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$ *avec* $Tr(\gamma^{2^i+1}) = 0$.

De plus, si $Tr(\gamma) = 1$ *lorsque* $i = 0$, *ou* $Tr(\gamma^{2^i+1}) = 1$ *lorsque* $i > 0$ *et* $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$, *alors la fonction* $F_{s,t,\gamma}$ *est 2-to-1.*

Démonstration. D'après le lemme 3.2.2, nous devons avoir que s est premier avec $2^n - 1$. Notons s^{-1} l'inverse de s pour la multiplication modulo $2^n - 1$.

Conformément au théorème 3.1.1, la fonction $F_{s,t,\gamma}$ est une permutation si et seulement si l'élément γ est une 0-structure linéaire de la fonction booléenne $x \mapsto \mathrm{Tr}(x^{ts^{-1}})$. Nous savons que les seules fonctions booléennes monomiales qui possèdent une structure linéaire sont quadratiques (voir [67, Théorème 5]), c'est-à-dire dans notre cas si et seulement si

$$ts^{-1} \equiv 2^j(2^i + 1) \pmod{2^n - 1},$$

pour certains entiers i et j.

Dans le cas où $i = 0$, nous remarquons que

$$F_{s,t,\gamma}(x) = x^s + \gamma \mathrm{Tr}(x^s).$$

D'après le théorème 3.1.1, nous savons que la fonction $F_{s,t,\gamma}$ est une permutation sur $\mathbb{F}_{2^n}$ si et seulement si γ est une 0-structure linéaire de la fonction booléenne $x \mapsto \mathrm{Tr}(x)$. Autrement dit, la fonction $F_{s,t,\gamma}$ est une permutation sur $\mathbb{F}_{2^n}$ si et seulement si $\mathrm{Tr}(\gamma) = 0$.

Dans le cas où $i \neq 0$, nous avons que $\mathrm{Tr}(x^{ts^{-1}}) = \mathrm{Tr}(x^{2^i+1})$ et γ est une 0-structure linéaire de $x \mapsto \mathrm{Tr}(x^{2^i+1})$ si et seulement si $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$, et alors $\mathrm{Tr}(\gamma^{2^i+1}) = 0$.

Maintenant, supposons que γ est une 1-structure linéaire de la fonction booléenne $x \mapsto \mathrm{Tr}(x^{ts^{-1}})$ alors

$$\mathrm{Tr}(x^{2^i+1}) + \mathrm{Tr}\left((x+\gamma)^{2^i+1}\right) = 1 \qquad \text{pour tout } x \in \mathbb{F}_{2^n}, \tag{3.4}$$

ou autrement dit que $\mathrm{Tr}(\gamma) = 1$ si $i = 0$ et

$$\begin{aligned}\mathrm{Tr}(x^{2^i}\gamma + \gamma^{2^i}x + \gamma^{2^i+1}) &= \mathrm{Tr}(x^{2^i}(\gamma + \gamma^{2^{2i}}) + \gamma^{2^i+1}) \\ &= \mathrm{Tr}(\gamma^{2^i+1}) \\ &= 1\end{aligned}$$

si $i > 0$ et $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$. Cela signifie que la fonction $F_{s,t,\gamma}$ est 2-to-1. En effet, d'après la remarque 3.1.2, l'équation $y = F_{s,t,\gamma}(x)$ se vérifie pour au plus deux éléments $x \in \mathbb{F}_{2^n}$, que nous noterons x_0 et x_1 avec $x_0 = (y+\gamma)^{s^{-1}}$ et $x_1 = y^{s^{-1}}$. Il suffit de prouver que ces solutions sont exactement au nombre de deux, pour tout $y \in \mathbb{F}_{2^n}$. Nous avons

$$\begin{aligned}F_{s,t,\gamma}(x_0) &= y + \gamma + \gamma\mathrm{Tr}((y+\gamma)^{2^i+1}) = y + \gamma + \gamma(1 + \mathrm{Tr}(y^{2^i+1})) \\ &= y + \gamma\mathrm{Tr}(x_1^t) = F_{s,t,\gamma}(x_1),\end{aligned}$$

d'après l'équation (3.4). Cela prouve que $F_{s,t,\gamma}(x_0) = y$ implique que $\mathrm{Tr}(x_1^t) = 0$ et donc que $F_{s,t,\gamma}(x_1) = y$, et réciproquement. □

Le cas (a) du théorème 3.2.3 est le cas trivial où la fonction $F_{s,t,\gamma}$ est EA-équivalente à la fonction puissance $x \mapsto x^s$. Il s'agit en effet de la composition de la fonction monomiale $x \mapsto x^s$ avec la fonction linéaire $x \mapsto x + \gamma\mathrm{Tr}(x)$. Si s est premier avec $2^n - 1$, c'est à dire que la fonction $x \mapsto x^s$ est bijective sur $\mathbb{F}_{2^n}$, alors la fonction $F_{s,t,\gamma}$ est une permutation de $\mathbb{F}_{2^n}$ si et seulement si la fonction linéaire $x \mapsto x + \gamma\mathrm{Tr}(x)$ est une permutation aussi, c'est-à-dire si et seulement si $\mathrm{Tr}(\gamma) = 0$. Rappelons aussi que dans le cas de l'EA-équivalence entre la fonction $F_{s,t,\gamma}$ et une fonction monomiale, nous avons le résultat suivant concernant l'uniformité différentielle, ainsi que la non-linéarité (voir section 1.5.3).

Corollaire 3.2.4. *Soit la fonction $F_{s,s,\gamma} : x \mapsto x^s + \gamma\, Tr(x^s)$ sur $\mathbb{F}_{2^n}$ avec $\gcd(s, 2^n-1) = 1$ et $Tr(\gamma) = 0$. Alors la fonction $F_{s,s,\gamma}$ est une permutation de $\mathbb{F}_{2^n}$ telle que*

$$\delta(F_{s,s,\gamma}) = \delta(x \mapsto x^s) \qquad et \qquad \mathcal{NL}(F_{s,s,\gamma}) = \mathcal{NL}(x \mapsto x^s).$$

De manière plus générale, en combinant le théorème 3.2.3 avec la proposition 3.1.5, nous obtenons une classe infinie de polynômes creux de permutations sur le corps $\mathbb{F}_{2^n}$ et ayant une borne supérieure sur l'uniformité différentielle.

Corollaire 3.2.5. *Soit un entier positif s tel que $\gcd(s, 2^n - 1) = 1$ et soit un entier $0 < i < n/2$. Soit la fonction $F_{s,s(2^i+1),\gamma}$ sur $\mathbb{F}_{2^n}$ définie par*

$$F_{s,s(2^i+1),\gamma} \ : \ x \mapsto x^s + \gamma\, Tr(x^{s(2^i+1)}). \tag{3.5}$$

Alors la fonction $F_{s,s(2^i+1),\gamma}$ est bijective lorsque $Tr(\gamma^{2^i+1}) = 0$ (et est une fonction 2-to-1 sinon) avec une borne supérieure démontrable sur son uniformité différentielle :

$$\delta(F_{s,s(2^i+1),\gamma}) \leq 2\delta(x \mapsto x^s).$$

Remarque 3.2.6. Avec les notations du théorème 3.2.3.

- Au lieu de s et $2^i + 1$, nous pouvons choisir n'importe quel représentant de leur classe cyclotomique (voir définition 2.1.1). En effet, $F_{s,2s(2^i+1),\gamma} = F_{s,s(2^i+1),\gamma}$ et

$$(F_{s,s(2^i+1),\gamma}(x))^2 = x^{2s} + \gamma^2 \mathrm{Tr}(x^{s(2^i+1)}), \qquad \forall x \in \mathbb{F}_{2^n}.$$

- Si la fonction monomiale $x \mapsto x^s$ est APN, alors $\delta(F_{s,s(2^i+1),\gamma}) \in \{2, 4\}$.

Nous terminerons cette sous-section par quelques remarques que nous pouvons formuler concernant les permutations $F_{s,t,\gamma}$ ayant une uniformité différentielle basse.

Proposition 3.2.7. *Soit la fonction $F_{s,t,\gamma} : x \mapsto x^s + \gamma\, Tr(x^t)$ sur $\mathbb{F}_{2^n}$ avec $\gamma \in \mathbb{F}_{2^n}^*$, et des entiers $1 \leq s, t \leq 2^n - 2$. Nous avons donc que :*

1. *si n est pair et que la fonction $x \mapsto x^s$ est APN, alors la fonction $F_{s,t,\gamma}$ n'est pas une permutation ;*
2. *si n est impair et $t = s(2^i + 1)$ où $\gcd(i, n) = 1$, alors la fonction $F_{s,t,\gamma}$ n'est pas une permutation. C'est une fonction 2-to-1 lorsque $\gcd(s, 2^n - 1) = 1$ et $\gamma = 1$.*

Démonstration.

1. D'après le lemme 2.1.6, nous savons que si la fonction $x \mapsto x^s$ est APN alors
$$\gcd(s, 2^n - 1) = \begin{cases} 1 & \text{si } n \text{ est impair,} \\ 3 & \text{si } n \text{ est pair.} \end{cases}$$
Donc lorsque n est pair, la fonction $x \mapsto x^s$ n'est pas bijective, et d'après le théorème 3.2.3 la fonction $F_{s,t,\gamma}$ ne peut être bijective sur le corps $\mathbb{F}_{2^n}$.

2. Nous appliquons de nouveau le théorème 3.2.3 pour un n impair. Lorsque i et n sont premiers entre eux, nous devons avoir $\gamma = 1$ pour construire une permutation. Puisque n est impair, $\mathrm{Tr}(1) = 1$ et donc la fonction $F_{s,t,\gamma}$ est 2-to-1.

□

Les fonctions $F_{s,t,\gamma}$ seraient encore plus intéressantes si l'on pouvait obtenir de nouvelles APN, bijectives, et de haut degré algébrique. Concernant la propriété APN, les seules fonctions de cette forme connues sont les fonctions $x \mapsto x^3 + \gamma\mathrm{Tr}(x^9)$ sur $\mathbb{F}_{2^n}$ pour tout entier positif n et pour des γ définis plus tard (à la section 3.4). Ces fonctions ne sont cependant pas bijectives. Dans le cas $\gamma = 1$, la fonction $F_{3,9,1}$ est celle donnée par Lilya Budaghyan, Claude Carlet et Gregor Leander [47].

Plus généralement :

Proposition 3.2.8. *Il n'y a pas de permutation sur $\mathbb{F}_{2^n}$ du type*

$$F : x \mapsto x^{2^j+1} + \gamma\, Tr(x^{(2^j+1)(2^i+1)}) \quad \text{avec } \gcd(j, n) = 1 \text{ et } \gcd(i, n) = 1.$$

Cela est notamment vrai pour la fonction $x \mapsto x^3 + \gamma\mathrm{Tr}(x^9)$, pour n'importe quel élément $\gamma \in \mathbb{F}_{2^n}$ et n'importe quel n.

Démonstration. Puisque la fonction $x \mapsto x^{2^j+1}$ est APN, la fonction F ne peut être une permutation d'après la Proposition 3.2.7. □

Exemple 3.2.9. Soit l'exposant de Kasami $s = 2^{2i} - 2^i + 1$ avec $\gcd(i, n) = 1$, et soit la fonction F de $\mathbb{F}_{2^n}$ donnée par

$$F : x \mapsto x^s + \gamma \mathrm{Tr}(x^{2^{3i}+1}), \qquad \gamma \in \mathbb{F}_{2^n}^*.$$

Rappelons que la fonction $x \mapsto x^s$ est APN et que $2^{3i} + 1 = (2^i + 1)s$ (voir section 2.3.3 du chapitre précédent). Nous avons donc $\delta(F) \leq 4$. Lorsque l'entier n est pair, la fonction F ne peut être bijective. Si n est impair, la fonction F est 2-to-1 lorsque $\gamma = 1$. Lorsque $i = 1$, nous obtenons la fonction $F_{3,9,1}$ qui est APN pour tout n.

L'inverse des permutations de la forme $F_{s,t,\gamma}$

Nous appliquons maintenant le théorème 3.1.4 aux fonctions $F_{s,t,\gamma} : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, définies à la définition 3.2.1, pour calculer leurs inverses lorsqu'elles sont bijectives. Il est important de rappeler que les inverses pour la composition possèdent la même non-linéarité et la même uniformité différentielle, mais que le degré algébrique ainsi que le nombre de coefficients de leur écriture univariée peut changer (voir section 1.5.3).

Théorème 3.2.10. *Soit une fonction $F_{s,t,\gamma}$ sur $\mathbb{F}_{2^n}$ vérifiant les conditions de bijectivité du théorème 3.2.3 avec $t = s(2^i + 1)$ pour un entier $i > 0$. Posons $\sigma = s^{-1} \pmod{2^n - 1}$. Alors*

$$\begin{aligned} F_{s,t,\gamma}^{-1}(x) &= \left(x + \gamma\, Tr(x^{2^i+1})\right)^{\sigma} \\ &= x^{\sigma} + \left(\sum_{j \prec \sigma} x^j \gamma^{\sigma-j}\right) Tr(x^{2^i+1}), \end{aligned}$$

pour tout $x \in \mathbb{F}_{2^n}$, où $j \prec \sigma$ signifie, rappelons-le, $supp(j) \subsetneq supp(\sigma)$.

Démonstration. Nous appliquons le théorème 3.1.4 avec $G : x \mapsto x^s$ (donc $G^{-1} : x \mapsto x^{\sigma}$) et

$$Q : x \mapsto x + \mathrm{Tr}\left(\left(G^{-1}(x)^t\right)\right) = x + \gamma \mathrm{Tr}(x^{t\sigma}) = x + \gamma \mathrm{Tr}(x^{2^i+1}).$$

Nous avons donc $F_{s,t,\gamma}^{-1} = G^{-1} \circ Q$. Plus précisément,

$$F_{s,t,\gamma}^{-1}(x) = \left(x + \gamma \mathrm{Tr}(x^{2^i+1})\right)^{\sigma} = \sum_{j \preceq \sigma} x^j \left(\gamma \mathrm{Tr}(x^{2^i+1})\right)^{\sigma-j}, \qquad \forall x \in \mathbb{F}_{2^n}.$$

Il ne reste plus qu'à observer que $\left(\mathrm{Tr}(x^{2^i+1})\right)^{\sigma-j} = \mathrm{Tr}(x^{2^i+1})$ pour tout $j \neq \sigma$. □

Nous pouvons remarquer plusieurs choses intéressantes concernant ces inverses. Tout d'abord, d'après le théorème 3.2.3, si la fonction $F_{s,t,\gamma}$ est bijective sur $\mathbb{F}_{2^n}$ alors $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$. Il en découle que la fonction $F_{s,t,\gamma}$ ainsi que son inverse ont des coefficients appartenant a un sous-corps de $\mathbb{F}_{2^n}$, le corps $\mathbb{F}_{2^{\gcd(2i,n)}}$.

Si nous supposons que $n = 2m$, alors la fonction $F_{s,t,\gamma}$ ne peut être APN si $\gcd(i, m) = 1$ ou $\gamma \in \mathbb{F}_{2^m}$. Cela est dû au fait qu'il n'existe pas de permutation APN qui corresponde à un polynôme de $\mathbb{F}_4[x]$ ou de $\mathbb{F}_{2^m}[x]$ (voir [91] et [14, Théorème 3]).

Proposition 3.2.11. *Soit une fonction $F_{s,t,\gamma}$ permutant le corps $\mathbb{F}_{2^n}$ avec $n = 2m$ et $\gamma \in \mathbb{F}_{2^{2\gcd(i,m)}}$. Si $\gamma \in \mathbb{F}_4$ ou $\gamma \in \mathbb{F}_{2^{\gcd(i,m)}}$, alors la fonction $F_{s,t,\gamma}$, ainsi que son inverse, ne peut être APN.*

Ensuite, nous notons que les résultats du chapitre 2 nous fournissent des outils efficaces pour calculer la valeur $\sigma = s^{-1} \pmod{2^n - 1}$ pour toutes les valeurs de n qui conviennent. De plus, le degré algébrique de l'inverse de $x \mapsto x^s$ (*i.e.* $wt(\sigma)$) influence le nombre de coefficients de la fonction $F^{-1}_{s,t,\gamma}$: plus $wt(\sigma)$ est petit, plus le polynôme $F^{-1}_{s,t,\gamma}$ est creux.

Nous pouvons aussi voir que le degré algébrique de la fonction $F^{-1}_{s,t,\gamma}$ vérifie

$$\deg(F^{-1}_{s,t,\gamma}) \leq wt(\sigma) + 1. \tag{3.6}$$

Bien qu'il semble difficile d'avoir des résultats plus précis concernant le degré algébrique, nous verrons dans la suite de ce chapitre que nous pourrons avoir des améliorations pour certaines classes de fonctions du type (3.3).

3.2.2. La sous-classe des fonctions 2-to-1

D'après le théorème 3.2.3, les fonctions sur $\mathbb{F}_{2^n}$ données par

$$F_{s,s(2^i+1),\gamma} : x \mapsto x^s + \gamma \mathrm{Tr}(x^{s(2^i+1)}), \qquad \text{avec } i > 0,\ \gcd(s, 2^n - 1) = 1 \text{ et } \gamma \in \mathbb{F}_{2^{\gcd(2i,n)}} \tag{3.7}$$

sont 2-to-1 si et seulement si $\mathrm{Tr}(\gamma^{2^i+1}) = 1$. Le théorème suivant nous montre que même dans ce cas là, il est encore possible de construire des permutations.

Théorème 3.2.12. *Soit la fonction $F_{s,i,\gamma}$ définie par l'équation (3.7) et soit $\sigma = s^{-1} \pmod{2^n - 1}$. Alors la fonction*

$$G_{s,i,\gamma} : x \mapsto x^s + \gamma\, Tr(x^{s(2^i+1)} + \gamma^{2^i} x^s)$$

est bijective sur $\mathbb{F}_{2^n}$ et son inverse est de la forme :

$$G^{-1}_{s,i,\gamma} : x \mapsto x^\sigma + \left(\sum_{j \prec \sigma} x^j \gamma^{\sigma - j} \right) Tr(x^{2^i+1} + \gamma^{2^i} x^s).$$

De plus, $\delta(G_{s,i,\gamma}) \leq 2\delta(x \mapsto x^s)$. En particulier, lorsque n est impair, si la fonction $x \mapsto x^s$ est APN alors $\delta(G_{s,i,\gamma}) \leq 4$.

Démonstration. Nous appliquons tout d'abord le théorème 3.1.1 : la fonction

$$R : x \mapsto x + \gamma \mathrm{Tr}(x^{2^i+1} + \gamma^{2^i} x)$$

est une permutation sur $\mathbb{F}_{2^n}$, puisque γ est une 0-structure linéaire de $x \mapsto \mathrm{Tr}(x^{2^i+1} + \gamma^{2^i} x)$. En effet, comme $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$, nous avons que

$$\mathrm{Tr}\left(x^{2^i+1} + (x+\gamma)^{2^i+1} + \gamma^{2^i} x + \gamma^{2^i}(x+\gamma) \right) = \mathrm{Tr}(x^{2^i}(\gamma^{2^{2i}} + \gamma)) = 0,$$

pour tout $x \in \mathbb{F}_{2^n}$ puisque $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}} \subseteq \mathbb{F}_{2^{2i}}$. Donc comme la fonction $x \mapsto x^s$ est bijective sur $\mathbb{F}_{2^n}$, la composition $x \mapsto R(x^s) = G_{s,i,\gamma}(x)$ est aussi une permutation de $\mathbb{F}_{2^n}$.

D'après le théorème 3.2.10 et le lemme 3.1.3, l'inverse pour la composition de la fonction $G_{s,i,\gamma}$ vérifie

$$\begin{aligned} G^{-1}_{s,i,\gamma}(x) &= \left(R^{-1}(x)\right)^{\sigma} = (R(y))^{\sigma} \\ &= x^{\sigma} + \left(\sum_{j \prec \sigma} x^j \gamma^{\sigma-j}\right) \mathrm{Tr}(x^{2^i+1} + \gamma^{2^i} x^s), \end{aligned}$$

pour tout $x \in \mathbb{F}_{2^n}$. Nous utilisons la proposition 3.1.5 pour compléter la preuve. □

3.3. Quelques classes de permutations spécifiques

Dans cette section, nous allons étudier quelques classes spécifiques de fonctions construites à partir de la forme (3.3). Notre but est de proposer des classes de permutations ayant une uniformité différentielle basse.

Depuis le début de ce chapitre, nous avons d'ores et déjà vu qu'il était relativement simple de construire des fonctions $F_{s,t,\gamma}$ dont l'uniformité différentielle est quatre. Lorsque n est impair, elles ne sont pas bijectives mais 2-to-1 ce qui n'enlève rien de l'intérêt que nous pouvons leur porter. Lorsque n est pair, nous pouvons aisément construire des permutations dont l'uniformité différentielle est au plus huit. Dans ce cas, les fonctions connues qui sont bijectives avec une uniformité différentielle quatre sont rares. Nous décrivons dans cette section deux classes de fonctions $F_{s,t,\gamma}$ pour lesquels leur uniformité différentielle est en dessous de la borne donnée par la proposition 3.1.5.

3.3.1. Les fonctions $F_{s,1,\gamma}$

Les fonctions que nous considérerons dans cette sous-section sont de la forme suivante :

$$F_{s,1,\gamma} : \begin{array}{ccc} \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ x & \mapsto & x^s + \gamma \mathrm{Tr}(x) \end{array}, \qquad \gcd(s, 2^n - 1) = 1, \quad \gamma \in \mathbb{F}^*_{2^n}, \tag{3.8}$$

où l'entier s n'est pas une puissance de deux. Rappelons que, conformément au théorème 3.2.3, la condition $\gcd(s, 2^n - 1) = 1$ est nécessaire pour que la fonction $F_{s,1,\gamma}$ soit bijective.

Nous allons décrire l'ensemble des permutations de cette forme et donner leurs paramètres (l'entier s et $\gamma \in \mathbb{F}^*_{2^n}$). Notons tout de même que l'étude de ces fonctions est un cas particulier de l'étude des fonctions bijectives de la forme $x \mapsto x^s + L(x)$, où L est une fonction linéaire sur $\mathbb{F}_2$ (voir [144, 145]). Dans [104], Faruk Göloğlu et Gary McGuire ont étudié ces fonctions avec $s = -1$ et sur des corps de caractéristiques impaires.

Théorème 3.3.1. *Soit $F_{s,1,\gamma}$ une fonction sur $\mathbb{F}_{2^n}$ définie par* (3.8)*. Alors, la fonction $F_{s,1,\gamma}$ est une permutation sur $\mathbb{F}_{2^n}$ si et seulement si*

$$s = \frac{2^j}{2^i + 1} \pmod{2^n - 1} \qquad \text{pour des entiers } 0 \le i, j \le n-1,\ i \ne n/2,$$

et les deux conditions suivantes sont vérifiées :

- $n/\gcd(i, n)$ *impair ;*
- $\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}}$ *tel que* $Tr(\gamma^{2^i+1}) = 0$.

Dans ce cas,

$$\delta(F_{s,1,\gamma}) = 2^{\gcd(i,n)} \quad \text{et} \quad \deg(F_{s,1,\gamma}) = \frac{n - \gcd(i,n) + 2}{2},$$

et la non-linéarité de la fonction $F_{s,1,\gamma}$ est égale à celle de la fonction $x \mapsto x^{2^i+1}$:

$$\mathcal{NL}(F_{s,1,\gamma}) = 2^{n-1} - 2^{(n+\gcd(2i,n)-2)/2}.$$

De plus,

$$F^{-1}_{s,1,\gamma} : x \mapsto x^{2^i+1} + \left(\gamma^{2^i+1} + \gamma^{2^i} x + \gamma x^{2^i}\right) Tr(x^{2^i+1}),$$

dans le cas où $j = 0$, avec $\deg(F^{-1}_{s,1,\gamma}) = 3$. Toutes les fonctions composantes de la fonction $F_{s,1,\gamma}$ ainsi que celles de son inverse sont plateaux.

Démonstration. Nous appliquons tout d'abord les résultats du théorème 3.1.1 à la fonction $F_{s,1,\gamma}$: la fonction $F_{s,1,\gamma}$ est bijective sur $\mathbb{F}_{2^n}$ si est seulement si γ est une 0-structure linéaire de la fonction booléenne $\mathrm{Tr}(x^{1/s})$. Nous savons d'après [67, Théorème 5] que de telles fonctions booléennes monomiales ont une structure linéaire si et seulement si elles sont quadratiques. Or dans le cas présent nous avons que $2^j/s = 2^i + 1$ pour des entiers i et j, ainsi la condition $n/\gcd(i,n)$ impair nous assure que $\gcd(s, 2^n - 1) = 1$ (voir lemme 2.3.17). De plus, lorsque $i > 0$, les conditions sur γ sont obtenues par le théorème 3.2.3.

L'uniformité différentielle de la fonction $F_{s,1,\gamma}$ est la même que celle de la fonction $x \mapsto x^{2^i+1}$, c'est-à-dire $\delta(F_{s,1,\gamma}) = 2^{\gcd(i,n)}$ (voir lemme 2.3.16).

Il en est de même pour la non-linéarité de cette fonction :

$$\mathcal{NL}(F_{s,1,\gamma}) = 2^{n-1} - 2^{(n+\gcd(2i,n)-2)/2}.$$

Nous utilisons le résultat du théorème 2.3.22 pour calculer le degré algébrique de la fonction $F_{s,1,\gamma}$:

$$\deg(F_{s,1,\gamma}) = wt(s) = wt(\mathrm{Inv}_{2^i+1}(n)) = \frac{n - \gcd(i,n) + 2}{2}.$$

La fonction $F^{-1}_{s,1,\gamma}$ est calculée en utilisant le théorème 3.2.10, et son degré algébrique est clairement $\deg(F^{-1}_{s,1,\gamma}) = 3$. □

Le fait que l'uniformité différentielle et la non-linéarité de la fonction $F_{s,1,\gamma}$ soient faciles à calculer provient du fait qu'il s'agit d'une fonction EA-équivalente à l'inverse d'une fonction monomiale. Nous avons déjà vu en introduction de cette thèse, à la section 1.5.3 plus précisément, que l'EA-équivalence ainsi que l'inversion pour la composition des fonctions laissaient invariantes l'uniformité différentielle et la non-linéarité. De plus nous étudions l'inverse d'une fonction quadratique monomiale, dont les critères cryptographiques sont bien connus (voir [23], [166] ou section 2.3.2 de ce document).

Nous pouvons assez facilement déduire du théorème précédent des constructions de fonctions creuses de $\mathbb{F}_{2^n}$ dont l'uniformité différentielle vaut quatre lorsque n est pair et qui sont APN lorsque n est impair.

Considérons tout d'abord le cas des fonctions sur $\mathbb{F}_{2^n}$ avec n pair.

Corollaire 3.3.2. *Soit $n = 2m$ avec m impair. Soit un entier i tel que $2 \leq i \leq m$ et $\gcd(i, n) = 2$. Soit $\gamma \in \mathbb{F}_{2^n}^*$. Alors la fonction*

$$F_{\frac{1}{(2^i+1)},1,\gamma} : x \mapsto x^{\frac{1}{2^i+1}} + \gamma\, Tr(x)$$

est une permutation sur $\mathbb{F}_{2^n}$ si et seulement si $\gamma = 1$. Cette fonction est 2-to-1 lorsque $\gamma \in \mathbb{F}_4 \backslash \mathbb{F}_2$.

De plus, cette fonction est plateau et vérifie

$$\delta(F_{\frac{1}{(2^i+1)},1,\gamma}) = 4, \qquad \deg(F_{\frac{1}{(2^i+1)},1,\gamma}) = m \quad \text{et } \mathcal{NL}(F_{\frac{1}{(2^i+1)},1,\gamma}) = 2^{n-1} - 2^{\frac{n}{2}}.$$

Lorsque $F_{\frac{1}{(2^i+1)},1,\gamma}$ est une permutation de $\mathbb{F}_{2^n}$, son inverse pour la composition est

$$F^{-1}_{\frac{1}{(2^i+1)},1,\gamma} : x \mapsto x^{2^i+1} + (1 + x + x^{2^i})\, Tr(x^{2^i+1}).$$

Démonstration. Cette preuve est essentiellement une application du théorème 3.3.1. Nous précisons tout de même que la fonction $x \mapsto x^{2^i+1}$ est bien une permutation sur $\mathbb{F}_{2^n}$ puisque $n/\gcd(i, n) = m$ et m est impair par hypothèse. Puisque $\gcd(2i, n) = 2$, la fonction $F_{\frac{1}{(2^i+1)},1,\gamma}$ est donc une permutation sur $\mathbb{F}_{2^n}$ si est seulement si $\gamma \in \mathbb{F}_4^*$ et $\mathrm{Tr}(\gamma^{2^i+1}) = 0$. Or i est pair et $\gamma \in \mathbb{F}_4$ donc $\gamma^{2^i+1} = \gamma^2$, d'où

$$\mathrm{Tr}(\gamma^{2^i+1}) = \mathrm{Tr}(\gamma) = \begin{cases} 0 & \text{si } \gamma \in \{0, 1\}, \\ 1 & \text{sinon.} \end{cases}$$

Seul le choix $\gamma = 1$ est possible pour que la fonction $F_{\frac{1}{(2^i+1)},1,\gamma}$ soit une permutation de $\mathbb{F}_{2^n}$. Le reste de la preuve découle directement du théorème 3.3.1. □

Dans [42], Carl Bracken, Chik How Tan et Yin Tan indiquent que le corpus des permutations sur $\mathbb{F}_{2^n}$, n pair, différentiellement 4-uniforme et qui ont un haut degré algébrique est de petite taille. Le corollaire 3.3.2 étend le nombre de telles fonctions connues. Bien que les fonctions considérées ici proviennent de transformations de fonctions monomiales quadratiques, elles possèdent un haut degré algébrique.

Regardons maintenant le cas des fonctions sur $\mathbb{F}_{2^n}$ avec n impair.

Corollaire 3.3.3. *Soient un entier positif impair n et un entier $2 \leq i \leq n$.*

(a) *Si $\gcd(i, n) = 1$, alors la fonction sur $\mathbb{F}_{2^n}$ donnée par*

$$F_{\frac{1}{2^i+1},1,1} : x \mapsto x^{\frac{1}{2^i+1}} + Tr(x)$$

est 2-to-1, AB et de degré algébrique $(n + 1)/2$.

(b) *Soit $\gamma \in \mathbb{F}_8$ avec $Tr(\gamma) = 0$. Si $\gcd(i, n) = 3$ alors la fonction*

$$F_{\frac{1}{2^i+1},1,\gamma} : x \mapsto x^{\frac{1}{2^i+1}} + \gamma\, Tr(x)$$

est bijective sur $\mathbb{F}_{2^n}$, avec $\delta(F_{\frac{1}{2^i+1},1,\gamma}) = 8$ et $\deg(F_{\frac{1}{2^i+1},1,\gamma}) = (n - 1)/2$.

Démonstration. Lorsque $\gcd(i,n)=1$, la fonction $x \mapsto x^{2^i+1}$ est AB (donc APN, voir proposition 1.5.28) ainsi que son inverse pour la composition. Donc la fonction $F_{\frac{1}{2^i+1},1,1}$ est aussi AB. Elle est 2-to-1 car d'après le théorème 3.2.3 il faut choisir un élément $\gamma \in \mathbb{F}_2$ non nul et comme n est impair le seul choix possible est $\gamma = 1$.

Lorsque $\gcd(i,n) > 1$, nous pouvons construire des permutations de $\mathbb{F}_{2^n}$ de la forme de $F_{\frac{1}{2^i+1},1,1}$ et dont les propriétés découlent directement du théorème 3.3.1. □

Remarquons qu'à l'aide du théorème 3.2.12, nous pouvons construire des permutations sur $\mathbb{F}_{2^n}$ héritées des fonctions du corollaire précédent lorsque n est impair et qui sont différentiellement 4-uniformes. Plus précisément, les fonctions sur $\mathbb{F}_{2^n}$, n impair, du type :

$$G_{\frac{1}{2^i+1},i,1} : x \mapsto x^{\frac{1}{2^i+1}} + \mathrm{Tr}(x + x^{\frac{1}{2^i+1}}), \quad 2 \leq i \leq n \quad \text{avec } \gcd(i,n)=1,$$

sont bijectives et telles que $\delta(G_{\frac{1}{2^i+1},i,1}) \leq 4$. En fait, $\delta(G_{\frac{1}{2^i+1},i,1}) = 4$ car ces fonctions possèdent une composante linéaire [14, Théorème 2]. En effet,

$$\mathrm{Tr}\left(G_{\frac{1}{2^i+1},i,1}(x)\right) = \mathrm{Tr}(x),$$

pour tout $x \in \mathbb{F}_{2^n}$.

Exemple 3.3.4. Prenons un entier n impair. Notons tout d'abord que

$$\frac{1}{2^{(n+1)/2}+1} \equiv 2^{(n+1)/2} - 1 \pmod{2^n - 1}.$$

Les fonctions

$$x \mapsto x^{2^{(n+1)/2}-1} + \mathrm{Tr}(x), \tag{3.9}$$

et

$$x \mapsto x^{2^{(n+1)/2}-1} + \mathrm{Tr}(x + x^{2^{(n+1)/2}-1}) \tag{3.10}$$

sont respectivement 2-to-1 et bijectives sur $\mathbb{F}_{2^n}$. De plus, elles sont aussi AB et différentiellement 4-uniformes respectivement.

3.3.2. La fonction inverse $x \mapsto x^{-1}$

Dans cette sous-section, nous nous intéressons à une sous-classe des fonctions sur $\mathbb{F}_{2^n}$:

$$F : x \mapsto x^{-1} + \gamma \mathrm{Tr}\left(H(x)\right), \quad \gamma \in \mathbb{F}_{2^n}^*, \quad \text{et où } H \text{ est une fonction quelconque.} \tag{3.11}$$

Nous nous intéresserons plus particulièrement au cas où la fonction H est une fonction monomiale, c'est-à-dire aux fonctions creuses sur $\mathbb{F}_{2^n}$ de la forme :

$$F_{-1,t,\gamma} : x \mapsto x^{-1} + \gamma \mathrm{Tr}(x^t), \quad \gamma \in \mathbb{F}_{2^n}^*, \quad \text{et où } t \text{ est un entier.} \tag{3.12}$$

Tout d'abord, nous allons voir quelques résultats concernant les fonctions F définies par (3.11). Nous pouvons écrire les dérivées de la fonction F comme suit :

$$\Delta_\alpha F : x \mapsto f_\alpha(x) + \gamma \mathrm{Tr}\left((\Delta_\alpha H)(x)\right), \quad \text{où} \quad f_\alpha : x \mapsto x^{-1} + (x+\alpha)^{-1}, \tag{3.13}$$

pour n'importe quel $\alpha \in \mathbb{F}_{2^n}^*$. Pour calculer l'uniformité différentielle de la fonction F, $\delta(F)$, nous devons calculer le nombre de solutions x des équations sur $\mathbb{F}_{2^n}$

$$E(\alpha, \beta): \qquad (\Delta_\alpha F)(x) = \beta, \qquad \alpha \in \mathbb{F}_{2^n}^*,\ \beta \in \mathbb{F}_{2^n}. \tag{3.14}$$

Notons que si x est une solution de $E(\alpha, \beta)$, alors $f_\alpha(x) \in \{\beta, \beta + \gamma\}$. De plus, nous pouvons constater quelques faits marquants :

($p1$) $x \in \{0, \alpha\}$ est solution de $E(\alpha, \beta)$ si et seulement si

$$\beta = \frac{1}{\alpha} \qquad (\text{avec } \mathrm{Tr}\left((\Delta_\alpha H)(0)\right) = 0)$$

ou bien

$$\beta = \gamma + \frac{1}{\alpha} \qquad (\text{avec } \mathrm{Tr}\left((\Delta_\alpha H)\,(0) = 1\right).$$

($p2$) si $x \notin \{0, \alpha\}$ est solution de l'équation $E(\alpha, \beta)$ alors $f_\alpha(x) = \alpha(x^2 + \alpha x)^{-1}$.

($p3$) x et $x + \alpha$ sont des solutions de $E(\alpha, \beta)$, avec $x \notin \{0, \alpha\}$, si et seulement si x vérifie au moins une des deux conditions suivantes :

$$x^2 + \alpha x + \frac{\alpha}{\beta} = 0 \qquad (\text{avec } \mathrm{Tr}\left((\Delta_\alpha H)(0)\right) = 0), \tag{3.15}$$

$$x^2 + \alpha x + \frac{\alpha}{\beta + \gamma} = 0 \qquad (\text{avec } \mathrm{Tr}\left((\Delta_\alpha H)(0)\right) = 1). \tag{3.16}$$

En effet, nous pouvons retrouver ce résultat en écrivant simplement l'expression $\alpha(x^2 + \alpha x)^{-1} = \beta'$, avec $\beta' \in \{\beta, \beta + \gamma\}$. Nous notons que pour de tels éléments β', l'équation $x^2 + \alpha x + \alpha/\beta' = 0$ possède exactement deux solutions si et seulement si $\mathrm{Tr}(1/\alpha\beta') = 0$ (voir Théorème 3.4.3 à la section 3.4 ou bien [140, Théorème 6.3]).

($p4$) Nous pouvons toujours choisir un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que

$$\mathrm{Tr}\left(\frac{1}{\alpha\beta}\right) = 0 \qquad \text{pour } \beta = \frac{1}{\alpha} + \gamma \quad \text{ et pour n'importe quel } \gamma \in \mathbb{F}_{2^n},$$

puisque la fonction $x \mapsto 1/(1 + \gamma x)$ est bijective sur $\mathbb{F}_{2^n}^*$. De plus si $\beta = 1/\alpha$ alors $\mathrm{Tr}(1/(\alpha\beta)) = \mathrm{Tr}(1)$.
Donc, lorsque n est pair, il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que $\mathrm{Tr}(1/(\alpha\beta)) = 0$ pour $\beta = 1/\alpha$ ainsi que pour $\beta = \gamma + 1/\alpha$.

Nous remarquons que nous venons de prouver implicitement la propriété bien connue (voir [166]) que la fonction inverse vérifie

$$\delta(x \mapsto x^{-1}) = \begin{cases} 2 & \text{lorsque } n \text{ est impair,} \\ 4 & \text{lorsque } n \text{ est pair.} \end{cases}$$

Cependant, avec les observations que nous venons d'effectuer ci-dessus, nous pouvons donner le résultat suivant concernant l'uniformité différentielle des fonctions définies par (3.11).

Lemme 3.3.5. *Soit la fonction sur $\mathbb{F}_{2^n}$ définie par $F : x \mapsto x^{-1} + \gamma Tr(H(x))$ où H est une fonction sur $\mathbb{F}_{2^n}$ et $\gamma \in \mathbb{F}_{2^n}^*$. Alors*

$$\delta(F) = \begin{cases} \{2,4\} & \text{lorsque } n \text{ est impair,} \\ \{4,6\} & \text{lorsque } n \text{ est pair.} \end{cases}$$

Démonstration. Le cas où n est impair provient directement de la proposition 3.1.5 et du fait que la fonction inverse est APN dans ce cas là. Remarquons de plus que si la fonction H est linéaire sur $\mathbb{F}_2$, alors la fonction F est APN.

Supposons maintenant que n est pair et choisissons un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que $\mathrm{Tr}(1/(1+\gamma\alpha)) = 0$ (voir $(p4)$). Conformément aux propositions $(p1)$ et $(p3)$ listées ci-dessus, si $\beta \notin \{1/\alpha, \gamma + 1/\alpha\}$ alors l'équation $E(\alpha, \beta)$ a au plus quatre solutions. Nous allons donc prouver que l'équation $E(\alpha, 1/\alpha)$, respectivement $E(\alpha, \gamma + 1/\alpha)$, a au moins quatre solutions.

Supposons premièrement que l'équation $E(\alpha, 1/\alpha)$ est vérifiée pour $x \in \{0, \alpha\}$, c'est à dire que $\mathrm{Tr}\left((\Delta_\alpha H)(0)\right) = 0$. Cette même équation possède deux autres solutions si et seulement si une des relations (3.15) ou (3.16) est vérifiée pour $x \notin \{0, \alpha\}$. Puisque nous avons le choix de α, nous avons que les équations

$$f_a(x) = \frac{1}{\alpha} \qquad \text{et} \qquad f_a(x) = \frac{1}{\alpha} + \gamma,$$

possèdent toutes les deux, deux solutions chacune sur $\mathbb{F}_{2^n}$: y, $y + \alpha$ et z, $z + \alpha$ respectivement.

Si $\mathrm{Tr}\left((\Delta_\alpha H)(y)\right) = 0$ alors y est une solution de l'équation $E(\alpha, 1/\alpha)$, sinon y est une solution de l'équation $E(\alpha, \gamma + 1/\alpha)$. De la même manière, si $\mathrm{Tr}\left((\Delta_\alpha H)(z)\right) = 1$ alors z est une solution de l'équation $E(\alpha, 1/\alpha)$, sinon z est une solution de l'équation $E(\alpha, \gamma 1/\alpha)$. Cela revient exactement à dire que si l'équation $E(\alpha, 1/\alpha)$ n'a pas de solution x, $x \notin \{0, \alpha\}$ alors l'équation $E(\alpha, \gamma + 1/\alpha)$ a quatre solutions : y, z, $y + \alpha$ et $z + \alpha$.

Si $\mathrm{Tr}\left((\Delta_\alpha H)(0)\right) = 1$ alors l'équation $E(\alpha, \gamma + 1/\alpha)$ est vérifiée pour $x \in \{0, \alpha\}$. De la même manière, nous prouvons que si l'équation $E(\alpha, \gamma + 1/\alpha)$ n'a pas de solution x, $x \notin \{0, \alpha\}$, alors l'équation $E(\alpha, 1/\alpha)$ a quatre solutions sur $\mathbb{F}_{2^n}$. Nous en concluons que $\delta(F) \geq 4$.

Par ailleurs, il est clair que $\delta(F) \neq 8$ lorsque n est pair, puisque l'équation $E(\alpha, \beta)$ ne peut avoir plus de quatre solutions uniquement si 0 et α sont des solutions. Dans ce cas là, $\beta \in \{1/\alpha, \gamma + 1/\alpha\}$, et pour chaque élément β de cette forme, le nombre de solutions est 4 ou 6. □

Nous venons de voir quelles peuvent être les valeurs de l'uniformité différentielle des fonctions définies par (3.11). Il est donc très intéressant d'étudier ces fonctions pour des classes spécifiques de fonctions H. Dans [190], Yin Tan, Longjiang Qu, Chik How Tan et Chao Li sont intéressés par les fonctions $H : x \mapsto x^2/(x+1)$. Ils obtiennent des permutations différentiellement 4-uniformes sur $\mathbb{F}_{2^n}$ lorsque n est pair. Comme nous nous intéressons ici aux fonctions creuses obtenues à partir de deux monômes, la question que nous nous posons naturellement est la suivante.

Problème 3.3.6. Est-il possible de construire des fonctions $F_{-1,t,\gamma}$ du type (3.12) bijectives sur $\mathbb{F}_{2^n}$ avec $\delta(F_{-1,t,\gamma}) = 4$?

Il est assez facile de construire des permutations à partir de la fonction inverse (du type (3.12)) qui ont une uniformité différentielle faible, d'après le lemme précédent et le corollaire 3.2.5.

Proposition 3.3.7. *Soit un entier $1 \leq i < n$, $i \neq n/2$ et soit $\gamma \in \mathbb{F}_{2^n}^*$. Alors la fonction sur $\mathbb{F}_{2^n}$*

$$F_{-1,i,\gamma} : x \mapsto x^{-1} + \gamma\, Tr(x^{2^{n-1}-2^{i-1}-1}) \tag{3.17}$$

est bijective lorsque

$$\gamma \in \mathbb{F}_{2^{\gcd(2i,n)}} \qquad \textit{tel que} \qquad Tr(\gamma^{2^i+1}) = 0.$$

Si $Tr(\gamma^{2^i+1}) = 1$, alors la fonction $F_{-1,i,\gamma}$ est 2-to-1. Dans les deux cas,

$$\delta(F_{-1,i,\gamma}) = \begin{cases} \{2,4\} & \textit{lorsque } n \textit{ est impair,} \\ \{4,6\} & \textit{lorsque } n \textit{ est pair.} \end{cases}$$

Démonstration. Nous remarquons une nouvelle fois que $-1 \equiv 2^n - 2 \pmod{2^n-1}$ et que

$$\begin{aligned}(2^n-2)(2^i+1) &\equiv 2^i + 2^n - 2^{i+1} - 2 \pmod{2^n-1} \\ &\equiv 2^n - 2^i - 2 \pmod{2^n-1} \\ &\equiv 2(2^{n-1} - 2^{i-1} - 1) \pmod{2^n-1}.\end{aligned}$$

Nous pouvons donc appliquer le corollaire 3.2.5. Les bornes sur $\delta(F_{-1,i,\gamma})$ sont obtenues directement à partir du lemme 3.3.5. □

Nous regardons maintenant un cas particulier des fonctions du type (3.17), lorsque $i = 1$. Notons que dans ce cas

$$\mathrm{Tr}(x^{2^{n-1}-2^{i-1}-1}) = \mathrm{Tr}(x^{2^{n-2}-1}) = \mathrm{Tr}(x^{-3}),$$

pour tout $x \in \mathbb{F}_{2^n}$. Le lemme suivant résume les conditions nécessaires et suffisantes pour construire des fonctions bijectives et 2-to-1 de la forme $F_{-1,-3,\gamma}$ ayant une uniformité différentielle basse.

Lemme 3.3.8. *Soit la fonction $F_{-1,-3,1} : x \mapsto x^{-1} + Tr(x^{-3})$ sur $\mathbb{F}_{2^n}$. Nous avons alors que :*

- *Si n est pair, alors la fonction $F_{-1,-3,1}$ est bijective sur $\mathbb{F}_{2^n}$.*
  Elle vérifie $\delta(F_{-1,-3,1}) = 6$ si et seulement si il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que

$$Tr(\alpha^{-3}) = 0, \qquad Tr((\alpha+1)^{-1}) = 0 \qquad \textit{et } Tr(\alpha^{-1}) = 1\ ;$$

- *Si n est impair, alors la fonction $F_{-1,-3,1}$ est 2-to-1 sur $\mathbb{F}_{2^n}$.*
  Elle vérifie $\delta(F_{-1,-3,1}) = 4$ si et seulement si il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que

$$Tr(\alpha^{-3}) = 0, \qquad Tr((\alpha+1)^{-1}) = 0 \qquad \textit{et } Tr(\alpha^{-1}) = 1.$$

De plus, pour tout $n \geq 38$, il existe toujours au moins un élément $\alpha \in \mathbb{F}_{2^n}^*$ vérifiant ces conditions.

La preuve étant longue et technique, nous l'ajoutons à l'annexe A.1.1. Nous notons tout de même que la condition sur n est due en grande partie à un résultat de Stephen D. Cohen [76] sur les éléments des corps finis avec des traces prescrites.

Remarque 3.3.9. Nous pouvons compléter le lemme précédent en utilisant le théorème 3.2.12 : lorsque n est impair, la fonction $x \mapsto x^{-1} + \mathrm{Tr}(x^{-3} + x^{-1})$ est bijective sur $\mathbb{F}_{2^n}$ et possède une uniformité différentielle *au plus* quatre.

Comme nous l'avons précédemment noté, il est difficile d'obtenir des permutations creuses différentiellement 4-uniformes lorsque n est pair. Cependant, il existe des fonctions $F_{-1,t,\gamma}$ sur $\mathbb{F}_{2^n}$, n pair, qui ne sont pas des permutations mais qui possèdent la même uniformité différentielle que la fonction inverse $x \mapsto x^{-1}$.

Théorème 3.3.10. *Soit un entier positif n pair et un élément $\gamma \in \mathbb{F}_{2^n}^*$. Alors les fonctions sur $\mathbb{F}_{2^n}$*

$$F_{-1,t,\gamma} : x \mapsto x^{-1} + \gamma\, Tr(x^t), \qquad \text{avec } t \in \{3, 5\},$$

vérifient $\delta(F_{-1,t,\gamma}) = 4$ pour n'importe quel élément γ tel que $\gamma^t = 1$, $t \in \{3, 5\}$.

De même que pour le lemme 3.3.8, nous ajoutons la preuve de ce théorème à l'annexe A.1.2.

Une question qu'il est alors naturel de se poser est la suivante :

Problème 3.3.11. Existe-t-il d'autres valeurs entières t pour lesquels les fonctions $F_{-1,t,\gamma}$ définies par (3.12) vérifiant $\delta(F_{-1,t,\gamma}) = 4$ pour certains éléments $\gamma \in \mathbb{F}_{2^n}^*$?

3.4. Avec deux exposants quadratiques

Dans cette sous-section, nous étudions les fonctions sur $\mathbb{F}_{2^n}$ de la forme

$$F_{2^j+1,2^k+1,\gamma} : x \mapsto x^{2^j+1} + \gamma \mathrm{Tr}(x^{2^k+1}), \qquad \text{avec } \gcd(j, n) = 1,\ \gamma \in \mathbb{F}_{2^n}^*, \tag{3.18}$$

où j et k sont des entiers non nuls. D'abord, on déduit des propositions 3.2.7 et 3.1.5 le corollaire suivant.

Corollaire 3.4.1. *Soit une fonction $F_{2^j+1,2^k+1,\gamma} : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ définie par* (3.18)*. Alors,*

(i) $\delta(F_{2^j+1,2^k+1,\gamma}) \leq 4$ *;*

(ii) la fonction $F_{2^j+1,2^k+1,\gamma}$ ne peut être bijective sur $\mathbb{F}_{2^n}$ à moins que $j = k$ et $Tr(\gamma) = 0$.

Démonstration. La condition sur j, $\gcd(j, n) = 1$, signifie que la fonction $x \mapsto x^{2^j+1}$ est APN sur $\mathbb{F}_{2^n}$. Donc, (i) vient directement de la proposition 3.1.5. La proposition 3.2.7 quant à elle implique (ii) lorsque n est pair.

Lorsque n est impair, la fonction $F_{2^j+1,2^k+1,\gamma}$ est bijective sur $\mathbb{F}_{2^n}$ seulement si

$$2^k + 1 \equiv 2^\ell(2^j + 1)(2^i + 1) \pmod{2^n - 1}, \qquad \text{pour des entiers } i,\ \ell,$$

d'après le théorème 3.2.3. Or cela n'est possible que si nous avons l'un des deux cas suivants :

- $k = j$, $i = 0$. Lorsque $\mathrm{Tr}(\gamma) = 0$, nous avons des permutations d'après le théorème 3.2.3 (a).
- $i = j = 1$. Ce cas est déjà traité par la proposition 3.2.8.

□

3.4.1. Les dérivées

Le degré algébrique des fonctions définies par (3.18) est clairement $\deg(F_{2^j+1,2^k+1,\gamma}) = 2$. Leurs dérivées seront donc au plus affines (*i.e.* affines ou constantes). Le fait qu'elles soient de degré algébrique ≤ 1 simplifie un petit peu l'étude de ces fonctions.

Nous allons donc étudier dans un premier temps les dérivées de la fonction $F_{2^j+1,2^k+1,\gamma}$ sur $\mathbb{F}_{2^n}$, c'est-à-dire les fonctions $x \mapsto G_\alpha(x) + F_{2^j+1,2^k+1,\gamma}(\alpha)$, $\alpha \in \mathbb{F}_{2^n}^*$, où

$$\begin{aligned} &G_\alpha : x \mapsto g_\alpha(x) + \gamma \mathrm{Tr}(h_\alpha(x)), \\ &g_\alpha : x \mapsto x^{2^j}\alpha + x\alpha^{2^j}, \qquad h_\alpha : x \mapsto x^{2^k}\alpha + x\alpha^{2^k}. \end{aligned} \tag{3.19}$$

Tout comme la fonction G_α, les fonctions g_α et h_α sont linéaires. Il en découle que la fonction G_α est 2-to-1 si et seulement si son noyau est de dimension 1, c'est à dire que son polynôme associé ne possède qu'une seule racine non nulle. Si toutes les fonctions G_α, pour tous les éléments $\alpha \in \mathbb{F}_{2^n}^*$, sont 2-to-1, alors la fonction $F_{2^j+1,2^k+1,\gamma}$ est APN.

Nous allons donc étudier les racines des polynômes $G_\alpha(x) \in \mathbb{F}_{2^n}[x]$ du type (3.19).

Puisque nous avons choisi les entiers j et n premiers entre eux (voir formule (3.18)), la fonction $x \mapsto x^{2^j+1}$ est APN et la fonction g_α est donc 2-to-1 pour tout élément $\alpha \in \mathbb{F}_{2^n}^*$. L'image de la fonction g_α, pour un certain élément non nul α, forme donc un hyperplan[1] de $\mathbb{F}_{2^n}$ (voir définition 1.3.10), que nous notons $\mathcal{H}_\alpha$. Déterminons maintenant l'élément $\lambda \in \mathbb{F}_{2^n}$ définissant l'hyperplan $\mathcal{H}_\alpha$. Nous devons avoir :

$$\mathrm{Tr}(\lambda(x^{2^j}\alpha + x\alpha^{2^j})) = \mathrm{Tr}(x(\lambda\alpha^{2^j} + (\lambda\alpha)^{2^{n-j}})) = 0,$$

pour tout $x \in \mathbb{F}_{2^n}$. D'où $\lambda\alpha = \lambda^{2^j}\alpha^{2^{2j}}$ impliquant que $\lambda = \alpha^{(-2^j+1)}$, et l'hyperplan $\mathcal{H}_\alpha$ est donc

$$\mathcal{H}_\alpha = \left\{ y \in \mathbb{F}_{2^n} \mid \mathrm{Tr}\left(\frac{y}{\alpha^{2^j+1}}\right) = 0 \right\}.$$

Le théorème suivant est une adaptation au cas binaire d'un résultat dû à Pascale Charpin et Gohar Kyureghyan [66, Théorème 6].

Théorème 3.4.2. *Soit une fonction $L : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, $\mathbb{F}_2$-linéaire, dont le noyau est l'ensemble des éléments $\{0, \alpha\} \subset \mathbb{F}_{2^n}$ et soient des éléments γ, $\beta \in \mathbb{F}_{2^n}^*$ avec $Tr(\beta\alpha) = 0$. Alors la fonction sur $\mathbb{F}_{2^n}$, $\mathbb{F}_2$-linéaire,*

$$x \mapsto L(x) + \gamma\, Tr(\beta x),$$

a un noyau de dimension au plus 2. De plus, son noyau est de dimension exactement 1 si et seulement si l'une des deux conditions suivantes est vérifiée :

(i) il existe un élément $\mu \in \mathbb{F}_{2^n}$ vérifiant $L(\mu) = \gamma$ et $Tr(\beta\mu) = 0$;

(ii) γ n'appartient pas à l'image de la fonction L.

Nous avons vu que les dérivées des fonctions du type (3.18), construites à partir de deux exposants quadratiques, sont construites à partir de deux fonctions binomiales $\mathbb{F}_2$-linéaires. Avant de continuer plus loin l'étude de l'uniformité différentielle des fonctions définies par (3.18), nous allons réaliser un *a parte* concernant les fonctions binomiales $\mathbb{F}_2$-linéaires. Nous rappelons des résultats classiques nous permettant de trouver les racines de tels polynômes et en profitons pour spécifier une classe de fonctions bijectives.

[1] cela ne peut être le complémentaire d'un hyperplan puisque 0 est racine de $g_\alpha(x)$

Les fonctions binomiales $\mathbb{F}_2$-linéaires

Nous commençons tout d'abord par rappeler le théorème connu sous le nom de *théorème 90 d'Hilbert*, dont une preuve peut être trouvée dans le livre de Serge Lang : "Algebra" [140, Théorème 6.3].

Théorème 3.4.3 (Théorème 90 de Hilbert)**.** *Soit une fonction génératrice du groupe d'automorphismes de l'extension galoisienne $\mathbb{F}_{2^n}/\mathbb{F}_{2^m}$, notée σ, et soit un élément $\alpha \in \mathbb{F}_{2^n}$. Alors*

$$Tr_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha) = 0 \qquad \Leftrightarrow \qquad \exists \beta \in \mathbb{F}_{2^n} \quad \text{tel que } \alpha = \beta + \sigma(\beta).$$

En particulier, soit $m = \gcd(\ell, n)$, une équation de la forme

$$x^{2^\ell} + x + \alpha = 0, \qquad 0 < \ell < n, \tag{3.20}$$

admet une solution sur $\mathbb{F}_{2^n}$ si et seulement si $\mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha) = 0$. Il existe de plus une méthode constructive pour obtenir les solutions de l'équation (3.20). Pour les caractériser, nous avons besoin d'introduire le polynôme linéarisé suivant.

Définition 3.4.4. Soient des entiers positifs $0 < \ell < n$, soit leur pgcd $m = \gcd(\ell, n)$ et soit un élément $\mu \in \mathbb{F}_{2^n}$. Nous définissons la fonction $\mathbb{F}_{2^m}$-linéaire $L_{\ell,\mu}$ par

$$\begin{array}{rccl} L_{\ell,\mu} & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & \displaystyle\sum_{i=0}^{n/m-1} x^{2^{i\ell}} \sum_{k=0}^{i} \mu^{2^{k\ell}}. \end{array}$$

Exemple 3.4.5. Soient $n = 4$ et $\ell = 3$, et soit $\mu \in \mathbb{F}_{2^4}^*$. Alors, pour tout $x \in \mathbb{F}_{2^4}$, nous avons

$$\begin{aligned} L_{3,\mu}(x) &= \mu x + (\mu + \mu^{2^3})x^{2^3} + (\mu + \mu^{2^3} + \mu^{2^6})x^{2^6} + (\mu + \mu^{2^3} + \mu^{2^6} + \mu^{2^9})x^{2^9} \\ &\equiv \mu x + (\mu + \mu^{2^3})x^{2^3} + (\mu + \mu^{2^3} + \mu^{2^2})x^{2^2} + (\mu + \mu^{2^3} + \mu^{2^2} + \mu^2)x^2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\mathrm{mod}\ x^{2^n} + x). \end{aligned}$$

Lemme 3.4.6. *Soient des entiers positifs $0 < \ell < n$, soit leur pgcd $m = \gcd(\ell, n)$ et soit un élément $\mu \in \mathbb{F}_{2^n}$. Alors, pour tout $x \in \mathbb{F}_{2^n}$, nous avons*

$$(L_{\ell,\mu}(x))^{2^\ell} + L_{\ell,\mu}(x) = \mu\, Tr_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(x) + Tr_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\mu)x. \tag{3.21}$$

Démonstration. Nous avons que

$$L_{\ell,\mu}(x) = \mu x + (\mu + \mu^{2^\ell})x^{2^\ell} + \cdots + (\mu + \cdots + \mu^{2^{(n/m-1)\ell}})x^{2^{(n/m-1)\ell}},$$

et

$$(L_{\ell,\mu}(x))^{2^\ell} = \mu^{2^\ell} x^{2^\ell} + \cdots + (\mu^{2^\ell} + \cdots + \mu^{2^{(n/m)\ell}})x,$$

pour tout $x \in \mathbb{F}_{2^n}$. Donc, en simplifiant nous obtenons que

$$\begin{aligned} (L_{\ell,\mu}(x))^{2^\ell} + L_{\ell,\mu}(x) &= \mu(x + x^{2^\ell} + \cdots + x^{2^{(n/m-1)\ell}}) + (\mu + \cdots + \mu^{2^{(n/m-1)\ell}})x \\ &= \mu \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(x) + \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\mu)x, \end{aligned}$$

car n est premier avec ℓ/m. □

À partir du polynôme défini à la définition 3.4.4, nous pouvons maintenant caractériser les solutions de l'équation 3.20.

Théorème 3.4.7. *Soient des entiers positifs $0 < \ell < n$, soit leur pgcd $m = \gcd(\ell, n)$ et soit un élément $\alpha \in \mathbb{F}_{2^n}$. Alors les solutions dans $\mathbb{F}_{2^n}$ de l'équation*

$$x^{2^\ell} + x + \alpha = 0 \quad \text{où} \quad Tr_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha) = 0,$$

sont données par l'ensemble

$$S := \left\{ \frac{1}{Tr_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\mu)} L_{\ell,\mu}(\alpha) + \omega \mid \omega \in \mathbb{F}_{2^m} \right\},$$

où $\mu \in \mathbb{F}_{2^n}^*$ est un élément quelconque satisfaisant $Tr_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\mu) = 1$.

Démonstration. Un élément $L_{\ell,\mu}(\alpha) + \omega \in S$, avec $\omega \in \mathbb{F}_{2^m}$ vérifie :

$$\begin{aligned}(L_{\ell,\mu}(\alpha) + \omega)^{2^\ell} + L_{\ell,\mu}(\alpha) + \omega &= (L_{\ell,\mu}(\alpha))^{2^\ell} + L_{\ell,\mu}(\alpha) \\ &= \mu \mathrm{Tr}_{\mathbb{F}_{2^n}/\mathbb{F}_{2^m}}(\alpha) + \alpha \\ &= 0 + \alpha.\end{aligned}$$

De plus, nous avons que

$$x^{2^\ell} + x = 0 \quad \Leftrightarrow \quad x \in \mathbb{F}_{2^\ell} \cap \mathbb{F}_{2^n} \simeq \mathbb{F}_{2^m},$$

ce qui implique que le nombre de solutions de l'équation $x^{2^\ell} + x + \alpha = 0$ est exactement 2^m, concluant ainsi la preuve. □

Nous utilisons les résultats précédents pour en déduire une classe de fonctions creuses, $\mathbb{F}_2$-linéaires bijectives, qui est un cas particulier du théorème 3.4.2.

Théorème 3.4.8. *Soient des entiers positifs $0 < \ell < n$ premiers entre eux et soit un élément $\mu \in \mathbb{F}_{2^n}$ tel que $Tr(\mu) = 1$. Alors la fonction*

$$\begin{array}{rccl} F & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^{2^\ell} + x + \mu\, Tr(\mu x) \end{array}$$

est bijective sur $\mathbb{F}_{2^n}$. De plus, son inverse pour la composition de fonction est

$$\begin{array}{rccl} F^{-1} & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & L_{\ell,\mu}(x) + \left(1 + L_{\ell,\mu}^*(\mu)\right) Tr(x). \end{array}$$

où $L_{\ell,\mu}^*$ est le polynôme adjoint de $L_{\ell,\mu}$, défini à la définition 1.3.14.

Démonstration. Nous avons vu au théorème 3.4.3, que l'image de la fonction $x \mapsto x^{2^\ell} + x$ est un ensemble dont les éléments $\alpha \in \mathbb{F}_{2^n}$ vérifient $\mathrm{Tr}(\alpha) = 0$. En d'autres termes, l'image de la fonction $x \mapsto x^{2^\ell} + x$ est l'hyperplan H_1 (voir définition 1.3.10).

Nous connaissons donc l'image de la fonction F par l'hyperplan H_μ et son complémentaire :

$$F(H_\mu) = H_1 \qquad \text{et} \qquad F(\overline{H_\mu}) = \overline{H_1}.$$

Nous en déduisons que l'image de la fonction F sur $\mathbb{F}_{2^n}$ est exactement $\mathbb{F}_{2^n}$.

Montrons maintenant que $L^*_{\ell,\mu}(\mu) \in \mathbb{F}_2$. En effet, pour tout $x \in \mathbb{F}_{2^n}$,

$$\begin{aligned}
L^*_{\ell,\mu}(x) &= \mu x + \left(\mu + \cdots + \mu^{2^{(n-1)\ell}}\right)^{2^\ell} x^{2^\ell} + \cdots + \left(\mu + \mu^{2^\ell}\right)^{2^{(n-1)\ell}} x^{2^{(n-1)\ell}} \\
&= \mu x + \mathrm{Tr}(\mu) x^{2^\ell} + \cdots + \left(\mathrm{Tr}(\mu) + \mu^{2^\ell} + \cdots + \mu^{2^{(n-1)\ell}}\right) x^{2^{(n-1)\ell}} \\
&= \mu x + \mu^{2^\ell} x^{2^{2\ell}} + \cdots + \left(\mu^{2^\ell} + \cdots + \mu^{2^{(n-1)\ell}}\right) x^{2^{(n-1)\ell}} + \mathrm{Tr}(\mu)\left(\mathrm{Tr}(x) + x\right) \\
&= \mu x + \left(L_{\ell,\mu}(x^{2^\ell})\right)^{2^\ell} + x^{2^\ell} + \mathrm{Tr}(x) + x + \left(\mathrm{Tr}(\mu) + \mu\right) x \\
&= \mathrm{Tr}(x) + x^{2^\ell} + \left(L_{\ell,\mu}(x^{2^\ell})\right)^{2^\ell}. \qquad (3.22)
\end{aligned}$$

Or, d'après le Lemme 3.4.6, $x^{2^\ell} + \left(L_{\ell,\mu}(x^{2^\ell})\right)^{2^\ell} = L(x^{2^\ell}) + \mu \mathrm{Tr}(x^{2^\ell})$ d'où

$$L^*_{\ell,\mu}(x) = \mathrm{Tr}(x) + x^{2^\ell} + \left(L_{\ell,\mu}(x^{2^\ell})\right)^{2^\ell} = \mathrm{Tr}(x) + L_{\ell,\mu}(x^{2^\ell}) + \mu \mathrm{Tr}(x).$$

En évaluant la fonction $L_{\ell,\mu}$ en μ, nous obtenons

$$\begin{aligned}
L^*_{\ell,\mu}(\mu) &= 1 + \mu^{2^\ell} + \left(L_{\ell,\mu}(\mu^{2^\ell})\right)^{2^\ell} = 1 + L_{\ell,\mu}(\mu^{2^\ell}) + \mu \\
&= \left(1 + \mu + L_{\ell,\mu}(\mu^{2^\ell})\right)^{2^\ell} = 1 + L_{\ell,\mu}(\mu^{2^\ell}) + \mu.
\end{aligned}$$

Cela signifie que $\left(L^*_{\ell,\mu}(\mu)\right)^{2^\ell} = L^*_{\ell,\mu}(\mu)$, impliquant $L_{\ell,\mu}(\mu) \in \mathbb{F}_2$ (car $\gcd(\ell, n) = 1$).

Nous pouvons donc conclure cette preuve en calculant $F \circ G(x)$ pour tout $x \in \mathbb{F}_{2^n}$ avec $G : x \mapsto L_{\ell,\mu}(x) + \left(1 + L^*_{\ell,\mu}(\mu)\right) \mathrm{Tr}(x)$:

$$\begin{aligned}
F\left(G(x)\right) &= \left(L_{\ell,\mu}(x)\right)^{2^\ell} + L_{\ell,\mu}(x) + \mu\left[\mathrm{Tr}\left(\mu L_{\ell,\mu}(x)\right) + \mathrm{Tr}\left(\mu\left(1 + L^*_{\ell,\mu}(\mu)\right)\right)\mathrm{Tr}(x)\right] \\
&= x + \mu \mathrm{Tr}(x) + \mu\left[L^*_{\ell,\mu}(\mu)\mathrm{Tr}(x) + \mathrm{Tr}(\mu)\mathrm{Tr}(x) + L^*_{\ell,\mu}(\mu)\mathrm{Tr}(\mu)\mathrm{Tr}(x)\right] \\
&= x + \mu \mathrm{Tr}(x) + \mu \mathrm{Tr}(x) \\
&= x.
\end{aligned}$$

□

Caractérisation des dérivées G_α qui sont des fonctions 2-to-1

Nous pouvons spécifier un peu plus le théorème 3.4.2, pour qu'il coïncide avec les fonctions creuses sur $\mathbb{F}_{2^n}$ du type (3.18). Il en résulte le corollaire suivant.

Corollaire 3.4.9. *Soient deux fonctions de $\mathbb{F}_{2^n}$, $F_{2^j+1,2^k+1,\gamma}$ donnée par* (3.18)*, et G_α donnée par* (3.19)*. Alors la fonction G_α est 2-to-1 si et seulement si l'une des deux conditions suivantes est vérifiée :*

(*i*) *il existe un élément $\mu \in \mathbb{F}_{2^n}$ tel que $\gamma = g_\alpha(\mu)$ et $Tr(h_\alpha(\mu)) = 0$;*

(*ii*) *γ n'appartient pas à l'image de la fonction g_α, c'est-à-dire $Tr(\gamma/\alpha^{2^j+1}) = 1$.*

Si la fonction G_α n'est pas 2-to-1, son noyau est de dimension 2.

Démonstration. Nous appliquons le théorème 3.4.2 avec $L = g_\alpha$ et $\beta = \alpha^{2^{n-k}} + \alpha^{2^k}$ car

$$\mathrm{Tr}(h_\alpha(x)) = \mathrm{Tr}(\alpha x^{2^k} + x\alpha^{2^k}) = \mathrm{Tr}(x(\alpha^{2^{n-k}} + \alpha^{2^k})),$$

pour tout $x \in \mathbb{F}_{2^n}$. Comme la fonction g_α est 2-to-1, son noyau est l'ensemble $\{0, \alpha\}$. Notons que $\mathrm{Tr}(\beta\alpha) = 0$ car

$$\mathrm{Tr}((\alpha^{2^{n-k}} + \alpha^{2^k})\alpha) = \mathrm{Tr}(\alpha^{2^{n-k}+1} + \alpha^{2^k+1}) = 0.$$

Nous terminons cette démonstration en appliquant directement le théorème 3.4.2. □

Remarquons que le fait que γ soit dans l'image de la fonction g_α signifie que $\gamma \in \mathcal{H}_\alpha$, et donc que $\mathrm{Tr}(\gamma/\alpha^{2^j+1}) = 0$. Nous savons aussi que $\delta(F_{2^j+1,2^k+1,\gamma}) \in \{2, 4\}$. Nous pouvons donc en déduire directement le corollaire suivant.

Corollaire 3.4.10. *Soit une fonction $F_{2^j+1,2^k+1,\gamma} : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ donnée par* (3.18) *avec $k > 0$. Alors la fonction $F_{2^j+1,2^k+1,\gamma}$ est APN si et seulement si pour tout élément $\alpha \in \mathbb{F}_{2^n}^*$ nous avons*

$$Tr\left(\frac{\gamma}{\alpha^{2^j+1}}\right) = 0 \qquad \Rightarrow \qquad Tr(\alpha\mu^{2^k} + \mu\alpha^{2^k}) = 0, \quad où \quad \alpha\mu^{2^j} + \mu\alpha^{2^j} = \gamma. \quad (3.23)$$

Sinon $\delta(F_{2^j+1,2^k+1,\gamma}) = 4$.

Remarque 3.4.11. Ce corollaire montre que lorsque $j = k$ avec $\gcd(j, n) = 1$, on obtient une fonction $F_{2^j+1,2^j+1,\gamma}$ APN sur $\mathbb{F}_{2^n}$ pour n'importe quel élément γ vérifiant $\mathrm{Tr}(\gamma) = 0$. De plus, une telle fonction $F_{2^j+1,2^j+1,\gamma}$ est bijective sur $\mathbb{F}_{2^n}$ lorsque n est impair d'après le corollaire 3.2.4.

Cela est dû au fait que la fonction $F_{2^j+1,2^j+1,\gamma}$ est EA-équivalente à la fonction quadratique $x \mapsto x^{2^j+1}$.

Corollaire 3.4.12. *Soit la fonction $F_{2^j+1,2^k+1,\gamma} : x \mapsto x^{2^j+1} + \gamma\, Tr(x^{2^k+1})$ sur $\mathbb{F}_{2^n}$ avec $\gcd(j, n) = 1$ et $k > 0$. Soit un élément $\mu \in \mathbb{F}_{2^n}^*$ tel que $Tr(\mu) = 1$. Alors la fonction $F_{2^j+1,2^k+1,\gamma}$ est APN si et seulement si pour tout élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que $Tr(\gamma/\alpha^{2^j+1}) = 0$, nous avons*

$$Tr\left(\alpha^{2^k+1}\left((L_{j,\mu}(A))^{2^k} + L_{j,\mu}(A)\right)\right) = 0, \qquad où \quad A = \frac{\gamma}{\alpha^{2^j+1}}, \qquad (3.24)$$

où la fonction linéaire $L_{j,\mu}$ est donnée à la définition 3.4.4.

Démonstration. Pour tout élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que $\mathrm{Tr}(\gamma/\alpha^{2^j+1}) = 0$, il existe exactement deux solutions à l'équation $\alpha x^{2^j} + \alpha^{2^j} x = \gamma$, puisque j et n sont premiers entre eux. Ces solutions peuvent être obtenues en utilisant le polynôme $\mathbb{F}_2$-linéaire $L_{j,\mu}$ comme cela est fait au théorème 3.4.7. Nous pouvons donc appliquer directement le corollaire 3.4.10, et nous obtenons que

$$\mathrm{Tr}\left(\alpha x^{2^k} + \alpha^{2^k} x\right) = \mathrm{Tr}\left(\alpha^{2^k+1}\left(L_{j,\mu}(A))^{2^k} + L_{j,\mu}(A)\right)\right) = 0,$$

où $A = \gamma/\alpha^{2^j+1}$ et $x = L_{j,\mu}(A)\alpha$. □

3.4.2. Un cas particulier

Pour compléter et illustrer les résultats des sous-sections précédentes sur les fonctions $F_{s,t,\gamma}$ construites à partir de deux exposants quadratiques, nous proposons quelques compléments concernant la fonction $x \mapsto x^3 + \gamma \mathrm{Tr}(x^9)$ sur $\mathbb{F}_{2^n}$. Cette fonction est APN sur $\mathbb{F}_{2^n}$ pour tout entier positif n non nul, et pour un élément $\gamma \in \mathbb{F}_{2^n}$ qui convienne.

Nous observons tout d'abord le résultat suivant.

Lemme 3.4.13. *Les notations sont identiques à celles utilisées dans le corollaire 3.4.12. Supposons que $k = \ell j$, $\ell > 1$. Alors, la condition 3.24 devient*

$$Tr\left(\alpha^{2^{\ell j}+1}\left(\left(\frac{\gamma}{\alpha^{2^j+1}}\right)^{2^{(\ell-1)j}} + \cdots + \left(\frac{\gamma}{\alpha^{2^j+1}}\right)^{2^j} + \frac{\gamma}{\alpha^{2^j+1}}\right)\right) = 0. \tag{3.25}$$

Démonstration. Nous utilisons simplement le fait que $(L_{j,\mu}(A))^{2^j} = L_{j,\mu}(A) + A$:

$$(L_{j,\mu}(A))^{2^j} = (L_{j,\mu}(A) + A)^{2^{(\ell-1)j}} = (L_{j,\mu}(A) + A)^{2^{(\ell-2)j}} + A^{2^{(\ell-1)j}} = \ldots$$

□

Nous en déduisons directement la proposition :

Proposition 3.4.14. *Soit la fonction $F : x \mapsto x^3 + \gamma\, Tr(x^9)$. Alors, la fonction F est APN pour tout entier positif n lorsque $\gamma = 1$. Lorsque n est pair, la fonction F est APN pour tout $\gamma \in \mathbb{F}_4$.*

Démonstration. Le résultat vient directement de l'équation (3.25), en remplaçant $j = 1$ et $k = \ell = 3$. Pour tout élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que $\mathrm{Tr}(\gamma/\alpha^3) = 0$, nous devons avoir

$$\mathrm{Tr}\left(\alpha^9\left(\left(\frac{\gamma}{\alpha^3}\right)^{2^2} + \left(\frac{\gamma}{\alpha^3}\right)^2 + \frac{\gamma}{\alpha^3}\right)\right) = \mathrm{Tr}\left(\frac{\gamma^4}{\alpha^3} + \alpha^3\gamma^2 + \alpha^6\gamma\right) = 0.$$

Cela est évidemment vrai lorsque $\gamma = 1$, pour tout entier n. Cependant, lorsque n est pair, cette relation est satisfaite aussi pour $\gamma \in \mathbb{F}_4$. En effet, dans ce cas, nous obtenons

$$\mathrm{Tr}\left(\frac{\gamma}{\alpha^3} + \alpha^6(\gamma^4 + \gamma)\right) = 0.$$

□

Dans le cas où l'entier n est impair, la fonction $x \mapsto x^3 + \mathrm{Tr}(x^9)$ est 2-to-1 sur $\mathbb{F}_{2^n}$. À la section 3.2.2, nous avons montré comment obtenir des permutations à partir de fonctions du type $F_{s,t,\gamma}$ qui sont 2-to-1 sur $\mathbb{F}_{2^n}$.

Proposition 3.4.15. *Soit une entier positif n. La fonction*

$$\begin{array}{rccl} G & : \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ & x & \mapsto & x^3 + Tr(x^9 + x^3) \end{array}$$

est bijective et satisfait $\delta(G) = 4$.

Démonstration. D'après le théorème 3.2.12, la fonction G est bijective telle que $\delta(G) \leq 4$. Calculons la dimension du noyau de la fonction $G_\alpha : x \mapsto \Delta_\alpha(G(x)) + G(\alpha)$ pour tout $\alpha \in \mathbb{F}_{2^n}^*$. Nous considérons l'équation

$$G_\alpha(x) = x^2\alpha + x\alpha^2 + \mathrm{Tr}(x^8\alpha + x\alpha^8 + x^2\alpha + x\alpha^2) = 0.$$

Premièrement, $x = 0$ et $x = \alpha$ sont solutions, quelque soit l'élément $\alpha \in \mathbb{F}_{2^n}^*$. De plus, si $\mathrm{Tr}(\alpha^{-3}) = 0$, alors il existe un élément $\mu \in \mathbb{F}_{2^n}$ tel que $\mu^2\alpha + \mu\alpha^2 = 1$. Puisque la fonction $x \mapsto x^3 + \mathrm{Tr}(x^9)$ est APN, nous avons dans ce cas que $\mathrm{Tr}(\mu^8\alpha + \mu\alpha^8) = 0$ d'après le corollaire 3.4.10. Ainsi μ et $\mu + \alpha$ sont solutions de l'équation

$$x^2\alpha + x\alpha^2 + 1 = 0 \quad \text{avec} \quad \mathrm{Tr}(x^8\alpha + x\alpha^8 + x^2\alpha + x\alpha^2) = 1,$$

impliquant donc que la fonction G_α n'est pas 2-to-1 lorsque $\mathrm{Tr}(\alpha^{-3}) = 0$. □

4. Différentiabilité et intégrabilité

Dans ce chapitre, nous allons aborder un point de vue matriciel des différentielles d'ordre supérieur. Cette façon de voir les choses, en particulier les objets, nous permet d'exprimer une condition nécessaire et suffisante pour qu'une fonction soit une différentielle d'une autre fonction. Nous verrons aussi une méthode d'intégration de fonction dans $\mathbb{F}_{2^n}$. Finalement, nous verrons une méthode de construction de fonctions quadratiques APN.

Il s'agit d'un travail personnel en cours de réalisation, mais les méthodes présentées dans ce chapitre permettent de porter un regard différent sur les différentielles des fonctions sur $\mathbb{F}_{2^n}$. En effet, pour construire ces différentielles, et intégrer, il faut bien souvent manipuler la totalité des coefficients des fonctions dans $\mathbb{F}_{2^n}$. De plus, les outils théoriques utilisés permettent une nouvelle caractérisation des 0-structures linéaires par exemple.

4.1. Rappels d'algèbre linéaire

Nous ne rappelons dans cette section que les éléments d'algèbre linéaire qui nous sont utiles pour la suite de ce chapitre. De très bons ouvrages permettent d'approfondir ses connaissances dans ce domaine [106, 140].

Définition 4.1.1. Un espace vectoriel sur un corps K est un ensemble E muni d'une addition (loi interne) telle que $(E, +)$ est un groupe abélien dont l'élément neutre est 0_E, et d'une multiplication scalaire (loi externe) distributive, associative et dont l'élément neutre est 1_E.

Un sous-espace vectoriel V de E est un sous-ensemble non vide de E stable par les lois (interne et externe) induites par E.

Définition 4.1.2 (Espace affine)**.** Soit A un sous-espace vectoriel de l'espace vectoriel E et soit $v \in E$. L'espace affine que nous noterons $A + v$ est le *translaté* de A par v :

$$A + v = \{a + v \mid a \in A\}.$$

On trouve dans la littérature scientifique, la dénomination "coset" (terminologie anglaise) pour désigner un espace affine.

Définition 4.1.3. Soit une application linéaire, donnée par sa forme matricielle M, entre deux espaces vectoriels A et B :

$$M \ : \ A \to B.$$

L'*image* de l'application M est le sous-espace vectoriel $\mathrm{Im}(M)$ défini comme suit :

$$\mathrm{Im}(M) = \{y \mid M \cdot x = y, \text{ pour tout } x \in A\} \subseteq B.$$

De la même manière, le *noyau* de l'application M est le sous-espace vectoriel $\ker(M)$ défini comme suit :

$$\ker(M) = \{x \mid M \cdot x = 0_B\} \subseteq A.$$

Remarque 4.1.4. Si $\ker(M) = A$, l'application M est l'application nulle.

Le rang d'une matrice M est égal à la dimension de l'image de M. Nous notons :

$$\text{rank}(M) = \dim(\text{Im}(M)).$$

Théorème 4.1.5 (Théorème du rang)**.** *Soit une matrice M définissant une application linéaire $A \to B$ entre deux espaces vectoriels A et B. Alors*

$$\dim(A) = \textit{rank}(M) + \dim(\ker(M)).$$

Théorème 4.1.6. *Soient A et B deux sous-espaces vectoriels d'un même espace vectoriel. Alors*

$$\dim(A + B) = \dim(A) + \dim(B) - \dim(A \cap B).$$

Proposition 4.1.7. *Soit une matrice M définissant une application linéaire sur un espace vectoriel A, et soit A' un sous-espace vectoriel de A. Alors la restriction de M au sous-espace vectoriel A' possède un noyau tel que*

$$\ker(M_{|A'}) = \ker(M) \cap A'.$$

Théorème 4.1.8. *Soient $F : A \to B$ et $G : B \to C$ deux applications linéaires données par leurs matrices, entre des espaces vectoriels A, B et C. Alors*

$$\dim(\ker(G \cdot F)) = \dim(\ker(F)) + \dim(\ker(G) \cap \textit{Im}(F)).$$

Nous remarquons aussi qu'il existe une correspondance (unique) entre les fonctions de $\mathbb{F}_{2^n}$ dans $\mathbb{F}_{2^n}$ et l'espace vectoriel $\mathbb{F}_{2^n}^{2^n}$. En effet, nous pouvons représenter une fonction comme un vecteur des coefficients de sa représentation polynomiale.

Définition 4.1.9. Soit $\varphi : \mathbb{F}_{2^n}[x] \to \mathbb{F}_{2^n}^{2^n}$ la fonction bijective qui transforme une fonction sous forme polynomiale en vecteur de ses coefficients.

Plus précisément, soit $F(x) = \sum_{i=0}^{2^n-1} f_i x^i \in \mathbb{F}_{2^n}[x]$, alors

$$\varphi(F) = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_{2^n-1} \end{pmatrix} \in \mathbb{F}_{2^n}^{2^n}. \tag{4.1}$$

Pour faciliter la lecture, nous utiliserons la notation suivante $\varphi(F) := \overrightarrow{F}$.

Ainsi, pour tout $x \in \mathbb{F}_{2^n}$, nous avons que $F(x) = (0,\ x,\ x^2,\ x^3, \ldots,\ x^{2^n-1}) \cdot \overrightarrow{F}$.

4.2. Rappels sur les différentielles d'ordre supérieur

Dans cette section, nous allons rappeler succinctement le concept des *différentielles d'ordre supérieur* introduit par Xuejia Lai en 1994 [134].

Définition 4.2.1. Soit V un sous-espace vectoriel de $\mathbb{F}_{2^n}$ engendré par la famille $\{\alpha_i\} \subset \mathbb{F}_{2^n}$, $1 \leq i \leq m$, et soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Alors la différentielle d'ordre supérieur de F relativement à l'espace vectoriel V est définie comme suit :

$$\Delta_{\alpha_1,\ldots,\alpha_m} F(x) = \Delta_{\alpha_1}(\Delta_{\alpha_2,\ldots,\alpha_m} F(x)) = \sum_{v \in V} F(x + v).$$

Nous pouvons déduire de la définition ci dessus, que l'ordre dans lequel les différentielles sont réalisées n'importe pas. C'est à dire que les différentielles d'une fonction commutent pour la composition.

Corollaire 4.2.2. *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Soit une famille $\{\alpha_i\}$ de m éléments de $\mathbb{F}_{2^n}$, et soit $\pi \in S_m$ une permutation quelconque de m éléments. Alors*

$$\Delta_{\alpha_1,\ldots,\alpha_m} F = \Delta_{\alpha_{\pi(1)},\ldots,\alpha_{\pi(m)}} F.$$

Proposition 4.2.3. *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Soit une famille $\{\alpha_i\}$ de $m-1$ éléments de $\mathbb{F}_{2^n}$, et soit α_m un élément $\mathbb{F}_2$-linéairement dépendant de cette famille. Alors,*

$$\Delta_{\alpha_1,\ldots,\alpha_m} F = 0.$$

Le concept inhérent de cette notion est la chute du degré algébrique au fur et à mesure des différentielles successives. La dérivée *classique* des polynômes fait diminuer le degré *polynomial* d'une fonction, alors que la différentielle fait diminuer son degré *algébrique*. La proposition suivante résume ceci de manière plus concrète.

Proposition 4.2.4. *Soit une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et $\alpha \in \mathbb{F}_{2^n}^*$. Alors*

$$\deg(\Delta_\alpha F) \leq \deg(F) - 1.$$

4.3. Un point de vue matriciel

Dans cette section, nous adoptons un point de vue matriciel sur l'étude des différentielles. Nous donnons aussi les principaux résultats de ce chapitre.

4.3.1. Développement des différentielles

Nous allons maintenant voir le développement de l'expression de la différentielle d'une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ dans une direction $\alpha \in \mathbb{F}_{2^n}$.

$$\begin{aligned}
\Delta_\alpha F(x) = F(x) + F(x+\alpha) &= \sum_i f_i x^i + \sum_i f_i (x+\alpha)^i \\
&= \sum_i f_i x^i + \sum_i f_i \sum_{j,\, j \preceq i} x^j \alpha^{i-j} \\
&= \sum_i f_i x^i + \sum_i \sum_{j,\, j \preceq i} f_i x^j \alpha^{i-j} \\
&= \sum_i f_i x^i + \sum_j x^j \sum_{i,\, i \succeq j} f_i \alpha^{i-j} \\
&= \sum_j x^j \sum_{i,\, i \succ j} f_i \alpha^{i-j}.
\end{aligned}$$

D'où

$$\Delta_\alpha F(x) = \sum_j d_j(\alpha) x^j, \text{ où } d_j(\alpha) = \sum_{i \succ j} f_i \alpha^{i-j}.$$

En utilisant la définition 4.1.9, nous avons que

$$\varphi(\Delta_\alpha F) = \overrightarrow{\Delta_\alpha F} = \begin{pmatrix} d_0(\alpha) \\ d_1(\alpha) \\ \vdots \\ d_{2^n-1}(\alpha) \end{pmatrix}.$$

Les coefficients de la différentielle $\Delta_\alpha F(x)$, que nous avons notés $d_j(\alpha) = \sum_{i \succ j} \alpha^{i-j} f_i$, peuvent être écrit comme suit :

$$d_j(\alpha) = v_j \cdot \overrightarrow{F},$$

où $v_j = (a_0, \ldots, a_{2^n-1}) \in \mathbb{F}_{2^n}^{2^n}$ avec

$$a_i = \begin{cases} \alpha^{i-j} & \text{si } i \succ j \\ 0 & \text{sinon.} \end{cases}$$

A partir de maintenant, nous notons $M(\alpha)$, pour $\alpha \in \mathbb{F}_{2^n}$, la matrice de taille $2^n \times 2^n$ définie comme suit :

$$M(\alpha) = \begin{pmatrix} - & v_0 & - \\ & \vdots & \\ - & v_{2^n-1} & - \end{pmatrix},$$

alors $\overrightarrow{\Delta_\alpha F} = \begin{pmatrix} d_0(\alpha) \\ d_1(\alpha) \\ \vdots \\ d_{2^n-1}(\alpha) \end{pmatrix} = M(\alpha) \cdot \overrightarrow{F}.$

Nous pouvons désormais définir la fonction différentielle sur $\mathbb{F}_{2^n}^{2^n}$ en une direction $\alpha \in \mathbb{F}_{2^n}$:

$$\begin{array}{rccc} \overrightarrow{\Delta_\alpha}: & \mathbb{F}_{2^n}^{2^n} & \to & \mathbb{F}_{2^n}^{2^n} \\ & \overrightarrow{F} & \mapsto & M(\alpha) \cdot \overrightarrow{F} \end{array}$$

Cette fonction, $\overrightarrow{\Delta_\alpha}$, nous donne les coefficients de la différentielle d'une fonction F dans une direction $\alpha \in \mathbb{F}_{2^n}$. Voyons un exemple.

Exemple 4.3.1. Prenons $n = 4$ et $\alpha \in \mathbb{F}_{2^n}$. Alors la matrice $M(\alpha)$ est égale à

$$\begin{pmatrix}
. & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} & \alpha^{12} & \alpha^{13} & \alpha^{14} & \alpha^{15} \\
. & . & . & \alpha^2 & . & \alpha^4 & . & \alpha^6 & . & \alpha^8 & . & \alpha^{10} & . & \alpha^{12} & . & \alpha^{14} \\
. & . & . & \alpha & . & . & \alpha^4 & \alpha^5 & . & . & \alpha^8 & \alpha^9 & . & . & \alpha^{12} & \alpha^{13} \\
. & . & . & . & . & . & . & \alpha^4 & . & . & . & \alpha^8 & . & . & . & \alpha^{12} \\
. & . & . & . & . & \alpha & \alpha^2 & \alpha^3 & . & . & . & . & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} \\
. & . & . & . & . & . & . & \alpha^2 & . & . & . & . & . & \alpha^8 & . & \alpha^{10} \\
. & . & . & . & . & . & . & \alpha & . & . & . & . & . & . & \alpha^8 & \alpha^9 \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & \alpha^8 \\
. & . & . & . & . & . & . & . & . & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 & \alpha^7 \\
. & . & . & . & . & . & . & . & . & . & . & \alpha^2 & . & \alpha^4 & . & \alpha^6 \\
. & . & . & . & . & . & . & . & . & . & . & \alpha & . & . & \alpha^4 & \alpha^5 \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & \alpha^4 \\
. & . & . & . & . & . & . & . & . & . & . & . & . & \alpha & \alpha^2 & \alpha^3 \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & \alpha^2 \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & \alpha \\
. & . & . & . & . & . & . & . & . & . & . & . & . & . & . & .
\end{pmatrix}$$

où . remplace $0 \in \mathbb{F}_{2^n}$ pour plus de lisibilité.

Définition par récurrence de la matrice $M(\alpha)$

Soit $\alpha \in \mathbb{F}_{2^n}$, notons $M_i(\alpha)$ la suite des matrices ainsi définies :

$$M_1(\alpha) = \begin{pmatrix} 0 & \alpha \\ 0 & 0 \end{pmatrix} \quad \text{et} \quad M_i(\alpha) = \begin{pmatrix} M_{i-1}(\alpha) & \left(\alpha^{2^{i-1}}(\mathsf{Id}_{i-1} + M_{i-1}(\alpha))\right) \\ 0 & M_{i-1}(\alpha) \end{pmatrix}$$

où $i > 1$ est un entier et Id_{i-1} est la matrice identité de taille 2^{i-1}. La matrice $M(\alpha)$ est alors la n-ième matrice de cette suite :

$$M(\alpha) = M_n(\alpha).$$

4.3.2. Propriétés des fonctions différentielles

Une des premières propriétés que nous pouvons exprimer avec ce point de vue concerne les différentielles d'ordre supérieur. Il s'agit en fait de l'interprétation de la définition 4.2.1, du corollaire 4.2.2 et de la proposition 4.2.3.

Proposition 4.3.2. *Soient des éléments $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_{2^n}$. Alors*

1. $\overrightarrow{\Delta_{\alpha_1,\ldots,\alpha_m} F} = M(\alpha_1) \ldots M(\alpha_m) \cdot \overrightarrow{F}$;
2. $M(\alpha_i)M(\alpha_j) = M(\alpha_j)M(\alpha_i)$ *pour* $1 \leq i, j \leq m$;
3. *Si un élément, disons α_m, est $\mathbb{F}_2$-linéairement dépendant des autres $\alpha_1, \ldots, \alpha_{m-1}$, alors $\Pi_{1 \leq i \leq m} M(\alpha_i) = 0$. En particulier, $M(\alpha_i)$ est nilpotente d'ordre 2, c'est à dire que $M(\alpha_i)^2 = (0)$.*

Théorème 4.3.3. *Soit un élément $\alpha \in \mathbb{F}_{2^n}$. Le rang de la matrice $M(\alpha)$ est*

$$rank(M(\alpha)) = 2^{n-1}.$$

De plus, le noyau de l'application linéaire défini par $M(\alpha)$ est de dimension $\dim(\ker(M(\alpha))) = 2^{n-1}$ *et est engendré par les colonnes de la matrice $K(\alpha)$ définie par*

$$K(\alpha) = \left(\frac{\mathsf{Id}_{n-1}}{\alpha^{2^{n-1}-1} M_{n-1}(\alpha)} \right).$$

Démonstration. Il est aisé de vérifier que $M(\alpha)K(\alpha) = 0$. Cela signifie que l'espace vectoriel engendré par la matrice $K(\alpha)$ est inclus dans l'espace vectoriel $\ker(M(\alpha))$. De plus cela signifie aussi que $\dim(\ker(M(\alpha))) \geq 2^{n-1}$. Cependant, puisque $M(\alpha) = (m_{i,j})$ avec $m_{i,j} = \alpha^{j-i}$ si $j \succ i$ et 0 sinon, $M(\alpha)$ est une matrice triangulaire supérieure stricte et $m_{i,i+1} = \alpha$ pour tout entier pair i. C'est à dire que $M(\alpha)$ possède exactement 2^{n-1} éléments non nuls sur la sur-diagonale. Donc, $\text{rank}(M(\alpha)) \geq 2^{n-1}$. Nous concluons en utilisant le théorème du rang 4.1.5. □

De manière à faciliter la lecture, nous notons désormais par $Ker(\alpha)$ le sous-espace vectoriel $\ker(M(\alpha))$. Sa matrice génératrice sera $K(\alpha)$.

Le corollaire suivant est une conséquence directe du théorème précédent, mais est toutefois un résultat important.

Corollaire 4.3.4.

$$Im(M(\alpha)) = Ker(\alpha).$$

Démonstration. Soit $\overrightarrow{d} \in \text{Im}(M(\alpha))$. Grâce à la proposition 4.3.2, nous savons que $M(\alpha) \cdot \overrightarrow{d} = 0$. Cela signifie que $\text{Im}(M(\alpha)) \subset Ker(\alpha)$. D'après les théorèmes 4.3.3 et 4.1.5, nous savons que $\dim(\text{Im}(M(\alpha))) = \dim(Ker(\alpha)) = 2^{n-1}$ ce qui complète la preuve. □

Exemple 4.3.5. Soit $n = 4$ et $\alpha \in \mathbb{F}_{2^n}$. Nous avons

$$K(\alpha) = \begin{pmatrix}
1 & . & . & . & . & . & . & . \\
. & 1 & . & . & . & . & . & . \\
. & . & 1 & . & . & . & . & . \\
. & . & . & 1 & . & . & . & . \\
. & . & . & . & 1 & . & . & . \\
. & . & . & . & . & 1 & . & . \\
. & . & . & . & . & . & 1 & . \\
. & . & . & . & . & . & . & 1 \\
. & \alpha^8 & \alpha^9 & \alpha^{10} & \alpha^{11} & \alpha^{12} & \alpha^{13} & \alpha^{14} \\
. & . & . & \alpha^9 & . & \alpha^{11} & . & \alpha^{13} \\
. & . & . & \alpha^8 & . & . & \alpha^{11} & \alpha^{12} \\
. & . & . & . & . & . & . & \alpha^{11} \\
. & . & . & . & . & \alpha^8 & \alpha^9 & \alpha^{10} \\
. & . & . & . & . & . & . & \alpha^9 \\
. & . & . & . & . & . & . & \alpha^8 \\
. & . & . & . & . & . & . & .
\end{pmatrix}.$$

Le résultat important que nous apprend le corollaire 4.3.4 est qu'une fonction f sur $\mathbb{F}_{2^n}$ est la différentielle d'une autre fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ en une direction $\alpha \in \mathbb{F}_{2^n}$ *si et seulement si* $f(x) + f(x+\alpha) = 0$ pour tout $x \in \mathbb{F}_{2^n}$. Ce que nous résumons dans le corollaire suivant.

Corollaire 4.3.6. *Soit une fonction $f : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Alors il existe $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et $\alpha \in \mathbb{F}_{2^n}$ tel que $\Delta_\alpha F = f$* si et seulement si *α est une 0-structure linéaire de f.*

De plus, les coefficients des fonctions ayant α pour 0-structure linéaire peuvent être obtenus par combinaison linéaire des colonnes de la matrice $K(\alpha)$.

4.3.3. Propriétés des fonctions différentielles d'ordre supérieur

Nous allons voir dans cette sous-section des résultats préliminaires concernant les différentielles d'ordre supérieur, *i.e.* la composition de fonctions différentielles.

Lemme 4.3.7. *Soient deux éléments distincts α et $\beta \in \mathbb{F}_{2^n}$. Alors,*

$$Im(M(\alpha)M(\beta)) = Ker(\alpha) \cap Ker(\beta), \tag{4.2}$$

et

$$\ker(M(\alpha)M(\beta)) = Ker(\alpha) + Ker(\beta). \tag{4.3}$$

Démonstration. Nous avons que

$$\begin{aligned}
\mathrm{Im}(M(\alpha)M(\beta)) &= \{(M(\alpha)M(\beta)) \cdot x \text{ pour } x \in \mathbb{F}_{2^n}^{2^n}\} \\
&= \{M(\alpha) \cdot x \text{ pour } x \in \mathrm{Im}(M(\beta))\} \\
&= \mathrm{Im}(M(\alpha)_{|\mathrm{Im}(M(\beta))}) \\
(&= \mathrm{Im}(M(\beta)_{|\mathrm{Im}(M(\alpha))}) \quad \text{car } M(\alpha) \text{ et } M(\beta) \text{ commutent.}).
\end{aligned}$$

Nous voyons aussi que $\mathrm{Im}(M(\beta))$ est invariant sous l'action de $M(\alpha)$, c'est à dire que pour tout $y \in \mathbb{F}_{2^n}^{2^n}$ tels que $y = M(\beta) \cdot x \in \mathrm{Im}(M(\beta))$ pour $x \in \mathbb{F}_{2^n}^{2^n}$, nous avons $M(\alpha) \cdot y = M(\alpha) \cdot (M(\beta) \cdot x) = M(\beta) \cdot (M(\alpha) \cdot x) \in \mathrm{Im}(M(\beta))$. Donc $\mathrm{Im}(M(\alpha)_{|\mathrm{Im}(M(\beta))}) = \ker(M(\alpha)_{|\mathrm{Im}(M(\beta))})$, et en utilisant la proposition 4.1.7 et le corollaire 4.3.4 il s'ensuit que

$$\ker(M(\alpha)_{|\mathrm{Im}(M(\beta))}) = Ker(\alpha) \cap \mathrm{Im}(M(\beta)) = Ker(\alpha) \cap Ker(\beta).$$

Pour prouver la seconde égalité (4.3), nous allons utiliser le théorème du rang 4.1.5 qui nous donne que $\dim(\ker(M(\alpha)M(\beta))) + \mathrm{rank}(M(\alpha)M(\beta)) = 2^n$. En connaissant l'égalité (4.2) nous pouvons écrire

$$\dim(\ker(M(\alpha)M(\beta))) + \dim(Ker(\alpha) \cap Ker(\beta)) = 2^n = \dim(Ker(\alpha)) + \dim(Ker(\beta))$$

c'est-à-dire

$$\begin{aligned}
\dim(\ker(M(\alpha)M(\beta))) &= \dim(Ker(\alpha)) + \dim(Ker(\beta)) - \dim(Ker(\alpha) \cap Ker(\beta)) \\
&= \dim(Ker(\alpha) + Ker(\beta))
\end{aligned}$$

en utilisant le théorème 4.1.6. Notons que nous avons l'inclusion naturelle :

$$Ker(\alpha) + Ker(\beta) \subseteq \ker(M(\alpha)M(\beta)),$$

qui nous permet de conclure. □

Nous pouvons voir que

$$Ker(\alpha) \cap Ker(\beta) \subset Ker(\alpha + \beta). \tag{4.4}$$

En effet, si $f(x) + f(x+\alpha) = f(x) + f(x+\beta) = 0$ pour tout $x \in \mathbb{F}_{2^n}$, on obtient que

$$f(y) + f(y + \alpha + \beta) = 0,$$

avec $y = x + \alpha$ ou bien $y = x + \beta$ et pour tout $x \in \mathbb{F}_{2^n}$.

Nous pouvons déduire du lemme 4.3.7 un résultat plus général, décrit dans le théorème suivant.

Théorème 4.3.8. *Soit $\alpha_1, \ldots, \alpha_m$ une famille d'éléments $\mathbb{F}_2$-linéairement indépendants de $\mathbb{F}_{2^n}$. Alors,*

$$Im(\overrightarrow{\Delta_{\alpha_1,\ldots,\alpha_m}}) = Im\left(\prod_{1\leq i\leq m} M(\alpha_i)\right) = \bigcap_{1\leq i\leq m} Ker(\alpha_i). \tag{4.5}$$

De plus,

$$\dim\left(\bigcap_{1\leq i\leq m} Ker(\alpha_i)\right) = 2^{n-m}. \tag{4.6}$$

Démonstration. Nous allons prouver la première égalité (4.5) par récurrence sur m. Notons que la méthode qui sera utilisée dans cette preuve est semblable, quoique plus générale, à celle utiliser lors de la preuve du lemme 4.3.7.

De plus, l'initialisation de la récurrence a déjà été effectuée au lemme 4.3.7 équation. (4.2).

Supposons maintenant que l'égalité (4.5) est vraie. Nous allons montrer que c'est encore vrai si nous ajoutons un autre élément $\mathbb{F}_2$-linéairement indépendant à notre famille de départ $\{\alpha_i\}$. Supposons que $\beta \in \mathbb{F}_{2^n}$ est cet élément. Nous avons que

$$\mathrm{Im}(M(\beta)\prod_i M(\alpha_i)) = \mathrm{Im}(M(\beta)_{|\mathrm{Im}(\prod_i M(\alpha_i))}).$$

Comme $M(\beta)$ commute avec les matrices $M(\alpha_i)$, l'espace vectoriel $\mathrm{Im}(\prod_i M(\alpha_i))$ est invariant sous l'action de $M(\beta)$. D'où

$$\begin{aligned}
\mathrm{Im}(M(\beta)_{|\mathrm{Im}(\prod_i M(\alpha_i))}) &= \ker(M(\beta)_{|\mathrm{Im}(\prod_i M(\alpha_i))}) \\
&= Ker(\beta) \cap \mathrm{Im}(\prod_i M(\alpha_i)) \\
&= Ker(\beta) \cap \left(\bigcap_i K(\alpha_i)\right),
\end{aligned}$$

où la dernière égalité est obtenue d'après l'hypothèse de récurrence.

D'après le théorème du rang 4.1.5, nous avons que

$$\dim(\ker(\prod_{1\leq i\leq m} M(\alpha_i))) + \dim(\mathrm{Im}(\prod_{i\leq i\leq m} M(\alpha_i))) = 2^n. \tag{4.7}$$

Notons que d'après le théorème 4.1.8, nous avons aussi que

$$\begin{aligned}
\dim(\ker(\prod_{i=1}^{m} M(\alpha_i))) &= \dim(\ker(\prod_{i=1}^{m-1} M(\alpha_i))) + \dim(Ker(\alpha_m) \cap \left(\mathrm{Im}(\prod_{i=1}^{m-1} M(\alpha_i))\right)) \\
&= \dim(\ker(\prod_{i=1}^{m-1} M(\alpha_i))) + \dim(\bigcap_{i=1}^{m} Ker(\alpha_i)) \qquad \text{d'après (4.5)} \\
&= \dim(\ker(\prod_{i=1}^{m-2} M(\alpha_i))) + \dim(\bigcap_{i=1}^{m-1} Ker(\alpha_i)) + \dim(\bigcap_{i=1}^{m} Ker(\alpha_i)) \\
&\vdots \\
&= \sum_{k=1}^{m} \dim(\bigcap_{i=1}^{k} Ker(\alpha_i)).
\end{aligned}$$

Donc l'équation (4.7) devient

$$\sum_{k=1}^{m} \dim(\bigcap_{i=1}^{k} Ker(\alpha_i)) + \dim(\bigcap_{i=1}^{m} Ker(\alpha_i)) = 2^n. \tag{4.8}$$

Finalement, nous savons d'après le théorème 4.3.3 et le corollaire 4.3.4 que $\dim(Ker(\alpha)) = 2^n - 1$. Il s'ensuit par récurrence sur m dans l'équation (4.8) que

$$\dim\left(\bigcap_{i=1}^{m} Ker(\alpha_i)\right) = 2^{n-m}.$$

□

Remarque 4.3.9. Cela veut dire aussi qu'une fonction est une différentielle d'ordre m si et seulement si elle possède m 0-structures différentes linéaires $\mathbb{F}_2$-linéairement indépendantes.

Les espaces vectoriels $Ker(\alpha_i)$ ont une structure particulière. En effet, l'intersection de tels espaces vectoriels est distributive sur la somme, comme le montre la proposition suivante.

Proposition 4.3.10. *Soit $\{\alpha_i\}$ une famille d'éléments de $\mathbb{F}_{2^n}$ et soit $\beta \in \mathbb{F}_{2^n}$. Alors,*

$$\left(\sum_i Ker(\alpha_i)\right) \cap Ker(\beta) = \sum_i \left(Ker(\alpha_i) \cap Ker(\beta)\right).$$

Démonstration. Puisque $\mathrm{Im}(M(\alpha_i)) = Ker(\alpha_i)$ et que

$$\mathrm{Im}(M(\alpha_i)M(\alpha_j)) = Ker(\alpha_i) \bigcap Ker(\alpha_j),$$

tout vecteur de $\sum_i (Ker(\alpha_i) \cap Ker(\beta))$ est de la forme $\sum_i M(\alpha_i)M(\beta) \cdot \overrightarrow{F_i}$ avec $\overrightarrow{F_i} \in \mathbb{F}_{2^n}^{2^n}$. Alors

$$\sum_i M(\alpha_i)M(\beta) \cdot \overrightarrow{F_i} = M(\beta) \underbrace{\left(\sum_i M(\alpha_i) \cdot \overrightarrow{F_i}\right)}_{\in \sum_i Ker(\alpha_i)} \in \left(\sum_i Ker(\alpha_i)\right) \cap Ker(\beta).$$

Réciproquement, un vecteur de $(\sum_i Ker(\alpha_i)) \cap Ker(\beta)$ est de la forme

$$\overrightarrow{F} = \sum_i M(\alpha_i) \cdot \overrightarrow{F_i} \text{ tel que } M(\beta) \cdot \overrightarrow{F} = 0_{\mathbb{F}_{2^n}^{2^n}}.$$

Donc $\overrightarrow{F} = M(\beta) \cdot \overrightarrow{G}$ pour $\overrightarrow{G}$ un vecteur de la forme $\overrightarrow{G} = \sum_i M(\alpha_i) \cdot \overrightarrow{G_i}$, avec $\overrightarrow{G_i} \in \mathbb{F}_{2^n}^{2^n}$. En conclusion,

$$\overrightarrow{F} = \sum_i M(\alpha_i)M(\beta) \cdot \overrightarrow{G_i} \in \sum_i (Ker(\alpha_i) \cap Ker(\beta)).$$

□

Cette propriété de distributivité reste vraie sur les sous-espaces vectoriels de $Ker(\alpha_i)$.

D'après la proposition 4.3.10, et puisque l'addition d'espaces vectoriels est une opération *additive* (*i.e.* si A et B sont des sous-espaces de l'espace vectoriel E tels que $A \cap B = 0_E$, alors $\dim(A + B) = \dim(A) + \dim(B)$), nous pouvons adapter le principe d'*inclusion-exclusion* à l'ensemble des espaces vectoriels $\{Ker(\alpha) \mid \alpha \in \mathbb{F}_{2^n}\}$ comme cela est fait dans le lemme suivant. La preuve de ce résultat peut être trouvée à l'annexe A.2.1.

Lemme 4.3.11. *Soit $\{\alpha_i\} \subset \mathbb{F}_{2^n}$ une famille de m éléments $\mathbb{F}_2$-linéairement indépendants. Alors,*

$$\dim(\sum_{i=1}^{m} Ker(\alpha_i)) = \sum_{k=1}^{m} (-1)^{k+1} \left(\sum_{1 \le i_1 \le \cdots \le i_k \le m} \dim(Ker(\alpha_{i_1}) \cap \cdots \cap Ker(\alpha_{i_k})) \right). \tag{4.9}$$

Démonstration. voir annexe A.2.1 □

Nous pouvons obtenir une formule plus simple de $\dim(\sum_{i=1}^{m} Ker(\alpha_i))$, en combinant l'équation (4.6) avec l'équation précédente (4.9).

Corollaire 4.3.12.

$$\dim(\sum_{i=1}^{m} Ker(\alpha_i)) = \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} 2^{n-k}.$$

Théorème 4.3.13. *Soit $\alpha_1, \ldots, \alpha_m$ une famille d'éléments $\mathbb{F}_2$-linéairement indépendants de $\mathbb{F}_{2^n}$. Alors,*

$$\ker(\overrightarrow{\Delta_{\alpha_1, \ldots, \alpha_m}}) = \ker(\prod_{1 \le i \le m} M(\alpha_i)) = \sum_{1 \le i \le m} Ker(\alpha_i). \tag{4.10}$$

De plus,

$$\dim\left(\sum_{i=1}^{m} Ker(\alpha_i)\right) = 2^n - 2^{n-m}. \tag{4.11}$$

Démonstration. Soit $\overrightarrow{v} = \sum_{i=1}^{m} \overrightarrow{v_i}$ avec $\overrightarrow{v_i} \in Ker(\alpha_i)$ pour $1 \le i \le m$, c'est-à-dire $\overrightarrow{v} \in \sum_{i=1}^{m} Ker(\alpha_i)$. Alors,

$$\left(\prod_{i=1}^{m} M(\alpha_i)\right) \cdot \overrightarrow{v} = 0$$

ce qui implique

$$\overrightarrow{v} \in \ker(\prod_{i=1}^{m} M(\alpha_i)).$$

Montrer que $\dim(\ker(\prod_{i=1}^{m} M(\alpha_i))) = \dim(\sum_{i=1}^{m} Ker(\alpha_i))$ est maintenant suffisant pour conclure cette preuve. Nous savons d'après les théorèmes 4.1.5 et 4.3.8 que

$$\dim(\ker(\prod_{i=1}^{m} M(\alpha_i))) = 2^n - \dim(\mathrm{Im}(\prod_{i=1}^{m} M(\alpha_i))) = 2^n - 2^{n-m},$$

et que d'après le corollaire 4.3.12,

$$\dim(\sum_{i=1}^{m} Ker(\alpha_i)) = \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} 2^{n-k}.$$

Nous allons donc montrer par récurrence sur m que

$$\sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} 2^{n-k} = 2^n - 2^{n-m}. \tag{4.12}$$

L'initialisation de la récurrence se fait pour $m = 1$ et nous voyons qu'elle est vérifiée par l'égalité $2^{n-1} = 2^n - 2^{n-1}$. Notre hypothèse de récurrence est donc que l'équation (4.12) est vraie jusqu'à un rang $m \leq n$. Alors,

$$\begin{aligned} \sum_{k=1}^{m+1} (-1)^{k+1} \binom{m+1}{k} 2^{n-k} &= \sum_{k=0}^{m} (-1)^{k} \binom{m+1}{k+1} 2^{n-(k+1)} \\ &= \sum_{k=0}^{m} (-1)^{k} \binom{m}{k} 2^{n-(k+1)} + \sum_{k=0}^{m} (-1)^{k} \binom{m}{k+1} 2^{n-(k+1)}. \end{aligned} \tag{4.13}$$

Nous voyons que nous pouvons utiliser l'hypothèse de récurrence pour réduire le premier terme de la dernière somme (4.13) ci-dessus. En effet,

$$\begin{aligned} \sum_{k=0}^{m} (-1)^{k} \binom{m}{k} 2^{n-(k+1)} &= 2^{n-1} + \sum_{k=1}^{m} (-1)^{k} \binom{m}{k} 2^{n-(k+1)} \\ &= 2^{n-1} - \frac{1}{2} \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} 2^{n-k} \\ &= 2^{n-1} - \frac{1}{2}(2^n - 2^{n-m}). \end{aligned}$$

Le dernier terme de la somme (4.13) peut aussi être légèrement modifié puisque, $(-1)^k = (-1)^{k+2}$ et par convention $\binom{m}{m+1} = 0$. En changeant alors les indices, nous pouvons identifier ce terme avec l'hypothèse de récurrence.

$$\sum_{k=0}^{m} (-1)^{k} \binom{m}{k+1} 2^{n-(k+1)} = \sum_{k=0}^{m-1} (-1)^{k+2} \binom{m}{k+1} 2^{n-(k+1)}$$

$$= \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} 2^{n-k}$$
$$= 2^n - 2^{n-m}.$$

Finalement, nous obtenons

$$\sum_{k=1}^{m+1} (-1)^{k+1} \binom{m+1}{k} 2^{n-k} = 2^{n-m-1} + 2^n - 2^{n-m} = 2^n - 2^{n-(m+1)}.$$

□

Le théorème 4.3.13 nous dit que si une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ est telle que sa différentielle d'ordre m est nulle, *i.e.* $\Delta_{\alpha_1,\ldots,\alpha_m} F(x) = 0$ pour des éléments $\alpha_1, \ldots, \alpha_m \in \mathbb{F}_{2^n}$ $\mathbb{F}_2$-linéairement indépendants, alors elle peut se décomposer comme une somme de différentielles :

$$F(x) = F_1(x) + \cdots + F_m(x),$$

où $\Delta_{\alpha_i} F_i(x) = 0$ pour $1 \leq i \leq m$.

4.4. Intégration

Dans cette section nous allons présenter sous quelles conditions, nécessaires et suffisantes, nous pouvons *intégrer* dans $\mathbb{F}_{2^n}$

Nous pouvons tout d'abord remarquer que deux fonctions sur $\mathbb{F}_{2^n}$ distinctes, disons F et G, partagent la même différentielle en une direction $\alpha \in \mathbb{F}_{2^n}$ si et seulement si $\overrightarrow{G}$ est dans l'espace affine (comme défini à la définition 4.1.2) $Ker(\alpha) + \overrightarrow{F}$. De manière équivalente,

$$\Delta_\alpha F = \Delta_\alpha G \Leftrightarrow \overrightarrow{G} \in Ker(\alpha) + \overrightarrow{F} = \{(\overrightarrow{F} + v) \text{ tel que } \overrightarrow{v} \in Ker(\alpha)\}.$$

En effet,

$$\Delta_\alpha F = \Delta_\alpha G \Leftrightarrow \Delta_\alpha (F + G) = 0 \Leftrightarrow \overrightarrow{F} + \overrightarrow{G} \in Ker(\alpha).$$

Ce résultat peut être généralisé à plusieurs différentielles qui seraient partagées, comme le montre le théorème suivant.

Théorème 4.4.1. *Soient deux fonctions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et soit $\{\alpha_i\}$ une famille de $m \leq n$ éléments $\mathbb{F}_2$-linéairement indépendants de $\mathbb{F}_{2^n}$. Alors*

$$\Delta_{\alpha_i} F = \Delta_{\alpha_i} G \quad \text{for } i = 1, \ldots, m \quad \Leftrightarrow \overrightarrow{G} \in \bigcap_i Ker(\alpha_i) + \overrightarrow{F}. \tag{4.14}$$

Démonstration. Supposons tout d'abord que $\overrightarrow{G} \in \bigcap_i Ker(\alpha_i) + \overrightarrow{F}$. Alors,

$$\overrightarrow{G} = \overrightarrow{F} + \overrightarrow{v}$$

avec $\overrightarrow{v} \in \bigcap_{1 \leq i \leq m} Ker(\alpha_i)$. D'où

$$M(\alpha_i) \cdot \overrightarrow{G} = M(\alpha_i) \cdot \overrightarrow{F} + M(\alpha_i) \cdot \overrightarrow{v} = M(\alpha_i) \cdot \overrightarrow{F} \Rightarrow \Delta_{\alpha_i} F = \Delta_{\alpha_i} G.$$

Réciproquement, pour tout $1 \leq i \leq m$ nous avons que $\Delta_{\alpha_i}(F+G) = 0$. Donc,

$$\exists \overrightarrow{v_i} \in Ker(\alpha_i) \text{ tel que } \overrightarrow{F} + \overrightarrow{G} = \overrightarrow{v_i} \text{ pour tout } 1 \leq i \leq m \Rightarrow v_1 = v_2 = \cdots = v_m.$$

Cela implique que $\overrightarrow{F} + \overrightarrow{G} \in \cap_i Ker(\alpha_i)$, qui conclut la preuve. □

Corollaire 4.4.2. *Deux fonctions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, partagent les mêmes différentielles dans les directions données par un sous-espace vectoriel de $\mathbb{F}_{2^n}$, disons V, si et seulement si tout élément de V est une $0-$structure linéaire de $F+G$.*

Ce résultat nous permet donc de définir une équivalence de fonctions de $\mathbb{F}_{2^n}$:

Définition 4.4.3 (Equivalence différentielle $\sim_V$)**.** Soient deux fonctions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ et $G : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$. Nous définissons l'*équivalence différentielle* entre F et G relativement à un sous-espace vectoriel $V \subseteq \mathbb{F}_{2^n}$, et que nous notons $F \sim_V G$, comme suit :

$$F \sim_V G \Leftrightarrow \Delta_v F = \Delta_v G, \quad \forall v \in V.$$

D'après le théorème 4.4.1 et l'équation (4.4), la *classe d'équivalence* (ou *coset*) de F respectivement au sous-espace vectoriel V de $\mathbb{F}_{2^n}$ est l'espace affine $\bigcap_{w \in V} Ker(w) + \overrightarrow{F}$, c'est-à-dire

$$F \sim_V G \Leftrightarrow \overrightarrow{G} \in \bigcap_{w \in V} Ker(w) + \overrightarrow{F}.$$

Nous pouvons vérifier aisément qu'il s'agit bien d'une relation d'équivalence :

- **Réflexivité :** $F \sim_V F$;
- **Symétrie :** si $F \sim_V G$ alors $G \sim_V F$;
- **Transitivité :** si $F \sim_V G$ et $G \sim_V H$ alors $F \sim_V H$.

Les fonctions $h : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, telles que $\overrightarrow{h} \in \cap_{w \in V} Ker(w)$, sont des fonctions dont le degré algébrique est inférieur à $n - \dim(V)$. En effet, d'après la proposition 4.2.4 et le fait que $\cap_{w \in V} Ker(w) = \mathrm{Im}(\Pi_{w \in V} M(w))$ (théorème 4.3.8), on a qu'il existe $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que $(\Pi_{w \in V} M(w))\, \overrightarrow{F} = \overrightarrow{h}$ et

$$n \geq \deg(F) > \deg(\Delta_w F) > \cdots > \deg(h) \Rightarrow n - \dim(V) \geq \deg(h).$$

Exemple 4.4.4. Soit $n = 4$, et soit z un générateur du groupe cyclique $\mathbb{F}_{2^n}^*$ tel que $z^4 = z + 1$. Prenons la fonction $F : x \mapsto x^7$ et pour espace vectoriel $V = \{0, z, z^2, z + z^2\}$. La fonction suivante, que nous noterons h,

$$x \mapsto z^6 x^{12} + z^3 x^{10} + z^{14} x^9 + z x^8 + z^2 x^6 + z^8 x^5 + x^4 + x^3 + z^6 x^2 + z^5 x + z^{14},$$

possède deux 0-structures linéaires, z et z^2. Ainsi

$$\Delta_z(F + h) = \Delta_z(F),$$
$$\Delta_{z^2}(F + h) = \Delta_{z^2}(F),$$

mais comme z et z^2 sont les deux seules 0-structures linéaires de h (avec $z + z^2$),

$$\Delta_w(F + h) \neq \Delta_w(F), \quad \forall w \notin V \qquad \text{et ainsi} \qquad F + h \sim_V F.$$

4.4.1. Cas général

Dans cette sous-section, nous allons voir le cas général de l'intégration dans $\mathbb{F}_{2^n}$. Nous verrons aussi un algorithme permettant de calculer effectivement cette fonction intégrale.

Le théorème suivant est un théorème de consistance, c'est à dire qu'il donne les conditions nécessaires et suffisantes pour pouvoir intégrer dans $\mathbb{F}_{2^n}$.

Théorème 4.4.5. *Soit $\{\alpha_i\}$ une famille de $m \leq n$ éléments $\mathbb{F}_2$-linéairement indépendants de $\mathbb{F}_{2^n}$ et soient des fonctions $f_i : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, $0 \leq i < m$, telles que $\overrightarrow{f_i} \in Ker(\alpha_i)$ pour $i = 0, \ldots, m-1$. Alors, il existe au moins une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que*

$$\Delta_{\alpha_i} F = f_i \quad \text{pour tout } 0 \leq i < m \tag{4.15}$$

si et seulement si

$$\Delta_{\alpha_j} f_i = \Delta_{\alpha_i} f_j \quad \text{pour tout } 0 \leq i, j < m. \tag{4.16}$$

Démonstration. Supposons que F est telle que $\Delta_{\alpha_i} F = f_i$ pour tout $0 \leq i < m$. Alors

$$\Delta_{\alpha_j} f_i = \Delta_{\alpha_j, \alpha_i} F = \Delta_{\alpha_i, \alpha_j} F = \Delta_{\alpha_i} f_j.$$

Réciproquement, pour tout $i \neq j$, nous avons $\Delta_{\alpha_j} f_i = \Delta_{\alpha_i} f_j$. Puisque $\overrightarrow{f_i} \in Ker(\alpha_i)$, il existe une fonction, disons $F_i : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, telle que $\Delta_{\alpha_i} F_i = f_i$. Donc l'équation (4.16) devient

$$\begin{aligned} & \Delta_{\alpha_i, \alpha_j} F_i = \Delta_{\alpha_i, \alpha_j} F_j \\ \Leftrightarrow \qquad & \Delta_{\alpha_i, \alpha_j} (F_i + F_j) = 0. \end{aligned}$$

D'après le lemme 4.3.7, la dernière égalité nous dit que $\overrightarrow{F_i + F_j} \in Ker(\alpha_i) + Ker(\alpha_j)$. Il existe donc $\overrightarrow{G_i} \in Ker(\alpha_i)$ et $\overrightarrow{G_j} \in Ker(\alpha_j)$ (probablement pas uniques) telles que

$$F_i + F_j = G_i + G_j.$$

Il est important de noter maintenant, que pour un entier i particulier, nous pouvons choisir que G_i sera la même fonction chaque fois que la fonction F_i sera impliquée. En effet, soient i, j et l trois entiers distincts et soient $\overrightarrow{G_i}, \overrightarrow{G_i'} \in Ker(\alpha_i)$, $\overrightarrow{G_j}, \overrightarrow{G_j'} \in Ker(\alpha_j)$ et $\overrightarrow{G_l}, \overrightarrow{G_l'} \in Ker(\alpha_l)$ tels que

$$F_i + F_j = G_i + G_j \tag{4.17}$$

$$F_j + F_l = G_j' + G_l \tag{4.18}$$

$$F_i + F_l = G_i' + G_l'. \tag{4.19}$$

En additionnant les équations (4.17) et (4.18) termes par termes, nous obtenons

$$F_i + F_l = G_i + G_j + G_j' + G_l = G_i' + G_l',$$

en identifiant avec l'équation (4.19). Cela nous montre donc que nous pouvons choisir sans restriction $G_i = G_i'$, $G_j = G_j'$ et $G_l = G_l'$. Cette remarque est toujours vraie lorsque nous considérons plus d'équations du type (4.17), (4.18) et (4.19).

En conclusion, pour chaque entier i, toutes les fonctions $F_i + G_i$ sont identiques et vérifient l'équation (4.15) :

$$\Delta_{\alpha_i}(F_i + G_i) = \Delta_{\alpha_i} F_i = f_i.$$

□

Remarque 4.4.6. Il existe un lien entre cette méthode d'intégration et la recherche de fonction potentiel dans le milieu continu.

L'algorithme suivant présente une méthode de reconstruction d'une fonction intégrale $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ à partir de fonctions différentielles vérifiant les conditions du théorème 4.4.5. Tout d'abord, nous rappelons que si un système linéaire $Ax = b$ (où A est une matrice et x et b sont des vecteurs) est *consistant* (*i.e.* possède une solution), il est assez facile de calculer une solution particulière en utilisant l'algèbre linéaire de base. nous notons cette solution par $A\backslash b$.

Algorithme 4 Algorithme d'intégration dans $\mathbb{F}_{2^n}$

Entrée :

$m \leq n$ éléments $\mathbb{F}_2$-linéairement indépendants $\{\alpha_i\} \subset \mathbb{F}_{2^n}$
et m fonctions $f_i : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, tels que $\overrightarrow{f_i} \in Ker(\alpha_i)$ et $\Delta_{\alpha_j} f_i = \Delta_{\alpha_i} f_j$ pour tout $0 \leq i, j < m$.

Sortie :

une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que $\Delta_{\alpha_i} F = f_i$ pour tout $0 \leq i < m$.

Fonction RECONSTRUCTION($\{\alpha_i, f_i\}$)

$M \leftarrow K(\alpha_0)$
$\overrightarrow{sol} \leftarrow O_{\mathbb{F}_{2^n}^{2^n}}$
$\overrightarrow{F_0} \leftarrow M(\alpha_0)\backslash\overrightarrow{f_0}$
Pour tout $i \in [1, m-1]$
    $\overrightarrow{F_i} \leftarrow M(\alpha_i)\backslash\overrightarrow{f_i}$
    $M' \leftarrow M(\alpha_i)$
    $\overrightarrow{sol} \leftarrow \overrightarrow{sol} + M \cdot \left((M'M)\backslash(M' \cdot (\overrightarrow{F_0 + F_i + sol}))\right)$
    $M \leftarrow M \cdot \kappa$ ▷ κ est la matrice génératrice de $\ker(M'M)$
$sol \leftarrow sol + F_0$
Retourner *sol*

Nous remarquons que la matrice M à la fin de l'exécution de l'algorithme correspond en fait à une matrice génératrice de l'espace vectoriel $\cap_i K(\alpha_i)$.

4.4.2. Construction de fonctions quadratiques APN

Dans cette sous-section, nous utilisons les résultats d'intégration de la sous-section précédente pour construire de manière effective des fonctions quadratiques APN. L'idée principale est d'utiliser des fonctions affines 2-to-1 qui seraient consistantes au sens du théorème 4.4.5 pour intégrer des fonctions quadratiques APN.

Dans le cas spécial des fonctions linéaires, nous noterons que le corollaire 4.3.6 nous apprend un fait intéressant. Toute fonction linéaire qui n'est pas une permutation de $\mathbb{F}_{2^n}$ est une fonction différentielle dans les directions de toutes ses racines.

Le corollaire suivant statue ce résultat de manière plus précise, et dans le cas légèrement plus général des fonctions affines de $\mathbb{F}_{2^n}$.

Corollaire 4.4.7. *Soit $L : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ une fonction linéaire non bijective, et soit $\ell \in \mathbb{F}_{2^n}$. Soit $V \subset \mathbb{F}_{2^n}$ l'ensemble des racines de L. Alors il existe des fonctions*

$$F_v : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \text{ telles que } \Delta_v F_v = L + \ell, \qquad \forall v \in V.$$

Ou de manière équivalente,

$$(L + \ell) \in \bigcap_{w \in V} Ker(w).$$

Démonstration. Nous avons que

$$\Delta_\alpha(L + \ell)(x) = L(x) + L(x + \alpha) = L(\alpha),$$

pour $\alpha \in \mathbb{F}_{2^n}^*$. Donc pour toutes racines $\alpha \in V$ de L et d'après le corollaire 4.3.4, nous avons :

$$\mathrm{Im}(M(\alpha)) = Ker(\alpha).$$

Nous pouvons en conclure qu'il existe des fonctions $F_\alpha : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telles que

$$\Delta_\alpha F_\alpha = L + \ell, \qquad \forall \alpha \in V.$$

□

Le théorème 4.4.5, qui nous donne une condition nécessaire et suffisante pour que des fonctions différentielles partagent une fonction intégrale commune, s'adapte dans le cas des fonctions affines.

Corollaire 4.4.8. *Soit $\{\alpha_i\}$ une famille de $m \leq n$ éléments $\mathbb{F}_2$-linéairement indépendants de $\mathbb{F}_{2^n}$. Soient des fonctions affines $L_i : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ non bijectives. Alors il existe au moins une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que*

$$\Delta_{\alpha_i} F = L_i \quad \textit{pour tout } 1 \leq i \leq m$$

si et seulement si

$$L_i(\alpha_j) + L_i(0) = L_j(\alpha_i) + L_j(0) \quad \textit{pour tout } 1 \leq i \leq m.$$

La méthode suivante permet de construire des fonctions quadratiques APN. Prenons une famille $\{\alpha_1, \ldots, \alpha_n\}$ d'éléments $\mathbb{F}_2$-linéairement indépendants de $\mathbb{F}_{2^n}$. Choisissons un vecteur $b_1 = (0, \beta_2, \ldots, \beta_n) \in \mathbb{F}_{2^n}^n$ tel que les éléments $\beta_2, \ldots, \beta_n$ soient $\mathbb{F}_2$-linéairement indépendants. Il existe alors une fonction affine L_1 sur $\mathbb{F}_{2^n}$ vérifiant

$$b_1 = (L_1(0) + L_1(\alpha_1), L_1(0) + L_1(\alpha_2), \ldots, L_1(0) + L_1(\alpha_n)). \tag{4.20}$$

Ainsi, la fonction L_1 est une fonction $\mathbb{F}_2$-linéaire, 2-to-1. Retrouver les coefficients du polynôme L_1 à partir de l'équation (4.20) demande simplement une petite manipulation d'algèbre linéaire.

Nous construisons les autres vecteurs $b_2, \ldots, b_n \in \mathbb{F}_{2^n}^n$ en suivant la récurrence :

$$b_i = (\beta_{(i-1)n+1}, \beta_{(i-1)n+2}, \ldots, \beta_{in}), \qquad 1 \leq i \leq n,$$

avec les restrictions suivantes :

$$\beta_{(i-1)n+j} = \begin{cases} \beta_{(j-1)n+i} & \text{si } j < i \\ 0 & \text{si } j = i \end{cases},$$

et en choisissant les $\beta_{(i-1)n+j} \in \mathbb{F}_{2^n}$, $i < j \leq n$, tels que les coefficients non nuls de b_i soient $\mathbb{F}_2$-linéairement indépendants.

Exemple 4.4.9. Posons $n = 6$. Les vecteurs $b_1, b_2, b_3, b_4, b_5, b_6 \in \mathbb{F}_{2^6}^6$ sont de la forme :

$$\begin{aligned} b_1 &= (0, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6) \\ b_2 &= (\beta_2, 0, \beta_9, \beta_{10}, \beta_{11}, \beta_{12}) \\ b_3 &= (\beta_3, \beta_9, 0, \beta_{16}, \beta_{17}, \beta_{18}) \\ b_4 &= (\beta_4, \beta_{10}, \beta_{16}, 0, \beta_{23}, \beta_{24}) \\ b_5 &= (\beta_5, \beta_{11}, \beta_{17}, \beta_{23}, 0, \beta_{30}) \\ b_6 &= (\beta_6, \beta_{12}, \beta_{18}, \beta_{24}, \beta_{30}, 0). \end{aligned}$$

La dernière contrainte à respecter pour construire les vecteurs $b_i \in \mathbb{F}_{2^n}^n$ est que les coefficients de toute combinaison linéaire sur $\mathbb{F}_2$ de ces b_i engendre un sous-espace vectoriel de $\mathbb{F}_{2^n}$ de dimension $n-1$ sur $\mathbb{F}_2$. En d'autres termes, nous voulons nous assurer que les fonctions affines correspondant aux vecteurs du type $b_1 + b_2$, $b_1 + b_2 + b_3$, etc ... soient bien des fonctions 2-to-1. Cette procédure est réalisable assez simplement en effectuant une recherche en *'retour sur trace'* ("backtracking" en anglais).

Une fois que nous avons les vecteurs $b_i \in \mathbb{F}_{2^n}$, $1 \leq i \leq n$, nous possédons des fonctions affines L_i, $1 \leq i \leq n$, 2-to-1, vérifiant les conditions du corollaire 4.4.8 :

$$L_i(0) + L_i(\alpha_j) = L_j(0) + L_j(\alpha_i),$$

pour tout entiers $1 \leq i, j \leq n$ distincts.

Nous pouvons utiliser l'algorithme 4 pour trouver une fonction $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ telle que

$$\Delta_{\alpha_i} F = L_i,$$

pour tout entier $1 \leq i \leq n$. De plus, puisque nous nous sommes assurés par construction que les combinaisons linéaires des fonctions L_i sont des fonctions 2-to-1, la fonction F ne comporte que des différentielles qui sont des fonctions affines 2-to-1. La fonction F est donc quadratique APN sur $\mathbb{F}_{2^n}$.

L'implémentation de cette méthode pour des corps de petite dimension ($\mathbb{F}_{2^5}$, $\mathbb{F}_{2^6}$, $\mathbb{F}_{2^7}$, et $\mathbb{F}_{2^8}$) nous a permis d'obtenir en quelques minutes un très grand nombre de fonctions quadratiques APN.

4.5. Perspectives

Dans ce chapitre, nous avons vu des outils intéressants permettant de porter un regard différent sur les différentielles des fonctions dans $\mathbb{F}_{2^n}$. Cela nous a permis notamment de donner une nouvelle caractérisation des fonctions ayant des 0-structures linéaires, ainsi que de décrire une équivalence de fonctions relativement à leurs différentielles. Les invariants de cette nouvelle équivalence sont *a priori* différents des invariants de l'équivalence CCZ, ce qui en fait un sujet d'étude à approfondir.

L'application de ces résultats pour construire des fonctions APN quadratiques est la première étape d'une étude plus systématique. En effet, cette méthode peut certainement être améliorée pour en diminuer la complexité dans un premier temps. Dans un second temps, elle pourrait nous permettre de construire des fonctions de degré algébrique supérieur vérifiant des propriétés particulières.

De plus, il serait intéressant de comprendre comment ces outils pourraient aider les applications en cryptographie symétrique, par exemple pour calculer l'uniformité différentielle d'une fonction.

Cryptanalyse différentielle impossible

5. Concepts de base de la cryptanalyse différentielle

L'ÉVALUATION de la sécurité des primitives est différente en cryptographie asymétrique et en cryptographie symétrique. En cryptographie asymétrique, il existe des preuves et il est possible de réaliser des réductions de sécurité, c'est-à-dire de montrer que la cryptanalyse d'un système est équivalent à résoudre une instance d'un problème reconnu difficile. L'évaluation de la sécurité en cryptographie symétrique s'effectue au cas par cas. En effet, les pseudo-preuves de sécurité des algorithmes symétriques se basent sur une modélisation idéale et très peu réaliste. La fiabilité des chiffrements par blocs (ou des fonctions de hachage et des chiffrements à flot de la même manière) au niveau de la sécurité apportée ne dépend que de l'effort qui est fait par la communauté cryptographique pour tenter de les « casser ». Le but de la cryptanalyse est donc d'établir le niveau de confiance que l'on peut porter à une fonction cryptographique. Néanmoins, la confiance en une fonction n'est valable que pour les cryptanalyses dont elle a bénéficié. Il est donc important d'analyser les fonctions cryptographiques par un ensemble de cryptanalyses et de modèles le plus large possible, quitte à en développer de nouvelles formes. Les meilleures attaques connues sur des versions réduites nous permettent aussi de bien évaluer la marge de sécurité des primitives utilisées et d'adapter en conséquence, par exemple, le nombre de tours à utiliser pour les fonctions itérées.

Les formes les plus connues de cryptanalyses sont la cryptanalyse différentielle [18] et la cryptanalyse linéaire [193, 155] et toutes leurs variantes et généralisations. Pour de plus amples informations sur les chiffrements par blocs et leurs cryptanalyses, nous renvoyons le lecteur vers un ouvrage très didactique de Lars Knudsen et Matthew Robshaw : "The block cipher companion" [127].

En 1991, Eli Biham et Adi Shamir [18] introduisent une forme de cryptanalyse à clairs choisis applicable à tout type de chiffrement itératifs par blocs : la *cryptanalyse différentielle*. Au cours des années, cette forme de cryptanalyse a connu des généralisations et des applications légèrement différentes mais elle n'en reste pas moins une des plus performantes lorsqu'il s'agit d'attaquer des primitives de cryptographie symétrique.

Dans cette section, nous allons définir les notions essentielles et expliquer le principe des attaques différentielles.

5.1. Préliminaires

La cryptanalyse différentielle se classe parmi les *cryptanalyses statistiques*, c'est-à-dire, dans le cas des chiffrements par blocs, des attaques qui permettent de retrouver de l'information sur la clé à partir de l'observation d'un comportement non-aléatoire de la fonction de chiffrement. Ce "comportement non-aléatoire" est en général détecté par un algorithme appelé *distingueur*, qui est capable de décider si un jeu de données provient d'une fonction de chiffrement particulière ou d'une permutation aléatoire (*i.e.* tirée uniformément dans l'ensemble des permutations).

Bien que nous ayons déjà abordé quelques notions dans le chapitre introductif de cette thèse (chapitre 1), nous donnons ici de plus amples détails concernant les dérivées des boîtes-S d'un chiffrement par blocs et leurs utilisations.

Définition 5.1.1 (Différentielle)**.** Soit une fonction $F : \mathbb{F}_2^s \to \mathbb{F}_2^s$. Une *différentielle* de la fonction F est un couple $(\alpha, \beta) \in \mathbb{F}_2^s \times \mathbb{F}_2^s$, $(\alpha, \beta \neq (0,0))$ vérifiant

$$\Delta_\alpha F(X) = \beta, \quad \text{pour certains } X \in \mathbb{F}_2^s, \tag{5.1}$$

où $\Delta_\alpha F$ est la dérivée de la fonction F en $\alpha \in \mathbb{F}_2^s$. La *probabilité de la différentielle* (α, β) est la probabilité que $X \in \mathbb{F}_2^s$ vérifie l'égalité (5.1), que nous notons

$$P_X \left[\Delta_\alpha F(X) = \beta\right].$$

Lorsque nous parlons d'une fonction itérée, une différentielle peut être obtenue de plusieurs manières. En effet, la différentielle d'une fonction itérée est en quelque sorte la succession des différentielles intermédiaires des fonctions (tours) qui la composent.

Définition 5.1.2 (Chemin différentiel)**.** Soient des fonctions $F_i : \mathbb{F}_2^s \to \mathbb{F}_2^s$, $0 \leq i < R$. Un *chemin différentiel* de la fonction $F_{R-1} \circ \cdots \circ F_0$ est un $(R+1)$-uplet

$$(\alpha_0 \to \alpha_R) = (\alpha_0, \ldots, \alpha_R) \in (\mathbb{F}_2^s)^{R+1},$$

tel que (α_i, α_{i+1}) est une différentielle de la fonction F_i, pour tout $0 \leq i < R$.

La connaissance de la probabilité d'une différentielle d'une fonction de chiffrement est essentielle pour la cryptanalyse différentielle. Cependant, il est relativement difficile d'évaluer une telle probabilité directement pour une fonction itérée. La proposition suivante, due à Xuejia Lai et James Massey [135], permet de calculer la probabilité théorique d'un chemin différentiel.

Proposition 5.1.3 ([135])**.** *Soient des fonctions bijectives paramétrées $F_i : \mathbb{F}_2^s \to \mathbb{F}_2^s$, $0 \leq i < R$, telles que la probabilité de la différence en sortie sachant la différence en entrée ne dépende pas des sous-clés indépendantes qui les paramètrent. La probabilité du chemin différentiel (α_0, α_R) de la fonction[1] $F_{R-1} \circ \cdots \circ F_0$, est le produit des probabilités des différentielles successives :*

$$\begin{aligned} &P_X \left[\Delta_{\alpha_0}(F_{R-1} \circ \cdots \circ F_0)(X) = \alpha_R\right] \\ &\quad = P_X \left[\Delta_{\alpha_0} F_0(X) = \alpha_1\right] \times P_X \left[\Delta_{\alpha_1}(F_{R-1} \circ \cdots \circ F_1)(X) = \alpha_R\right] \\ &\quad = P_X \left[\Delta_{\alpha_0} F_0(X) = \alpha_1\right] \times \cdots \times P_X \left[\Delta_{\alpha_{R-1}} F_{R-1}(X) = \alpha_R\right]. \end{aligned}$$

Si nous connaissions l'ensemble des chemins différentiels composant une différentielle et leur probabilité, nous pourrions évaluer la probabilité théorique de la différentielle de la fonction itérée. En effet, nous supposons que les sous-clés de tours sont indépendantes, impliquant donc que les probabilités des chemins au sein d'une même différentielle le sont aussi.

Proposition 5.1.4. *La probabilité de la différentielle est égale à la somme des probabilités de ses chemins différentiels.*

[1] Un tel chiffrement itératif est appelé *de Markov* [135].

En pratique, comme il est assez difficile de connaître tous les chemins différentiels de probabilité non nulle associés à une différentielle, nous trouvons un (ou quelques) chemin différentiel qui va avoir une probabilité élevée. Nous approximons ensuite la somme théorique de la proposition ci-dessus avec cette probabilité, en supposant qu'il s'agit du terme dominant (dans tous les cas, il s'agit d'une borne).

Distingueur différentiel

La découverte d'une différentielle sur une fonction de chiffrement $E_K : \mathbb{F}_2^s \to \mathbb{F}_2^s$ ayant une bonne probabilité, c'est-à-dire $> 1/2^s$ qui correspond à la probabilité uniforme, nous permet d'avoir un distingueur sur cette fonction de chiffrement. En effet, si une différentielle a une probabilité élevée, nous sommes capables de dire que la fonction de chiffrement n'est pas une fonction bijective choisie uniformément parmi l'ensemble des permutations de $\mathbb{F}_2^s$. À la section 5.3, nous verrons comment exploiter ce biais sur $R-1$ tours d'une fonction itérée pour obtenir de l'information sur la clé de chiffrement utilisée pour R tours.

La cryptanalyse différentielle a été développée dans plusieurs directions dont nous donnons ici les idées principales.

Différentielles tronquées. Le concept de *différence tronquée*, introduit en 1994 par Lars Knudsen [125] tend à généraliser celui de différence. Lorsque nous parlons de différence, le mot *tronquée* signifie que nous *relâchons* des contraintes sur la différence que nous considérons. En d'autres termes, nous autorisons que les différences soient portées par des *mots* de plusieurs bits, plutôt que par des bits.

La notion de différentielle peut être adaptée aux *différentielles tronquées*. Il s'agit dans ce cas d'un ensemble de différences en entrée d'une fonction, et d'un ensemble de différences en sortie.

Définition 5.1.5 (Différentielle tronquée)**.** Soit une fonction $F : \mathbb{F}_2^s \to \mathbb{F}_2^s$. Une *différentielle tronquée* de la fonction F est un couple (δ_0, δ_1) d'ensembles de différences dans $\mathbb{F}_2^s$ vérifiant

$$\forall \alpha \in \delta_0, \Delta_\alpha F(X) \in \delta_1, \quad \text{pour certains } X \in \mathbb{F}_2^s.$$

La probabilité d'une différentielle tronquée est définie par

$$P_{X,\alpha\in\delta_0}[\Delta_\alpha F(X) \in \delta_1] = P_X[\Delta_\alpha F(X) \in \delta_1 \mid \alpha \in \delta_0], \tag{5.2}$$

où $P[A \mid B]$ est la probabilité conditionnelle de l'évènement A sachant l'évènement B.

Étant donné que les différentielles tronquées s'expriment en termes de probabilités conditionnelles, nous pouvons utiliser le théorème de Bayes :

Théorème 5.1.6 (Théorème de Bayes)**.** *Avec les mêmes notations que la définition ci-dessus, nous avons l'égalité :*

$$P_X[\Delta_\alpha F(X) \in \delta_1 \mid \alpha \in \delta_0] = \frac{\#\delta_1}{\#\delta_0} P_X[\alpha \in \delta_0 \mid \Delta_\alpha F(X) \in \delta_1], \tag{5.3}$$

où $\#\delta_i$, désigne la taille de l'ensemble δ_i, avec $i = 0, 1$.

La notion de chemin différentiel se transpose naturellement dans le cas des différentielles tronquées.

Les différentielles tronquées sont très pratiques pour la cryptanalyse de primitives en cryptographie symétrique. Lors de la compétition SHA-3, visant à standardiser une nouvelle fonction de hachage, les différentielles tronquées se sont révélées très efficaces contre les candidats basés sur l'AES. En effet, les différentielles tronquées permettent souvent de construire des chemins différentiels qui ne dépendent pas des valeurs que peuvent prendre les boîtes-S, mais seulement du fait qu'elles sont *actives*[2] ou non dans ce chemin. Cela peut permettre à un attaquant de profiter de la mauvaise diffusion d'un système de cryptographie symétrique.

Différentielles impossibles. Le concept de *différentielle impossible*, introduit indépendamment par Lars Knudsen [126] et Eli Biham, Alex Biryukov et Adi Shamir [17] en 1999, est d'utiliser une différentielle de probabilité zéro pour arriver à déterminer les clés qui *ne* peuvent *pas* être la clé ayant servi au chiffrement. Ce type de cryptanalyse étant le sujet du chapitre suivant, nous invitons le lecteur à le consulter pour le comprendre en détail et découvrir des méthodes originales permettant de l'améliorer.

Différentielles d'ordre supérieur. La cryptanalyse d'ordre supérieur est basée sur les dérivées d'ordre supérieur (voir chapitre 4) introduites par Xuejia Lai [134] en 1994 puis utilisées par Lars Knudsen [125] pour cryptanalyser le DES. La cryptanalyse différentielle d'ordre supérieur est souvent reliée à l'étude du degré algébrique de la fonction de chiffrement. En effet, l'attaquant cherche en général à trouver par quel espace il peut dériver sa fonction de chiffrement pour obtenir une différence nulle entre tous les chiffrés (voir section 4.2). La connaissance du degré algébrique de la fonction de chiffrement fournit naturellement de tels espaces. Il existe plusieurs type de distingueurs utilisant des dérivées d'ordre supérieur, par exemple les "zero-sum distinguishers" [10] ("distingueurs à somme nulle" en français) dont une étude approfondie fut effectuée par Anne Canteaut et Christina Boura [31], ou bien encore les "distingueurs cube" [85].

5.2. Notions importantes

Au chapitre 1, nous avons abordé les concepts de *substitution* et de *diffusion* d'un chiffrement par blocs. Dans cette section nous allons voir plus de détails concernant ces deux notions.

Nous avons vu que la fonction de substitution (non-linéaire) d'un chiffrement par blocs devait avoir une uniformité différentielle basse pour mieux résister à la cryptanalyse différentielle. Une donnée importante lorsque nous considérons les propriétés différentielles d'une boîte-S est la *table de distribution des différences*.

Définition 5.2.1 (Table de distribution des différences)**.** La *table de distribution des différences* d'une boîte-S, $S : \mathbb{F}_2^s \to \mathbb{F}_2^s$, est une table contenant toutes les valeurs $\delta_S(\alpha, \beta)$, $\alpha, \beta \in \mathbb{F}_2^s$.

Pour des boîtes-S de petite dimension (typiquement pour $n = 4$ ou 8), il est tout à fait raisonnable de considérer que nous pouvons calculer leur table de distribution

[2] Une boîte-S est active si la différence à l'entrée ou à la sortie est non-nulle

des différences. Outre la connaissance évidente de l'uniformité différentielle, la table de distribution des différences nous fournit surtout les *transitions différentielles* d'une boîte-S. En d'autres termes, une fois la table de distribution des différences calculée pour une boîte-S, $S : \mathbb{F}_2^s \to \mathbb{F}_2^s$, une simple recherche dans cette table nous indique si l'équation

$$\Delta_\alpha S(x) = \beta,$$

pour $\alpha \neq 0, \beta \in \mathbb{F}_2^s$, possède des solutions ou non. De plus, puisque nous possédons le nombre exact de solutions de cette équation, nous avons donc aussi la probabilité de la différentielle $(\alpha \to \beta)$ de la fonction S.

Exemple 5.2.2. Soit une boîte-S, $S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$, dont la table de distribution des différences est donnée à la table 5.1. Nous pouvons lire dans cette table, que la fonction S est bijective car $\Delta_\alpha S(x) \neq 0 \ \forall \alpha \in \mathbb{F}_2^s \backslash \{0\}^s$, et que son uniformité différentielle est $\delta(S) = 10$, qui est la plus grande valeur de la table si l'on exclut $\alpha = 0$.

$\alpha\backslash\beta$	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
1	.	.	6	.	.	.	.	2	.	2	.	.	2	.	4	.
2	.	6	6	.	.	.	.	.	.	2	2	.	.	.	.	.
3	.	.	.	6	.	2	.	.	2	.	.	.	4	.	2	.
4	.	.	.	2	.	2	4	.	.	2	2	2	.	.	2	.
5	.	2	2	.	4	.	.	4	2	.	.	2	.	.	.	.
6	.	.	2	.	4	.	.	2	2	.	2	2	2	.	.	.
7	.	.	.	.	.	4	4	.	2	2	2	2	.	.	.	.
8	.	.	.	.	.	2	.	2	4	.	.	4	.	2	.	2
9	.	2	.	.	.	2	2	2	.	4	2	.	.	.	.	2
a	.	.	.	.	2	2	.	.	.	4	4	.	2	2	.	.
b	.	.	.	2	2	.	2	2	2	.	.	4	.	.	2	.
c	.	4	.	2	.	2	.	.	2	.	.	.	.	.	6	.
d	.	.	.	.	.	.	2	2	.	.	.	.	6	2	.	4
e	.	2	.	4	2	.	.	.	.	.	2	.	.	.	.	6
f	.	.	.	.	2	.	2	.	.	.	.	.	.	10	.	2

TABLE 5.1. : Table de distribution des différences de la boîtes-S, notée S.

Un critère permet de quantifier en partie la *qualité* de diffusion d'une fonction linéaire : il s'agit du 'branch-number' ("nombre d'embranchements" en français).

Définition 5.2.3 ('branch-number')**.** Soit une matrice M de taille $m \times m$ à coefficients dans $\mathbb{F}_{2^n}$. Quels que soient les vecteurs non nuls $X = (x_0, \ldots, x_{m-1}) \in \mathbb{F}_{2^n}^m$ et $Y = (y_0, \ldots, y_{m-1}) \in \mathbb{F}_{2^n}^m$ tels que

$$M \cdot X = Y,$$

le nombre minimal de la somme de leurs coordonnées non nulles est appelé le *'branch-number'*, noté **b**, de la matrice M. C'est-à-dire

$$\mathbf{b} = |\{x_i \mid x_i \neq 0\}| + |\{y_i \mid y_i \neq 0\}|.$$

Le 'branch-number' maximal que la matrice M puisse atteindre est $\mathbf{b} = m + 1$.

Un 'branch-number' maximal, est un gage de qualité pour la diffusion, car cela signifie que la plus petite différence en entrée de la fonction va se propager sur tous les mots de la sortie, comme nous pouvons le voir avec l'exemple 5.2.4.

Les matrices ayant un 'branch-number' maximal sont en fait dérivées de matrices génératrices de code *MDS* [3]. Ces matrices ayant un 'branch-number' maximal offrent évidemment la meilleure diffusion possible. Bien qu'importante, l'étude des codes MDS est bien trop vaste pour être traitée dans ce document, nous renvoyons donc le lecteur vers des ouvrages fournissant plus d'informations [150, 80]. Nous noterons simplement que la recherche et la construction de matrices génératrices de codes MDS est assez difficile. De plus, l'implantation de matrices MDS dans des primitives cryptographiques est parfois coûteuse en raison des contraintes matérielles et/ou logicielles. C'est pourquoi il n'est pas étonnant que des chiffrements par blocs utilisent des fonctions de diffusion n'ayant pas un 'branch-number' maximal.

Exemple 5.2.4. Soit une matrice M, de taille 4×4, à coefficients dans $\mathbb{F}_2^8$ et de 'branch-number' maximal (*i.e.* 5). Soit un vecteur $(\alpha, 0, 0, 0) \in (\mathbb{F}_2^8)^4$ non nul. Nous savons que les coordonnées du vecteur obtenu en effectuant l'opération

$$M \cdot \begin{pmatrix} \alpha \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix}$$

sont toutes non nulles. Autrement dit, nous savons qu'une différence $(\alpha, 0, 0, 0)$ se propage en une différence $(\beta_0, \beta_1, \beta_2, \beta_3) \in (\mathbb{F}_2^8 \backslash \{0\})^4$ par la matrice M.

5.3. Cryptanalyse différentielle sur le dernier tour

Nous allons maintenant décrire le fonctionnement et les principes de base de la cryptanalyse différentielle sur le dernier tour d'un chiffrement par blocs itératif. Notons que l'attaque statistique sur le dernier tour d'un système cryptographique itéré est un procédé pouvant être appliqué plus généralement (cryptanalyse linéaire par exemple) que dans le cas des cryptanalyses différentielles. Le principe reste cependant le même.

La cryptanalyse différentielle sur le dernier tour représente la famille des cryptanalyses différentielles dans sa forme la plus simple et la plus classique. Du fait de leurs efficacités, ces attaques ont bénéficiées (et subissent toujours) des efforts de la communauté cryptographique pour les améliorer, les généraliser et trouver de nouvelles applications des différences pour évaluer la sécurité des primitives en cryptographie symétrique. Dans le chapitre suivant, nous verrons les détails de l'utilisation des différentielles de probabilité zéro, dites *impossibles*, pour cryptanalyser des chiffrements par blocs construits avec des schémas de Feistel généralisés. Nous verrons aussi des développements et des améliorations de cette cryptanalyse.

La cryptanalyse différentielle est une attaque à clair choisi, cela signifie que nous supposons que l'attaquant est capable de construire tous les messages qu'il veut (dans la limite des messages possibles) et de les chiffrer par la fonction de chiffrement, E_K dont il souhaite retrouver la clé K.

[3] "Maximum distance separable" en anglais. Ce sont des codes qui atteignent la borne de *Singleton*. Dans le cas des codes linéaires $[n, k, d]$: $d = n - k + 1$

Supposons que la fonction de chiffrement est une fonction itérée sur R tours. Si l'attaquant possède une différentielle $(\alpha, \beta) \in (\mathbb{F}_2^s)^2$ sur $R-1$ tours de la fonction de chiffrement avec une bonne probabilité $p > 1/2^s$, où s est la taille (en bit) de l'état, alors il peut monter l'attaque en procédant ainsi :

1. il chiffre N paires de données ayant la différence α ;
2. il déchiffre ensuite partiellement chacune de ces paires d'un tour, avec une sous-clé candidate (et d'éventuelles hypothèses sur cette sous-clé) ;
3. il incrémente un compteur correspondant à la sous-clé candidate à chaque fois que la différence des paires ayant été déchiffrées sur un tour est β ;
4. une fois toutes les sous-clés testées, il regarde quel compteur à une valeur proche de $N \times p$, qui correspond à la bonne sous-clé. Les autres compteurs auront une valeur proche de ce qui aurait été obtenu avec une permutation aléatoire (i.e $N \times 1/2^s$).

La cryptanalyse différentielle sur le dernier tour d'un chiffrement par blocs est schématisé à la figure 5.1.

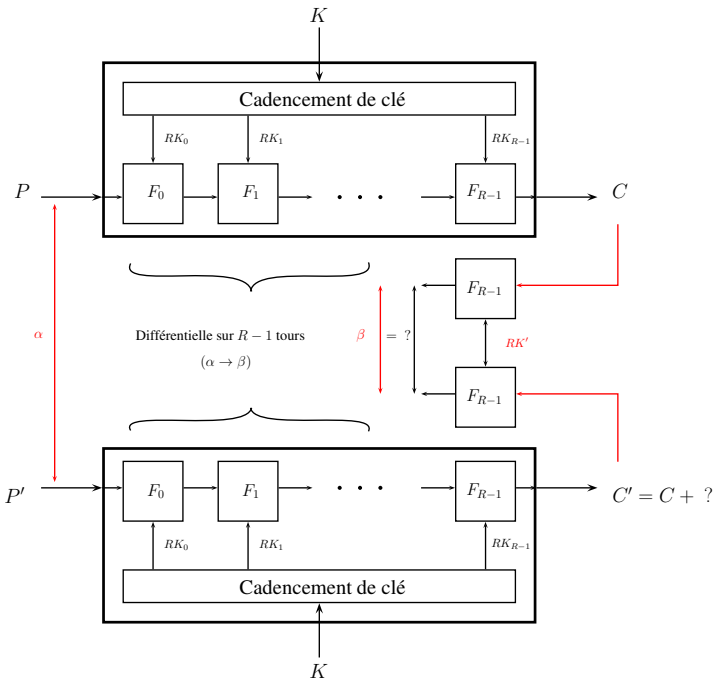


FIGURE 5.1. : Principe de la cryptanalyse différentielle sur le dernier tour.

Notons que ce type d'attaque peut être étendu à plus qu'un tour de chiffrement (voir notamment [26]). La borne inférieure sur la complexité en temps est $2^{|RK'|}$. Lorsque nous parlerons de cryptanalyse différentielle impossible dans le chapitre suivant, nous verrons que ce principe est repris lorsque nous étendons une différentielle de probabilité zéro par des chemins différentiels.

6. Cryptanalyse différentielle impossible

Les applications en environnements contraints (RFID et réseaux de capteurs entre autres) ont entraîné l'apparition de la cryptographie dite « à bas coût »("lightweight" en anglais). En conséquence, une multitude de fonctions aux conceptions parfois innovantes sont apparues depuis l'an 2000. Certaines d'entre elles ont même été standardisées [7, 184, 28]. Cependant, un bas coût implique bien souvent de rogner sur les marges de sécurité en diminuant la taille de la clé (dans le cas des chiffrements par blocs) ou bien le nombre de tours effectués (dans le cas des systèmes itérés) par exemple. Il est important que les cryptanalystes cherchent à déceler les moindres faiblesses de ces chiffrements qui tendent à être utilisés de plus en plus dans la vie courante. L'évaluation de la sécurité d'un système de cryptographie symétrique passe par des attaques qui ont souvent des complexités élevées pour s'assurer de la véritable marge de sécurité qu'ils offrent. Dans le cas des systèmes itérés, il est important de connaître le nombre de tours qu'il est possible d'attaquer en ayant une complexité inférieure à la recherche exhaustive du secret.

Du fait de leur simplicité d'implantation, les schémas de Feistel (généralisés) sont largement utilisés pour les constructions à bas coût. En effet, le croisement des branches à la fin d'un tour d'un schéma de Feistel autorise l'utilisation de fonctions internes sur des états plus petits. Néanmoins, une contrepartie de cette simplicité se paye souvent au niveau de la qualité de la diffusion. La *cryptanalyse différentielle impossible*, exploitant beaucoup les défauts de diffusion, est particulièrement efficace contre les schémas de Feistel utilisés dans des chiffrements à bas coût. C'est pour cela que nous nous concentrons dans ce chapitre à la cryptanalyse différentielle impossible des Feistel généralisés, bien que cette cryptanalyse puisse s'appliquer à d'autres types de construction (par exemple les réseaux de substitution-permutation).

La cryptanalyse différentielle impossible fut introduite indépendamment par Lars Knudsen [126] et Eli Biham, Alex Biryukov et Adi Shamir [17] en 1999. Elle a été utilisée et analysée à de nombreuses reprises et est aujourd'hui considérée comme l'une des attaques les plus efficaces contre les chiffrements par blocs itératifs. Néanmoins, ce type de cryptanalyse est très technique, et certains points restent assez « obscurs ». Nous avons remarqué que de nombreuses applications de cette attaque contiennent des erreurs, ou des défauts d'évaluation de complexité comme décrit à la table 6.1, qui n'est pas une liste exhaustive.

La mise en place d'une cryptanalyse différentielle impossible est souvent très technique, beaucoup de paramètres sont à considérer. La façon de choisir les données claires/chiffrées, la manière d'évaluer la quantité de données nécessaire à l'attaque, la complexité en temps de l'attaque ou encore les paramètres amenant les attaques les plus efficaces sont autant de points délicats qu'il est important de maîtriser pour optimiser une cryptanalyse différentielle impossible. Cependant, jusqu'à présent, aucune méthode unifiée ne permet de répondre à toutes les questions d'un attaquant désirant utiliser ces attaques contre un chiffrement par blocs. La plupart des cryptanalyses différentielles impossibles publiées sont très complexes et utilisent des détails spécifiques aux fonctions de chiffrement qu'elles considèrent. En conséquence, les méthodes varient d'un travail à l'autre et il n'est pas

Algorithme	Référence	Type d'erreur	Gravité	Mis à jour
CLEFIA-128 (sans WK)	[219]	plus de données que le codebook	attaque non valide	[194]
CLEFIA-128	[195]	invérifiable sans implémentation	-	[22]
Camellia (sans FL/FL^{-1})	[212]	erreur de calcul	attaque non valide	section 6.6 idem à [211]
Camellia-128	[211]	plus long que la recherche exhaustive	attaque non valide	[152]
Camellia (sans FL/FL^{-1})	[149]	petites erreurs de calcul	valide une fois corrigée	[212]
LBlock	[157]	petites erreurs de calcul	valide une fois corrigée	[158]
Simon	[6]	plus de données que le codebook	attaques non valides	[6, Table 1]
Simon	[3, 4]	erreurs de calcul	attaques non valides	section 6.8

Table 6.1. : Résumé des défauts et des erreurs dans les cryptanalyses différentielles impossibles sur CLEFIA-128, Camellia, LBlock et Simon.

étonnant de voir apparaître des erreurs dans les analyses.

Les travaux présentés dans ce chapitre ont été effectué en commun avec Christina Boura, Marine Minier et María Naya-Plasencia [34, 35, 36]. Nous présentons tout d'abord la cryptanalyse différentielle impossible, son fonctionnement et ses mécanismes. Dans un deuxième temps, nous formalisons cette cryptanalyse et nous l'améliorons. La formalisation commence par détailler les étapes de cette cryptanalyse. Dans un même temps, nous expliquons comment évaluer les paramètres inhérents à l'attaque que nous souhaitons mener.

Nous fournissons ensuite des formules générales pour évaluer les complexités (temps, données, mémoire) d'une telle attaque. Ces formules, dont les variables sont les différents paramètres dont nous venons de parler, nous permettent d'optimiser et de simplifier (évitant ainsi les erreurs d'évaluation) la procédure pour monter des cryptanalyses différentielles impossibles.

Avec ces formules, nous avons pu développer un outil automatique nous permettant de simplifier le travail qu'un cryptanalyste doit fournir s'il veut trouver les meilleurs paramètres et les meilleurs compromis de complexité contre un chiffrement par blocs utilisant un schéma de Feistel, pour la cryptanalyse différentielle impossible. Ce programme demande à son utilisateur de décrire les spécifications du schéma de Feistel généralisé qu'il veut considérer, et il retourne les meilleures cryptanalyses différentielles impossibles qu'il est possible de monter contre ce chiffrement, tant en nombre de tours atteints qu'en complexité temps/données.

Nous avons aussi proposé des techniques permettant d'améliorer ces complexités ainsi que le nombre de tours qu'il est possible d'attaquer. Parmi ces techniques, nous formalisons l'usage des différentielles impossibles multiples, introduites par Tsunoo et al. [197] en 2008, et les utilisons pour réduire la complexité en données d'une cryptanalyse différentielle impossible. Nous introduisons aussi une nouvelle technique qui diminue significativement

la complexité en temps d'une cryptanalyse différentielle impossible : la *state-test* technique (pour "test d'état"). Cette technique consiste à vérifier les valeurs de l'état interne plutôt que de tester certains bits de sous-clé.

Finalement, pour illustrer la force de notre approche des cryptanalyses différentielles impossibles, nous présentons des applications de nos optimisations à des chiffrements par blocs utilisant des schémas de Feistel généralisés : CLEFIA, Camellia, LBlock et Simon. Ces applications nous ont permis de proposer les meilleures attaques connues contre ces chiffrements dans la plupart des cas.

6.1. Déroulement d'une cryptanalyse différentielle impossible

La cryptanalyse différentielle impossible et la cryptanalyse différentielle "classique" partagent beaucoup de points communs, notamment d'étudier la propagation des différences dans un chiffrement par blocs itératif. Cependant, elles n'en restent pas moins différentes sur l'utilisation qui est faite de ces différentielles. Comme nous l'avons vu au chapitre précédent, lors d'une cryptanalyse différentielle, nous cherchons et exploitons une différentielle ayant une grande probabilité pour tester les clés candidates. À l'inverse, la cryptanalyse différentielle impossible utilise une différentielle de probabilité nulle (d'où sa dénomination : "impossible") pour écarter des clés de l'ensemble des clés candidates.

Dans cette section, nous allons détailler les deux étapes principales que nous nous devons d'appliquer afin de réaliser une cryptanalyse différentielle impossible.

La première étape est de trouver et de placer une différentielle impossible (*i.e.* de probabilité zéro). Lors de la deuxième étape, nous étendons cette différentielle impossible par des chemins différentiels dans le sens du chiffrement et/ou du déchiffrement. Ces chemins différentiels nous servent à *filtrer* les clés : une clé qui amène à la différentielle impossible ne peut être celle qui a servi au chiffrement, nous pouvons donc la retirer de l'ensemble des clés candidates.

Enfin, nous montrerons que l'évaluation de la complexité en temps, données et mémoire, d'une telle attaque peut être mise en équation, dont les variables sont les paramètres de l'attaque que nous expliciterons. Avant cela, nous spécifions dans la sous-section suivante un *chiffrement exemple*, qui nous servira à illustrer nos propos tout au long de ce chapitre.

6.1.1. Spécifications d'un chiffrement exemple

Nous prenons pour notre chiffrement par blocs itératif *exemple* un schéma de Feistel à deux branches comportant quatre mots de 4 bits (des quartets) chacune. La fonction interne est composée des trois étapes suivantes :

- un xor bit à bit de la sous-clé de tour RK_i et de l'état ;
- une fonction non-linéaire S composée de 4 boîtes-S (S_0, S_1, S_2, S_3) inversibles de 4 bits ;
- une matrice linéaire M inversible, de taille 4×4 quartets, telle que son 'branch-number' n'est pas maximal (< 5). Nous la prendrons de la forme :

$$M = \begin{pmatrix} * & * & * & 0 \\ * & * & 0 & * \\ * & 0 & * & * \\ 0 & * & * & * \end{pmatrix},$$

où $*$ vaut un coefficient non nul de $\mathbb{F}_2^4$.

Nous ne précisons pas plus les boîtes-S ou la matrice M car nous ne les utiliserons qu'avec des différences tronquées. De plus, pour des raisons de simplicité, nous supposons que les sous-clés sont indépendantes, ou tout du moins que nous ne pouvons pas tirer parti de l'algorithme de cadencement de clé pour trouver des relations entre les différents bits des clés de tours. Un tour de notre chiffrement *exemple* est décrit à la figure 6.1.

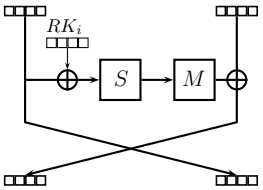


FIGURE 6.1. : Un tour du chiffrement *exemple*.

Notation 6.1.1. Nous notons L et R les branches gauche et droite respectivement, de notre chiffrement *exemple*. Nous utilisons la notation $L\rfloor_i$ pour parler du i-ème quartet en partant de la gauche, de la branche L.

Ce chiffrement, bien que simplifié, n'en reste pas moins représentatif des chiffrements par blocs existants.

6.1.2. Trouver une différentielle impossible

Dans un premier temps nous devons mettre en place une différentielle impossible. En 1999, Eli Biham, Alex Biryukov et Adi Shamir [17] introduisent une méthode originale pour trouver une différentielle impossible, qu'ils appellent *miss-in-the-middle* ("rater au milieu" en français). Cette méthode, particulièrement efficace, consiste à propager une différence dans le sens du chiffrement et une différence dans le sens du déchiffrement, et de trouver une incompatibilité au bout du nombre maximal de tours possible.

Notation 6.1.2. Nous notons $(\delta_X \nrightarrow_{r_\delta} \delta_Y)$ une différentielle (δ_X, δ_Y) ayant une probabilité zéro après r_δ tours d'un chiffrement par blocs itératif.

Exemple 6.1.3. Dans cet exemple, nous allons voir une application de la technique du *miss-in-the-middle* sur un schéma de Feistel classique. Nous allons pour cela utiliser le chiffrement *exemple* que nous avons décrit à la sous-section 6.1.1. La figure 6.2 décrit une différentielle impossible sur 5 tours de ce chiffrement, placée entre les tours 2 et 6. À partir de la différence δ_X, nous ne pouvons pas avoir la différence δ_Y au bout de 5 tours de chiffrement. En effet, à la sortie du troisième tour, nous voyons que le quartet le plus à droite, $R_3\rfloor_3$, doit avoir une différence nulle dans le sens du chiffrement, et une différence non nulle dans le sens du déchiffrement, ce qui n'est pas possible.

Bien qu'*a priori*, trouver une différentielle impossible atteignant un grand nombre de tours est une étape cruciale pour réaliser une cryptanalyse différentielle impossible, nous ne traiterons pas cette partie dans ce chapitre. En effet, de nombreux travaux ont été réalisés, depuis l'introduction de la technique *miss-in-the-middle*, pour essayer de théoriser le nombre de tours qu'il est possible d'atteindre suivant les différentes constructions des chiffrements [123, 122]. Néanmoins, il peut s'avérer que certains cas particuliers rendent

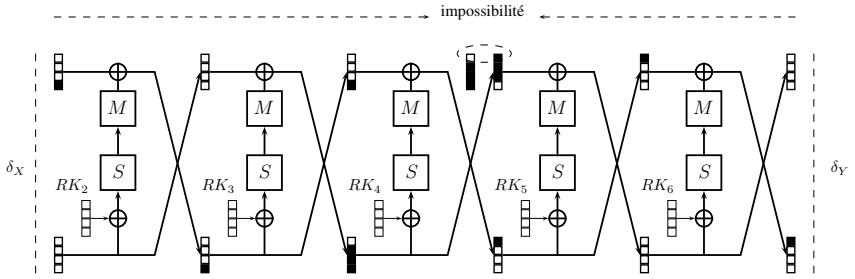


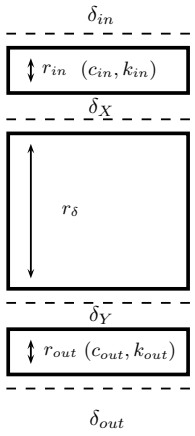
FIGURE 6.2. : Différentielle impossible sur 5 tours du chiffrement *exemple*.
■ : quartet avec une différence non nulle.

la recherche de différentielles impossibles un petit peu plus compliquée. Par exemple, pour le chiffrement par blocs Camellia, les fonctions FL/FL^{-1} présentes sur les branches du schéma de Feistel tous les six tours rendent la tâche plus difficile pour trouver une différentielle impossible (voir section 6.6).

Il peut aussi exister plusieurs différentielles impossibles atteignant le même nombre de tours pour un chiffrement donné. Certaines de ces différentielles impossibles peuvent partager des similitudes, comme par exemple : avoir la même différence en entrée, être la rotation circulaire d'une autre différentielle impossible, etc ... Nous verrons à la sous-section 6.4.1, ainsi que lorsque nous aborderons les applications, comment exploiter ces différentielles impossibles multiples.

6.1.3. Étendre par des chemins différentiels

Introduisons maintenant les notations dont nous nous servirons tout au long de ce chapitre. Dans toute la suite, s désignera la taille en bit d'un bloc.



- δ_X, δ_Y : différences en entrée (resp. en sortie) de la différentielle impossible.
- r_δ : nombre de tours de la différentielle impossible.
- δ_{in}, δ_{out} : différences tronquées possibles sur les clairs (resp. chiffrés) du chiffrement.
- r_{in} : nombre de tours du chemin différentiel $(\delta_{in} \to \delta_X)$.
- r_{out} : nombre de tours du chemin différentiel $(\delta_{out} \to \delta_Y)$.

Le paramètre c_{in} (resp. c_{out}) correspond à l'opposé du logarithme (en base 2) de la probabilité de la différentielle $(\delta_{in} \to \delta_X)$ (resp. $(\delta_{out} \to \delta_Y)$). Cela signifie que c_{in} (resp. c_{out}) vérifie

$$P_{x,d_{in}\in\delta_{in}}\left[\Delta_{d_{in}}\left(F_{r_{in}-1}\circ\cdots\circ F_0\right)(x)\in\delta_X\right]=\frac{1}{2^{c_{in}}},$$

$$\left(\text{resp.} \quad P_{x,d_{out}\in\delta_{out}}\left[\Delta_{d_{out}}\left(F^{-1}_{r_{in}+r_\delta}\circ\cdots\circ F^{-1}_{r_{in}+r_\delta+r_{out}-1}\right)(x)\in\delta_Y\right]=\frac{1}{2^{c_{out}}}\right)$$

où F_i est le i-ème tour du chiffrement. En d'autres termes, c_{in} (resp. c_{out}) correspond au nombre de conditions, comptées en bits, qui doivent être respectées pour que des données claires (resp. chiffrées) vérifient la différentielle $(\delta_{in}\to\delta_X)$ (resp. $(\delta_{out}\to\delta_Y)$). Le paramètre k_{in} (resp. k_{out}) désigne quant à lui le plus petit ensemble des bits d'informations des sous-clés intervenant dans la différentielle $(\delta_{in}\to\delta_X)$ (resp. $(\delta_{out}\to\delta_Y)$).

Nous supposons maintenant qu'une différentielle impossible $(\delta_X \nrightarrow_{r_\delta} \delta_Y)$ d'un nombre de tours maximal, r_δ, a été placée entre les tours r_{in} et $r_\delta + r_{in}$. Nous ne décrivons ici que la procédure pour trouver les paramètres (c_{in}, k_{in}) du chemin différentiel $(\delta_{in}\to\delta_X)$, le principe étant le même pour les paramètres (c_{out}, k_{out}) de la différentielle $(\delta_{out}\to\delta_Y)$. Nous notons par $|\delta_{in}|$ (resp. $|\delta_{out}|$), le logarithme (en base 2) de la taille de l'ensemble des différences tronquées possibles en entrée (resp. sortie) du chiffrement. Cela signifie que nous pouvons obtenir $2^{|\delta_{in}|}-1$ valeurs de différences non nulles avec les différences tronquées δ_{in}. Nous allons utiliser l'égalité de Bayes (voir théorème 5.1.6) utilisant les probabilités conditionnelles des différentielles tronquées (voir définition 5.1.5) :

$$\begin{aligned}
\frac{1}{2^{c_{in}}} &= P_{x,d_{in}\in\delta_{in}}\left[\Delta_{d_{in}}\left(F_{r_{in}-1}\circ\cdots\circ F_0\right)(x)\in\delta_X\right]\\
&= P_x\left[\Delta_{d_{in}}\left(F_{r_{in}-1}\circ\cdots\circ F_0\right)(x)\in\delta_X \mid d_{in}\in\delta_{in}\right]\\
&= \frac{2^{|\delta_X|}}{2^{|\delta_{in}|}}P_x\left[d_{in}\in\delta_{in} \mid \Delta_{d_{in}}\left(F_{r_{in}-1}\circ\cdots\circ F_0\right)(x)\in\delta_X\right]\\
&= \frac{2^{|\delta_X|}}{2^{|\delta_{in}|}}P_x\left[\Delta_{d_X}\left(F_0^{-1}\circ\cdots\circ F^{-1}_{r_{in}-1}\right)(x)\in\delta_{in} \mid d_X\in\delta_X\right].
\end{aligned}$$

Nous propageons donc les différences $d_X\in\delta_X$ sur un tour en déchiffrant. Notons δ_X^1 l'ensemble des différences obtenues, nous avons donc :

$$\Delta_{d_X}F^{-1}_{r_{in}-1}(x)\in\delta_X^1,$$

et notons c_{in}^1 la valeur vérifiant

$$P_x\left[\Delta_{d_X^1}F_{r_{in}-1}(x)\in\delta_X \mid d_X^1\in\delta_X^1\right]=\frac{1}{2^{c_{in}^1}}.$$

La valeur c_{in}^1 correspond au nombre de bits qui ont besoin d'être maîtrisés pour pouvoir vérifier la différentielle.

Nous pouvons ainsi remonter r_{in} tours. La probabilité de ce chemin différentiel est comme vu à la proposition 5.1.3 :

$$\begin{aligned}
&P_x\left[\Delta_{d_{in}}\left(F_{r_{in}-1}\circ\cdots\circ F_0\right)(x)\in\delta_X \mid d_{in}\in\delta_{in}\right]\\
&\quad= P_x\left[\Delta_{d_{in}}F_0(x)\in\delta_X^{r_{in}-1} \mid d_{in}\in\delta_{in}\right]\times\cdots\times P_x\left[\Delta_{d_X^1}F_{r_{in}-1}(x)\in\delta_X \mid d_X^1\in\delta_X^1\right]\\
&\quad= 2^{-c_{in}}.
\end{aligned}$$

L'exemple suivant va nous permettre de mieux visualiser.

Exemple 6.1.4. Reprenons notre chiffrement *exemple* de la section 6.1.1. La différentielle $(\delta_{in} \to \delta_X)$ est décrit à la figure 6.3. Nous partons d'une différence $\delta_X = (0,0,0,0)|(\alpha,0,0,0)$ au deuxième tour. Nous remontons deux tours avec une probabilité 1 pour obtenir une différence δ_{in} de la forme $M(\alpha',0,0,0)|(*,*,*,*)$, où $\alpha' \in \mathbb{F}_2^4$ est la différence obtenue en sortie de la fonction S du deuxième tour et $*$ peut être n'importe quelle valeur. Nous avons donc que $|\delta_{in}| = 4 + 16 = 20$.

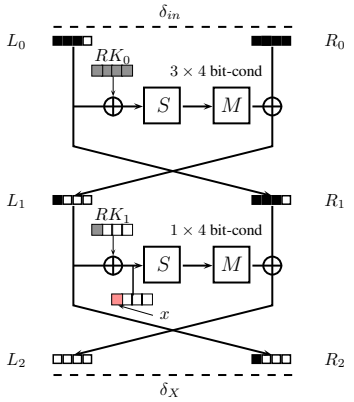


FIGURE 6.3. : Chemin différentiel sur 2 tours d'un schéma de Feistel.

Nous partons maintenant d'une différence de δ_{in} et nous voulons que les *différences* sur les trois quartets les plus à droite de la sortie de la fonction interne du premier tour soient les mêmes que pour les trois quartets les plus à droite des clairs. Autrement dit, nous voulons annuler les différences sur trois quartets de manière à obtenir une différence à l'entrée du deuxième tour du type : $(\alpha,0,0,0)|M(\alpha',0,0,0)$. Cela fait 3×4 bits de conditions, ou bien dit autrement, qu'une paire a une probabilité $1/2^{12}$ de vérifier la différence à la sortie du premier tour. Au deuxième tour, par construction nous savons que le quartet de différence à la sortie de la première boîte-S, S_0, se *diffuse* sur les trois quartets de gauche après la fonction linéaire M, mais cette différence ne dépend que de 4 bits. Donc une paire de données qui a passé les conditions du premier tour, a une probabilité $1/2^4$ d'avoir une différence δ_X à la sortie du deuxième tour. La probabilité de la différentielle tronquée $(\delta_{in} \to \delta_X)$ est donc

$$\frac{1}{2^{12}} \times \frac{1}{2^4} = \frac{1}{2^{16}}.$$

Dans cet exemple, et puisque la probabilité de la différentielle $(\delta_X \to \delta_{in})$ est 1, l'équation de Bayes (équation (5.3)) nous donne directement la probabilité de la différentielle $(\delta_{in} \to \delta_X)$:

$$\frac{2^{|\delta_X|}}{2^{|\delta_{in}|}} = \frac{2^4}{2^{20}} = \frac{1}{2^{16}}.$$

Une fois le nombre de bits de conditions estimé, il nous faut maintenant évaluer l'ensemble k_{in}, ce qui est un petit peu plus délicat. En effet, cette partie dépend grandement de l'algorithme de cadencement de clés. La procédure à suivre est la suivante :

- collecter tous les bits des sous-clés qu'il nous faut connaître afin de vérifier le chemin différentiel $(\delta_{in} \to \delta_X)$ (voir exemple 6.1.5) ;

- vérifier les relations qu'il peut exister entre ces bits de sous-clés à l'aide de l'algorithme de cadencement de clés.

Le paramètre qui nous intéressera lors de l'évaluation de la complexité est surtout la taille, en bits, de l'ensemble k_{in}. Plus encore, la taille de l'ensemble $k_{in} \cup k_{out}$, que nous notons $|k_{in} \cup k_{out}|$ et qui correspond au nombre minimal de bits de sous-clés que nous devons connaître afin de vérifier les deux chemins différentiels ($(\delta_{in} \to \delta_X)$ et $(\delta_{out} \to \delta_Y)$). Du point de vue de la *théorie de l'information*, la valeur $|k_{in} \cup k_{out}|$ correspond au logarithme de l'entropie des bits de sous-clés intervenant dans les deux chemins différentiels. Nous les appellerons parfois les *bits d'information de la clé*.

Exemple 6.1.5. Dans l'exemple précédent, nous avons vu comment calculer le nombre de bits de conditions, c'est-à-dire la probabilité d'une différentielle, sur notre chiffrement *exemple*. Nous allons maintenant étudier quels bits de clés il nous faut connaître pour pouvoir savoir si le chemin différentiel se vérifie ou non.

Comme nous pouvons le voir à la figure 6.3, nous avons besoin de connaître les trois premiers quartets de RK_0 pour pouvoir contrôler la différence à la sortie de la fonction S et ainsi vérifier les conditions du premier tour. Pour pouvoir maîtriser la différence à la sortie de S au deuxième tour, nous avons besoin de connaître la *valeur* du premier quartet en entrée, noté x sur la figure 6.3, qui est égale à

$$x = RK_1\rfloor_0 \oplus M \circ S(RK_0 \oplus L_0)\rfloor_0 \oplus R_0\rfloor_0.$$

Pour pouvoir connaître la valeur x, nous devons donc avoir la valeur du premier quartet de la sous-clé RK_1, noté $RK_1\rfloor_0$, et de l'intégralité de la sous-clé RK_0. Puisque dans cet exemple nous supposons les sous-clés indépendantes, nous avons donc que $|k_{in}| = 20$ bits.

6.2. Analyse de complexité théorique

Dans cette section, nous montrons comment formaliser l'évaluation des complexités d'une cryptanalyse différentielle. Nous proposons, pour la première fois, une méthode unifiée pour réaliser cette analyse.

Supposons que nous attaquons une fonction de chiffrement par blocs E_K, dont la taille des blocs est s bits, et qui est paramétrée par une clé K de taille $|K|$ bits. Supposons aussi que les paramètres de l'attaque soient déjà déterminés. Autrement dit, nous avons une différentielle impossible $(\delta_X \nrightarrow_{r_\delta} \delta_Y)$ placée entre les tours r_{in} et $r_{in} + r_\delta - 1$, un chemin différentiel $(\delta_{in} \to \delta_X)$ ayant une probabilité $2^{-c_{in}}$ de se vérifier, un chemin différentiel $(\delta_{out} \to \delta_Y)$ ayant une probabilité $2^{-c_{out}}$ de se vérifier, et un total de $|k_{in} \cup k_{out}|$ bits de sous-clés qui interviennent dans les chemins différentiels.

Nous déterminons maintenant la quantité de données nécessaires (*i.e.* le nombre de blocs de messages) qu'il nous faut construire pour appliquer notre attaque avec succès. Nous déterminons aussi les complexités en temps et en mémoire d'une cryptanalyse différentielle impossible.

6.2.1. Complexité en données

Puisque nous considérons des fonctions de chiffrement par blocs itératif, qui sont des permutations paramétrées par des clés, nous considérons que nous pouvons raisonnablement prendre des données en chiffrement ou en déchiffrement, de manière indifférente.

La probabilité que pour une clé donnée, une paire de blocs satisfaisant les différences tronquées δ_{in} en entrée et δ_{out} en sortie du chiffrement E_k, vérifie les deux chemins différentiels est $2^{-(c_{in}+c_{out})}$. En d'autres termes, c'est la probabilité, pour une paire de blocs vérifiant la différentielle tronquée $(\delta_{in} \to \delta_{out})$, d'éliminer une clé de l'ensemble des clés possibles. La probabilité qu'une clé reste dans cet ensemble est donc $1 - 2^{-(c_{in}+c_{out})}$, pour une paire de données. Si nous répétons la procédure pour un nombre N de paires de données vérifiant la différentielle sur les entrées/sorties du chiffrement E_K, la probabilité qu'une mauvaise clé reste parmi les clés candidates (*i.e.* une clé qui pourrait être la clé de chiffrement K) est donc $\mathcal{P} = (1 - 2^{-(c_{in}+c_{out})})^N$, que l'on approche généralement par $\mathcal{P} \simeq e^{-N \times 2^{-(c_{in}+c_{out})}}$.

La stratégie pour choisir le nombre de paires N n'est pas unique. Ce choix dépend principalement de la complexité en temps, qui est influencée par N, et de la quantité de données nécessaires pour trouver ces N paires vérifiant la différentielle voulue $(\delta_{in} \to \delta_{out})$. La méthode la plus répandue est cependant de choisir N suffisamment grand tel que seule la bonne clé, la clé K reste après la phase de filtrage. Cela revient à choisir N telle que la probabilité $\mathcal{P}$ vérifie

$$\mathcal{P} = (1 - 2^{-(c_{in}+c_{out})})^N \simeq e^{-N \times 2^{-(c_{in}+c_{out})}} < \frac{1}{2^{|k_{in} \cup k_{out}|}},$$

puisque la quantité de clés à tester est $2^{|k_{in} \cup k_{out}|}$.

Ici, nous souhaitons adopter une approche légèrement différente, qui nous permet de réduire le nombre de paires N nécessaire pour l'attaque, et donc de réduire le nombre de blocs à générer pour les construire. Cela nous permet plus de compromis temps/données, et des meilleurs, comme nous allons le voir. En procédant ainsi, nous allons potentiellement nous retrouver avec plusieurs clés candidates à la fin de l'attaque. Plusieurs clés qu'il faudra ensuite tester mais, puisque nous aurons besoin de chiffrer moins de données, la complexité en temps sera dans la plupart des cas plus basse. En pratique, nous allons commencer par évaluer nos compromis temps/données/mémoire avec une valeur de N telle que la probabilité $\mathcal{P}$ soit légèrement inférieure à $1/2$, c'est-à-dire que nous réduisons la recherche de la clé d'au moins un bit par rapport à la recherche exhaustive. La plus petite valeur de N vérifiant

$$\mathcal{P} = (1 - 2^{-(c_{in}+c_{out})})^N \simeq e^{-N \times 2^{-(c_{in}+c_{out})}} < \frac{1}{2},$$

est approximativement $N_{\min} = 2^{c_{in}+c_{out}}$. Suivant le nombre de données, que nous notons $C_{N_{\min}}$, dont nous allons avoir besoin pour générer ces $N_{\min}$ paires vérifiant la différentielle en entrée/sortie de la fonction de chiffrement, nous pourrons ensuite éventuellement construire plus de paires (*i.e.* augmenter la valeur N).

Nous fournissons maintenant la première solution générique, à notre connaissance, au problème suivant.

Problème 6.2.1. Comment déterminer le nombre de données nécessaire C_N sur s bits, pour obtenir N paires vérifiant une différentielle tronquée $(\delta_{in}, \delta_{out}) \in (\mathbb{F}_2^s)^2$?

En 2010, Henri Gilbert et Thomas Peyrin [99] donnent une solution au problème appelé *limited-birthday*, qui est le cas le plus simple, $N = 1$, du problème 6.2.1. Suivant leur solution, le coût en données pour trouver une paire vérifiant une certaine différentielle $(\delta_{in} \to \delta_{out})$ (voir figure 6.4) est donné par

$$C_1 = \max \left\{ \min_{\delta \in \{\delta_{in}, \delta_{out}\}} \left\{ \sqrt{2^{s+1-|\delta|}} \right\}, 2^{s+1-(|\delta_{in}|+|\delta_{out}|)} \right\}.$$

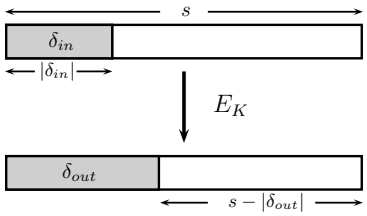


FIGURE 6.4. : Une paire de données vérifiant la différentielle tronquée ($\delta_{in} \to \delta_{out}$) sur la fonction de chiffrement E_K.

Une façon directe de trouver une solution C_N, $N > 1$, au problème 6.2.1 pourrait être de l'estimer par $C_N = N \times C_1$. Cependant, cette solution n'est pas toujours optimale, comme nous allons le voir. Rappelons que puisque notre fonction de chiffrement est une fonction bijective de $\mathbb{F}_2^s$, les rôles joués par δ_{in} et δ_{out} sont complètement interchangeables.

Nous pouvons distinguer deux cas, dépendant de la dimension des différentielles tronquées δ_{in} et δ_{out}, c'est-à-dire $|\delta_{in}|$ et $|\delta_{out}|$, ainsi que du nombre N de paires que nous voulons construire. Définissons tout d'abord ce qu'est une *structure*.

Définition 6.2.2 (Structure)**.** Une *structure* relativement à une différence tronquée δ_{in} est l'ensemble des données prenant toutes les valeurs possibles pour la différence δ_{in} et tel que les $s - |\delta_{in}|$ bits restants sont fixés à une valeur constante. Une structure contient $2^{|\delta_{in}|}$ données.

La quantité $2^{|\delta_{in}|}2^{|\delta_{in}|-1}$ correspond au nombre de paires que nous pouvons générer à l'aide d'une structure sur δ_{in}. D'un autre côté, la valeur $s-|\delta_{out}|$ correspond à la dimension de la collision partielle que nous souhaitons sur les chiffrés, puisque nous voulons que la différence en sortie de la fonction de chiffrement E_K appartienne à l'ensemble δ_{out} (voir figure 6.4). Nous voulons donc comparer la valeur

$$\frac{2^{|\delta_{in}|}2^{|\delta_{in}|-1}}{2^{s-|\delta_{out}|}},$$

qui correspond au nombre de paires ayant la différentielle tronquée ($\delta_{in} \to \delta_{out}$) que nous pouvons générer à l'aide d'une structure sur δ_{in}, au nombre N de paires voulues (voir problème 6.2.1).

- Si $N \leq \dfrac{2^{|\delta_{in}|}2^{|\delta_{in}|-1}}{2^{s-|\delta_{out}|}}$, cela signifie que $C_N \leq 2^{|\delta_{in}|}$. Plus précisément, une seule structure sur δ_{in} est suffisante pour construire N paires ayant la bonne différentielle tronquée. Nous avons donc
$$N = \frac{C_N C_N/2}{2^{s-|\delta_{out}|}},$$
qui signifie qu'il nous faut $C_N = \sqrt{N2^{s-\delta_{out}+1}}$ données.
- Sinon, si $N > \dfrac{2^{|\delta_{in}|}2^{|\delta_{in}|-1}}{2^{s-|\delta_{out}|}}$, nous devons alors considérer plusieurs structures. Soit 2^y le nombre de structures dont nous ayons besoin tel que $N = 2^y \dfrac{2^{|\delta_{in}|}2^{|\delta_{in}|-1}}{2^{s-|\delta_{out}|}}$. Le nombre de données nécessaires est dans ce cas donnée par
$$C_N = 2^y 2^{|\delta_{in}|} = N2^{s-|\delta_{out}|-|\delta_{in}|+1}.$$

En conclusion, le nombre de données, à chiffrer ou déchiffrer, qu'il faut à un attaquant pour obtenir N paires avec la différentielle tronquée $(\delta_{in} \to \delta_{out})$ est

$$C_N = \max\left\{\min_{\delta\in\{\delta_{in},\delta_{out}\}}\left\{\sqrt{N2^{s+1-|\delta|}}\right\}, N2^{s+1-|\delta_{in}|-|\delta_{out}|}\right\}. \tag{6.1}$$

Nous remarquons que dans le premier cas ci-dessus, nous gagnons un facteur $\sqrt{N}$ par rapport à la solution naïve qui aurait été de prendre $C_N = N \times C_1$.

Il est évident que pour que l'attaque soit valide, il ne faut pas que l'attaque requiert plus de données pour construire N paires que le nombre total de blocs possibles. En d'autres termes, l'inégalité suivante doit être vérifiée

$$C_N \leq 2^s.$$

Dans certains travaux (par exemple dans [219] et [6]), cette inégalité n'est pas respectée, invalidant de fait les attaques proposées.

6.2.2. Complexités en temps et en mémoire

Maintenant que nous avons tous les paramètres nous permettant de monter une cryptanalyse différentielle impossible, nous allons voir comment évaluer la complexité en temps, c'est-à-dire le nombre de chiffrements à effectuer avec E_K pour pouvoir retrouver la clé K, ainsi que la complexité en mémoire. Nous fournissons donc des formules génériques permettant d'évaluer ces complexités.

En 2008, Lu et al. [149] introduisent une technique appelée *early-abort* ("abandon précoce" en français) pour améliorer les cryptanalyses différentielles impossibles. Suivant cette technique, l'attaque consiste en premier lieu à stocker les N paires de données. Ensuite, nous testons les parties des sous-clés *morceau par morceau*, en réduisant au fur et à mesure de façon conséquente le nombre de paires qui pourraient mener à la différentielle impossible. Le but est, pour réaliser cela, de ne pas avoir une complexité $2^{|k_{in}\cup k_{out}|}N$, qui correspond à la complexité naïve de tester pour chaque clé, toutes les paires. Au lieu de cela, la technique *early-abort* permet de réduire, dès que nous en avons la possibilité, la liste des paires possibles avec les conditions sur la clé testée. Une fois qu'une clé a été testée complètement (tous les morceaux ont été testés), le nombre approximatif de paires qu'il nous reste à vérifier pour voir si la différentielle impossible $(\delta_X \nrightarrow_{r_\delta} \delta_Y)$ est impliquée au moins une fois est $N \times 2^{-(c_{in}+c_{out})}$. Si cela arrive, nous pouvons considérer que la clé qui a été testée petit à petit ne peut pas être la clé de chiffrement. En effet, il s'agit d'une clé telle qu'il existe des données nous amenant à une différentielle de probabilité zéro, ce n'est donc pas la clé qui a servi au chiffrement. À la fin, nous aurons testé chacune des $2^{|k_{in}\cup k_{out}|}$ clés à vérifier.

La complexité en temps est déterminée par trois quantités. La première est la quantité de données C_N, qu'il faut chiffrer (ou déchiffrer) pour obtenir les N paires avec la bonne différentielle sur la fonction de chiffrement E_K où N est tel que $\mathcal{P} \simeq e^{-N\times 2^{-(c_{in}+c_{out})}} < 1/2$. La seconde quantité correspond au nombre de clés à tester, $2^{|k_{in}\cup k_{out}|}$, multiplié par le coût de vérifier les paires avec la technique *early-abort*. Ce coût peut être approximé par

$$\frac{N}{2^{c_{in}+c_{out}}}C'_E,$$

où C'_E est le coût relatif d'un chiffrement partiel, c'est-à-dire le coût d'évaluer une paire sur les deux chemins différentiels (δ_{in}, δ_X) et (δ_{out}, δ_Y)[1]. Précisons que C'_E ne peut être calculée avec précision que lorsque les paramètres de l'attaque sont connus. Néanmoins, C'_E est généralement estimé en effectuant le ratio entre le nombre d'opérations non-linéaires[2] actives dans les deux chemins différentiels, et le nombre total d'opérations non-linéaires de la fonction de chiffrement E_K. À ce coût s'ajoute celui de chiffrer partiellement au moins une fois chaque paire : $N \times C'_E$.

Finalement, la dernière quantité à évaluer est le coût de tester les clés restantes dans l'ensemble des clés possibles. Plus précisément, il reste $|K| - |k_{in} \cup k_{out}|$ bits de clés qui n'ont pas été testés durant le filtrage, et il reste $\mathcal{P}2^{|k_{in} \cup k_{out}|}$ clés dans l'ensemble des clés candidates. Ce coût est donc $2^{|K|-|k_{in} \cup k_{out}|}\mathcal{P}2^{|k_{in} \cup k_{out}|} = 2^{|K|}\mathcal{P}$.

En prenant en compte le coût d'un chiffrement avec la fonction E_K, que nous notons C_E, nous en concluons que la complexité en temps d'une cryptanalyse différentielle impossible est

$$T_{comp} = \left(C_N + \left(2^{|k_{in} \cup k_{out}|} \frac{N}{2^{c_{in}+c_{out}}} + N \right) C'_E + 2^{|K|}\mathcal{P} \right) C_E, \tag{6.2}$$

où, rappelons-le, $C_N = \max \left\{ \min_{\delta \in \{\delta_{in}, \delta_{out}\}} \left\{ \sqrt{N 2^{s+1-|\delta|}} \right\}, N 2^{s+1-|\delta_{in}|-|\delta_{out}|} \right\}$ avec N tel que $\mathcal{P} \simeq e^{-N \times 2^{-(c_{in}+c_{out})}} < 1/2$. Il est facile de voir que l'utilisation de ces formules nous permet d'obtenir des meilleurs compromis temps/données, et des attaques plus optimisées, que si nous nous limitions à ne garder qu'une clé dans l'ensemble des clés candidates. En effet dans le cas où $\mathcal{P}2^{|K|} < 2^{-|k_{in} \cup k_{out}|}$, le dernier terme de la formule (6.2) est plus réduit, mais les deux autres termes sont maximisés.

Pour qu'une telle attaque soit valide, il faut qu'elle ait une meilleure complexité que celle de la recherche exhaustive de la clé, c'est-à-dire que T_{comp} doit vérifier

$$T_{comp} < 2^{|K|} C_E.$$

Puisque nous n'avons à stocker que les N paires durant l'attaque, cette valeur correspond donc aussi à la complexité en mémoire. Néanmoins, nous pouvons avoir $N > 2^{|k_{in} \cup k_{out}|}$, même s'il est très rare que cela se produise. Dans ce cas là, nous stockons les clés éliminées lors du filtrage plutôt que de stocker les N paires.

6.2.3. Bien choisir ses N paires

Nous expliquons maintenant deux manières de choisir les différences δ_{in}, δ_{out}, et de calculer les nombres de conditions c_{in} et c_{out} qui sont associés aux différentielles $(\delta_{in} \to \delta_X)$ et $(\delta_{out} \to \delta_Y)$ respectivement. Pour cela, nous utilisons une nouvelle fois le chiffrement *exemple* de la section 6.1.1. Nous ne traiterons que le cas pour δ_{in} et c_{in}, mais l'approche est identique pour δ_{out} et c_{out} sur des données en déchiffrement. Nous ne prenons qu'un chemin différentiel sur un tour, comme décrit à la figure 6.5.

1. La manière la plus intuitive de faire, est de considérer que $|\delta_{in}| = 4 + 4$ et $c_{in} = 4$. Dans ce cas, pour une certaine clé, une paire parmi les $2^{4+4}2^{4+4-1}$ ayant une différence δ_{in}, aura une différence δ_X après un tour avec une probabilité 2^{-4}.

[1] le coût du test du premier morceau de sous-clé, en utilisant la technique *early-abort*, peut être réduit en utilisant la méthode de la sous-section 6.2.3

[2] Il s'agit en général de boîtes-S. Cependant pour SIMON, il s'agit du nombre de ET (logique) bit-à-bit.

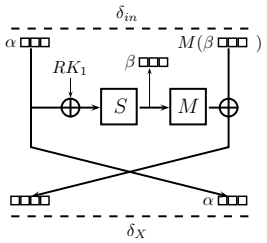


FIGURE 6.5. : Choisir δ_{in} et c_{in}

2. La différence notée par α dans la figure 6.5, peut prendre $2^4 - 1$ valeurs. Cependant, chacune de ces valeurs peut être associée par la table de distribution des différences de la boîte-S, S_0, à 2^3 valeurs en moyenne. C'est à dire que pour chaque valeur de différence α, la différence β ne peut prendre qu'au plus 2^3 valeurs différentes. Nous pouvons donc considérer que $|\delta_{in}| \simeq 4 + 3$. Dans ce cas, $c_{in} = 3$ puisque pour chaque paire d'entrée, appartenant aux $2^{4+3}2^{4+3-1}$ possibles, il existe en moyenne deux valeurs rendant la transition différentielle de S_0 possible (au lieu de 1 dans le cas précédent).

En nous replaçant dans le contexte de la section précédente, que ces deux cas mènent à la même évaluation de complexité en temps. En effet, les variations sur le nombre de paires N, se compensent avec les variations sur les nombres de conditions à vérifier $c_{in} + c_{out}$. Néanmoins, la complexité en mémoire (donnée par N) est sensiblement meilleure dans le deuxième cas. De plus, le deuxième cas, dans lequel un filtrage préliminaire est effectué, permet de réduire le coût moyen lorsque l'on utilise la technique *early-abort* [149].

Dans de nombreux travaux, par exemple [211] et [149], les auteurs utilisent le deuxième cas, mais souvent de manière partielle (soit sur les entrées soit sur les sorties) sans raison apparente. Nous remarquons de plus que dans ces articles, le nombre de conditions associées c_{out} n'est pas toujours calculé correctement (8 bits au lieu de 7). Pour des raisons de simplicités, nous utiliserons le premier cas dans nos applications, et nous déterminerons après coup la mémoire nécessaire à nos attaques.

6.3. Outil Automatique

Dans les sections précédentes, nous avons vu quels étaient les paramètres importants à connaître (et comment les calculer) pour monter une cryptanalyse différentielle impossible ainsi que l'évaluation des complexités en temps, données et mémoire.

Très souvent, la partie la plus technique et la plus chronophage de la cryptanalyse différentielle impossible est celle qui consiste à étendre la différentielle impossible par des chemins différentiels. En effet, la détermination des paramètres est un point délicat qui demande beaucoup d'attention et qui est souvent source d'erreurs. Tous les cas recensés dans le tableau 6.1 ont des défauts dans cette partie. D'autre part, il est délicat de tester à la main toutes les possibilités pour les chemins différentiels et de vérifier lesquels ont les meilleures complexités finales. Par exemple, pour une certaine différentielle impossible sur r_δ tours, il n'est pas facile de savoir, *a priori*, si ajouter une différentielle sur $r_{in} = 2$ tours en entrée et $r_{out} = 2$ tours en sortie donnera de meilleures complexités par rapport à ajouter $r_{in} = 0$ tours en entrée et $r_{out} = 4$ tours en sortie.

Nous décrivons ici un algorithme, que nous avons conçu et implanté, permettant d'automatiser la recherche des meilleures cryptanalyses différentielles impossibles, celles qui atteignent le plus grand nombre de tours, avec plusieurs compromis temps/données. Sans l'analyse théorique que nous avons fournie aux sections précédentes, il n'aurait pas été possible de concevoir et d'implanter des programmes simplifiant le travail du cryptanalyste.

De manière plus précise, nous avons conçu des programmes nous permettant de tester exhaustivement tous les chemins différentiels à ajouter en entrée et en sortie des schémas de Feistel pour la cryptanalyse différentielle impossible.

La première étape à effectuer, est de collecter toutes les différentielles impossibles $\delta_X \nrightarrow_{r_\delta} \delta_Y$ telles que r_δ est maximal. Cette étape est raisonnablement réalisable si l'on considère les travaux déjà effectués [123], et le fait que, pour les schémas de Feistel, des différences δ_X, δ_Y de petits poids engendrent en général un plus grand nombre de tours r_δ. Elle peut être de toute façon optimisée indépendemment de notre algorithme.

La deuxième étape consiste à "créer" les chemins différentiels $(\delta_{in} \to \delta_X)$ et $(\delta_{out} \to \delta_Y)$, tour par tour. Pour cela, il suffit de vérifier à chaque étape que $C_N < 2^s$ et $T_{comp} < 2^{|K|}$.

Notons toutefois, que dans la plupart des cas (sauf pour Camellia dans nos exemples, à cause des fonctions FL/FL^{-1}), une différentielle impossible sur r_δ tours est indépendante des tours où elle est placée. Nous pouvons alors considérer tous les chemins différentiels $(\delta_{in} \to \delta_X)$ et $(\delta_{out} \to \delta_Y)$ que nous souhaitons.

Enfin, pour chaque ensemble de paramètres, nous pouvons faire varier la quantité de données $C_{N_{\min}} \leq C_N \leq 2^s$ et ainsi obtenir tous les compromis temps/données.

Pour chaque différentielle impossible, la procédure générale pour automatiser la recherche des paramètres est décrite par l'algorithme 5. Cet algorithme retourne une liste d'attaques valides qu'il est possible de trier suivant les critères désirés (nombre de tours, complexité en temps, ...).

L'algorithme procède en ajoutant au fur et à mesure des tours aux chemins différentiels, c'est-à-dire en augmentant les valeurs r_{in} et r_{out}. La boucle de la ligne 1 s'arrête lorsque toutes les combinaisons (r_{in}, r_{out}) menant à des paramètres vérifiant $C_N < 2^s$ ont été testées. Les opérations des lignes 2 et 3 visent à déterminer les paramètres de chaque chemin différentiel suivant la méthode que nous avons donnée à la sous-section 6.1.3. La détermination des bits d'information de la clé à la ligne 4, quant à elle, dépend de ce que l'utilisateur donne comme consigne, c'est-à-dire s'il spécifie une recherche de relation entre les bits des sous-clés dans l'algorithme de cadencement de clé de la fonction de chiffrement analysée.

La boucle de la ligne 7 permet d'obtenir tous les compromis de complexité en temps, en données, et en mémoire de notre cryptanalyse différentielle impossible. Il est ainsi aisé de choisir quelle complexité nous voulons minimiser. On remarque bien ici que l'évaluation de la complexité en temps d'une cryptanalyse différentielle impossible ne peut varier qu'en fonction du nombre de paires de données, N, lorsque les chemins différentiels et leurs paramètres ont été fixés.

En plus de l'établissement de cette méthode générale valable pour n'importe quel type de chiffrement par blocs, nous avons conçu un programme[3] fonctionnant dans le cas des chiffrements de Feistel généralisés. Nous avons donc implanté l'algorithme 5 en langage C. Pour des raisons pratiques nous avons aussi inclus la recherche de différentielles impossibles et nous autorisons l'utilisateur à spécifier le nombre de tours qu'il souhaite

[3] me contacter par mail pour recevoir une version.

Algorithme 5 Procédure générale pour déterminer les paramètres d'une cryptanalyse différentielle impossible.

Entrée :
Une différentielle impossible $(\delta_X \nrightarrow_{r_\delta} \delta_Y)$ d'une fonction de chiffrement E_K.
Sortie :
Un ensemble $\mathcal{L}_{sol}$ formé des paramètres :
$\{((\delta_{in} \to \delta_X), r_{in}, c_{in}), ((\delta_{out} \to \delta_Y), r_{out}, c_{out}), |k_{in} \cup k_{out}|, N, C_N, T\}$,
correspondants à une cryptanalyse différentielle impossible valide de E_K.

Fonction $\textsc{Tool}_{E_K}(\delta_X \nrightarrow_{r_\delta} \delta_Y)$
Pour tous les couples (r_{in}, r_{out})
  Déterminer $((\delta_{in} \to \delta_X), r_{in}, c_{in})$; ▷ sous-section 6.1.3
  Déterminer $((\delta_{out} \to \delta_Y), r_{out}, c_{out})$;
  Déterminer $|k_{in} \cup k_{out}|$ et C'_E ;
  $N \leftarrow N_{\min} = 2^{c_{in}+c_{out}}$;
  $C_N \leftarrow C_{N_{\min}} = \max\left\{\min_{\delta\in\{\delta_{in},\delta_{out}\}}\left\{\sqrt{N_{\min}2^{s+1-|\delta|}}\right\}, N_{\min}2^{s+1-|\delta_{in}|-|\delta_{out}|}\right\}$;
  Tant que $C_N < 2^s$
    $T \leftarrow C_N + \left(2^{|k_{in}\cup k_{out}|}\frac{N}{2^{c_{in}+c_{out}}} + N\right)C'_E + 2^{|K|}e^{-N\times 2^{-c_{in}-c_{out}}}$;
    Si $T < 2^{|K|}$
      Ajouter les paramètres à $\mathcal{L}_{sol}$;
    Augmenter N ;
    Recalculer C_N ;
  Retourner $\mathcal{L}_{sol}$.

attaquer. Cet outil automatique est un moyen efficace de tester toutes les combinaisons possibles, toutes les cryptanalyses différentielles impossibles contre un chiffrement par blocs de type Feistel.

L'utilisateur de notre outil automatique doit premièrement donner les spécifications du chiffrement par blocs qu'il souhaite cryptanalyser, ainsi que la méthode pour déterminer les bits d'information de la clé. Ce dernier point est une tâche difficile qui est entièrement dépendante de l'algorithme de cadencement de clés.

Le programme s'exécute ensuite en fournissant toutes les informations nécessaires pour réaliser une cryptanalyse différentielle impossible contre cette fonction de chiffrement. L'utilisateur a ensuite le choix de choisir un jeu de paramètres particulier pour obtenir plus de détails concernant l'attaque associée, incluant notamment la description complète des chemins différentiels ainsi que la table des compromis de complexité.

Sans cet outil automatique, trouver les *meilleures* cryptanalyses différentielles contre les chiffrements par blocs de la famille Simon (section 6.8) ainsi que pour réaliser la cryptanalyse différentielle impossible sur 23 tours de LBlock (section 6.7) nous auraient pris un temps considérablement plus long.

6.4. Techniques avancées pour améliorer la complexité

Dans cette section, nous présentons des techniques qui permettent d'améliorer sensiblement les complexités d'une cryptanalyse différentielle impossible. Nous avons déjà introduit la technique *early-abort*, qui nous permet de ne pas tester toutes les paires

pour toutes les clés sur les chemins différentiels. Nous introduisons ici deux nouvelles techniques originales : les *différentielles impossibles multiples* et la technique *state-test*. La première reprend un concept existant, et nous nous en servons pour réduire la complexité en données d'une attaque. La *state-test* technique est, quant à elle inédite, et permet de réduire significativement la complexité en temps d'une cryptanalyse différentielle impossible.

6.4.1. Les différentielles impossibles multiples

L'idée d'utiliser des différentielles impossibles multiples a été initiée par Tsunoo et al. [197] en 2008, et appliquée pour cryptanalyser douze tours du chiffrement par blocs CLEFIA-128. Dans cette sous-section, nous décrivons comment utiliser les différentielles impossibles multiples pour réduire la complexité en données, c'est-à-dire réduire le nombre de données, C_N, nécessaire pour construire les N paires. L'idée est de considérer simultanément plusieurs différentielles impossibles plutôt que de n'en considérer qu'une seule. Pour des raisons de simplicité, et sans perte de généralité, nous considérons qu'il existe *a priori* deux façons de procéder :

1. Prendre n_{in} différentielles impossibles $\delta_{X_i} \nrightarrow_{r_\delta} \delta_{Y_i}$, $1 \leq i \leq n_{in}$, telles que
$$i \neq j \qquad \Rightarrow \qquad \begin{cases} \delta_{X_i} \neq \delta_{X_j} & \text{et} \\ \delta_{Y_i} \neq \delta_{Y_j}. \end{cases}$$
2. Prendre n_{out} différentielles impossibles ayant la même entrée :
$$\delta_X \nrightarrow_{r_\delta} \begin{cases} \delta_{Y_1} \\ \vdots \\ \delta_{Y_{n_{out}}}. \end{cases}$$

Toujours pour des raisons de simplicité, mais aussi pour considérer le cas le plus défavorable pour l'attaquant, on considère que l'intersection entre les différences δ_{in} générées est nulle. Reprenons notre chiffrement *exemple* de la section 6.1.1 pour mieux comprendre ces deux cas.

Exemple 6.4.1. Soit $\alpha, \beta \in \mathbb{F}_2^4$ des différences non nulles. Nous avons par exemple les différentielles impossibles suivantes pour le chiffrement *exemple* de la section 6.1.1 :

$$\begin{aligned}
(0,0,0,0,\alpha,0,0,0) &\nrightarrow_5 (0,0,0,\beta,0,0,0,0) \\
(0,0,0,0,0,\alpha,0,0) &\nrightarrow_5 (0,0,\beta,0,0,0,0,0) \\
(0,0,0,0,0,0,\alpha,0) &\nrightarrow_5 (0,\beta,0,0,0,0,0,0) \\
(0,0,0,0,0,0,0,\alpha) &\nrightarrow_5 (\beta,0,0,0,0,0,0,0),
\end{aligned}$$

ainsi $n_{in} = 4$. Nous avons aussi les différentielles impossibles :

$$\begin{aligned}
(0,0,0,0,\alpha,0,0,0) &\nrightarrow_5 (0,0,0,\beta,0,0,0,0) \\
(0,0,0,0,\alpha,0,0,0) &\nrightarrow_5 (0,0,\beta,0,0,0,0,0) \\
(0,0,0,0,\alpha,0,0,0) &\nrightarrow_5 (0,\beta,0,0,0,0,0,0),
\end{aligned}$$

et dans ce cas là nous avons $n_{out} = 3$.

Il est important de noter que pour que notre analyse soit valide, dans les deux cas le nombre de conditions associées aux chemins différentiels (*i.e.* c_{in} et c_{out}) doit rester le même. Dans les deux cas, il résulte de l'ajout de différentielles impossibles une augmentation, virtuelle, du nombre de données disponibles. En effet, nous considérons plusieurs différentielles, nous pouvons donc définir deux quantités, $|\delta'_{in}|$ et $|\delta'_{out}|$ qui joueront les rôles de $|\delta_{in}|$ et $|\delta_{out}|$ respectivement, telles que

$$|\delta'_{in}| = |\delta_{in}| + \log_2(n_{in}) \qquad \text{et} \qquad |\delta'_{out}| = |\delta_{out}| + \log_2(n_{out}).$$

Ainsi, $|\delta'_{in}|$ (resp. $|\delta'_{out}|$) est le logarithme (en base 2) de la taille totale de l'ensemble des différences en entrée (resp. sortie) possibles.

La complexité en données, C_N est maintenant calculée de la même façon qu'avant, mais avec les valeurs corrigées pour les dimensions des structures. Nous voyons que cette nouvelle valeur de C_N est inférieure à la précédente (voir formule (6.1)).

À part la quantité C_N, le reste de l'évaluation de la complexité en temps reste la même. En effet, le terme de la somme dans la formule (6.2) correspondant au coût de tester les clés reste identique puisque, pour une paire donnée, le nombre de bits de clé considérés va être le maximum des bits d'information de la clé de tous les chemins impossibles, que nous notons k' . De la même manière, le dernier terme de la formule 6.2 est maintenant

$$\frac{2^{|K|}}{2^{k'}}\left(\mathcal{P}2^{k'}\right) = 2^{|K|},$$

et reste donc le même.

Cette technique se trouve être très utile pour monter des cryptanalyses différentielles impossibles lorsque la conception du chiffrement par blocs ne nous permet pas d'avoir suffisamment de données. Nous en verrons une application à la section 6.8 où, pour certaines versions du chiffrement Simon, les attaques n'auraient pas été valides sans l'utilisation de cette technique. À la section 6.5, nous montrons, sur le chiffrement CLEFIA, que l'utilisation de cette technique permet d'améliorer la complexité en données d'une attaque valide.

6.4.2. La technique state-test

Nous proposons maintenant une nouvelle méthode qui consiste à vérifier les *valeurs* possibles sur une partie de l'état, plutôt que d'utiliser les bits de sous-clés nécessaires pour les calculer. Cela peut rappeler en un sens, la technique utilisée par Orr Dunkelman, Gautham Sekar et Bart Preneel [92], dans le contexte des attaques du type *rencontre-au-milieu* ("meet-in-the-middle" en anglais), où ils testent quelques bits de l'état interne. Cependant, la technique que nous appelons *state-test* et que nous présentons dans cette sous-section est différente puisque nous n'avons pas besoin de tester des bits de l'état interne, mais juste de vérifier qu'ils prennent certaines valeurs.

Il arrive souvent durant la phase de filtrage d'une cryptanalyse différentielle impossible, que la taille de l'état qui a besoin d'être connu soit très inférieure au nombre de bits de sous-clé dont il dépend. En effet, la diffusion du chiffrement faisant son office, plus on avance dans les tours, plus un certain bit dépend des bits des tours précédents. Comme nous allons le voir, sous certaines conditions, nous pourrons nous passer de calculer des parties de l'état et quand même arriver à éliminer des clés candidates. Le principe de la technique *state-test* est de *fixer* quelques bits des données pour avoir ce même nombre de bits de clés en moins à tester.

Nous allons expliquer comment cette méthode fonctionne en illustrant chaque étape d'un exemple, à l'aide du chiffrement *exemple* de la section 6.1.1. Rappelons que ce chiffrement est un schéma de Feistel sur 32 bits, où chaque branche est la concaténation de quatre mots de 4 bits (quartets).

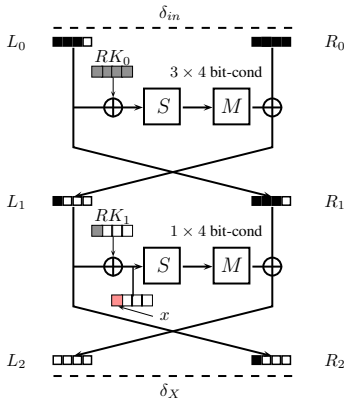


FIGURE 6.6. : Chemin différentiel ($\delta_{in} \to \delta_X$). Le quartet x est la partie de l'état sur laquelle nous allons appliquer la technique *state-test*.

Les paramètres $|\delta_{in}| = 20$, $c_{in} = 16$ et $k_{in} = 20$ ont déjà été calculés aux exemples 6.1.4 et 6.1.5 respectivement. Par souci de simplicité et de clarté, nous ne rajoutons pas de chemin différentiel en sortie de notre chiffrement *exemple*. La complexité en temps d'une cryptanalyse différentielle impossible contre notre chiffrement *exemple* sur $5 + 2$ tours est donc donnée par la formule 6.2 :

$$\left(C_N + \left(2^{20}\frac{N}{2^{16}} + N\right) C'_E + 2^{|K|-20}\mathcal{P}2^{20}\right) C_E. \tag{6.3}$$

Voyons maintenant comment l'utilisation de la technique *state-test* permet d'améliorer la complexité en temps. Nous allons appliquer cette technique sur le quartet noté x dans la figure 6.6. Notons que mathématiquement, x peut être exprimé par

$$\begin{aligned} x &= RK_1\rfloor_0 \oplus M \circ S(RK_0 \oplus L_0)\rfloor_0 \oplus R_0\rfloor_0 \\ x \oplus R_0\rfloor_0 &= RK_1\rfloor_0 \oplus m_0 S_0(RK_0\rfloor_0 \oplus L_0\rfloor_0) \oplus m_1 S_1(RK_0\rfloor_1 \oplus L_0\rfloor_1) \\ &\quad \oplus m_2 S_2(RK_0\rfloor_2 \oplus L_0\rfloor_2) \oplus m_3 S_3(RK_0\rfloor_3 \oplus L_0\rfloor_3), \end{aligned} \tag{6.4}$$

où les $m_i \in \mathbb{F}_2^4$ sont les coefficients de la matrice M.

Supposons maintenant, que pour toutes les données, nous fixons les 4 derniers bits de la branche de gauche, *i.e.* $L_0\rfloor_3$, à une valeur constante. En d'autres termes, s'il nous fallait plusieurs structures pour construire les N paires, nous imposons que la valeur de $L_0\rfloor_3$ soit la même pour toutes les structures. Nous pouvons vérifier qu'il s'agit d'une supposition raisonnable puisqu'il nous reste encore suffisamment de données pour que notre attaque soit valide. En effet, $C_{N_{\min}} \simeq 2^{22}$, nous pouvons donc fixer jusqu'à $32 - 22 = 10$ bits sur les données.

La procédure de filtrage commence alors comme auparavant, en testant les trois premiers quartets de RK_0 pour vérifier les conditions du premier tour. Après avoir fixé ces 12 bits de sous-clé, et parce que la probabilité de la différentielle du premier tour est $1/2^{12}$, il nous reste approximativement $N \times 2^{-12}$ paires. Pour chacune de ces paires, nous

connaissons donc les différences en entrée et en sortie de la boîte-S du deuxième tour. Ainsi, en cherchant dans la table de distribution des différences (voir définition 5.2.1) de la boîte-S impliquée, ici S_0, nous obtenons en moyenne une valeur pour x qui vérifie les conditions du deuxième tour, pour chaque paire (la transition n'est parfois pas possible, mais lorsqu'elle l'est, nous obtenons plusieurs valeurs de x possibles). D'où, en remplaçant la somme de toutes les valeurs connues par $x' \in \mathbb{F}_2^4$,

$$x' = x \oplus R_0\rfloor_0 \oplus m_0 S_0(RK_0\rfloor_0 \oplus L_0\rfloor_0) \oplus m_1 S_1(RK_0\rfloor_1 \oplus L_0\rfloor_1) \oplus m_2 S_2(RK_0\rfloor_2 \oplus L_0\rfloor_2),$$

nous pouvons réécrire l'équation (6.4) par

$$x' = RK_1\rfloor_0 \oplus m_3 S_3(RK_0\rfloor_3 \oplus L_0\rfloor_3). \tag{6.5}$$

Pour chaque valeur de sous-clé $(RK_0\rfloor_0, RK_0\rfloor_1, RK_0\rfloor_2)$, nous listons les $N \times 2^{-12}$ valeurs de x'. La dernière étape, mais non la moindre, consiste à regarder si les 2^4 valeurs possibles de x' sont présentes dans cette liste. Nous notons que puisque $N \geq 2^{c_{in}} = 2^{16}$, la liste contient bien au moins 2^4 éléments.

Rappelons que nous avons fixé la valeur de $L_0\rfloor_3$ à une même valeur constante pour toutes les paires. D'où les seules inconnues dans l'équation (6.5) sont $RK_0\rfloor_3$ et $RK_1\rfloor_0$. Maintenant, si toutes les valeurs de x' sont dans la liste, et puisque la boîte-S, S_3, est bijective, pour n'importe quelle valeur de $RK_0\rfloor_3$ et n'importe quelle valeur de $RK_1\rfloor_0$, il existera *toujours* au moins une paire telle que les conditions du second tour se vérifient, amenant donc à la différentielle impossible.

En conclusion, si toutes les valeurs de x' sont dans la liste, nous pouvons enlever des clés candidates *l'ensemble des clés* composées de la valeur $(RK_0\rfloor_0, RK_0\rfloor_1, RK_0\rfloor_2)$ et de toutes les combinaisons possibles de $RK_0\rfloor_3$ et $RK_1\rfloor_0$, puisqu'elles amènent toutes, pour au moins une paire, à la différentielle impossible.

Si par contre, toutes les valeurs de x' ne sont pas présentes dans la liste pour une certaine valeur de sous-clé $(RK_0\rfloor_0, RK_0\rfloor_1, RK_0\rfloor_2)$, nous pouvons tester comme clé de chiffrement potentielle, les clés composées de cette valeur, $(RK_0\rfloor_0, RK_0\rfloor_1, RK_0\rfloor_2)$, ainsi que de toutes les combinaisons des bits de sous-clés vérifiant l'équation (6.5) pour les valeurs manquantes de x', puisqu'elles restent dans l'ensemble des clés candidates.

Le gain de cette technique est que nous diminuons le nombre de bits de sous-clé à tester, et ainsi nous diminuons la complexité en temps. Autrement dit, au lieu d'éliminer les clés de l'ensemble des clés candidates une par une, nous le faisons par paquets. Dans cet exemple, la variable x' peut être vue comme 4 bits de sous-clé plutôt que comme les 2×4 bits (ceux de $RK_0\rfloor_3$ et $RK_1\rfloor_0$) qu'il aurait fallu en temps normal.

Grâce aux $f = 4$ bits que nous avons fixés dans les structures, nous avons $f = 4$ bits de clés en moins à tester dans notre filtrage. La complexité en temps dans ce cas est donnée par

$$\left(C_N + \left(2^{20-f}\frac{N}{2^{16}} + N\right) C'_E + 2^{|K|-(20-f)}\mathcal{P}2^{20-f}\right) C_E. \tag{6.6}$$

En comparant les équations (6.3) et (6.6), nous voyons que la complexité en temps d'une cryptanalyse différentielle impossible est plus faible avec la technique *state-test* qu'avec l'approche classique. En effet, le premier et le dernier terme restent les mêmes entre les formules (6.3) et (6.6), tandis que le deuxième terme, qui est le terme contraignant pour les meilleurs compromis, est plus petit avec la technique *state-test*.

Finalement, nous remarquons que la probabilité $\mathcal{P}$ qu'une clé reste dans l'ensemble des clés candidates reste inchangée. En effet, durant l'attaque, nous détectons et éliminons les mêmes clés avec les mêmes paires dans les deux cas.

Remarque 6.4.2. Une autre manière, équivalente, de voir le gain de la technique *state-test* sur la complexité en temps est de considérer que nous avons 2×4 bits de sous-clés en moins à tester (ceux de $RK_0\rfloor_3$ et $RK_1\rfloor_0$), mais par contre, nous ne filtrons pas de clé sur le deuxième tour réduisant ainsi la valeur de $c_{in} = 16 - 4$. La formule (6.3) devient alors

$$\left(C_N + \left(2^{20-8}\frac{N}{2^{16-4}} + N\right)C'_E + 2^{|K|-(20-8)}\mathcal{P}2^{20-8}\right)C_E. \tag{6.7}$$

Cela ne change évidemment rien sur l'évaluation finale de la complexité en temps puisque les formules (6.7) et (6.6) sont égales.

Application de la technique state-test en parallèle.

Avec cette technique, nous fixons certains bits des clairs (ou des chiffrés), disons f bits, à une valeur constante. La quantité de données disponibles, et donc le nombre de paires N que nous pouvons construire sont donc diminués. Or, la probabilité $\mathcal{P} = (1-2^{-(c_{in}+c_{out})})^N$ qu'une clé reste dans l'ensemble des clés candidates est directement liée à la valeur N. Plus le nombre de paires N est faible, plus la probabilité $\mathcal{P} \simeq e^{-N\times 2^{-(c_{in}+c_{out})}}$ est élevée, et moins notre attaque éliminera de clés. Dans une telle situation, le terme dominant de la complexité en temps, donnée par la formule (6.2), est généralement celui donné pour "finir" la recherche de la clé, c'est-à-dire $2^{|K|}\mathcal{P}$.

Plus précisément, nous avons besoin que la somme de $\log_2(C_N)$ et de f, le nombre de bits que nous avons fixés, soit inférieur à la taille d'un bloc, s :

$$\log_2(C_N) + f < s.$$

Pour éviter que la technique *state-test* n'augmente la complexité en temps de notre attaque lorsqu'on est limité en le nombre de données disponibles, nous pouvons la répéter en *parallèle* un certains nombre de fois, disons $Y \leq 2^f$ fois. Par répéter en parallèle, nous voulons dire appliquer la technique *state-test* plusieurs fois en même temps, mais pour différentes valeurs fixées des données. Dans ce cas, les données et la mémoire nécessaires sont multipliées par Y. De plus, répéter l'attaque en parallèle permet de détecter plus efficacement, lors du filtrage, si une clé testée peut être la bonne clé de chiffrement. En effet, pour une certaine clé testée, si aucune des Y listes ne contient toutes les valeurs x', nous pouvons regarder s'il s'agit de la clé de chiffrement que nous cherchons. La probabilité qu'une clé reste dans l'ensemble des clés candidates devient donc $\mathcal{P}^Y$.

En résumé, même si nous multiplions par Y les données disponibles C_N, ainsi que le nombre de paires disponibles à tester N, la probabilité de garder une clé dans l'ensemble des clés candidates diminue quant à elle exponentiellement. La formule (6.2) devient donc

$$T_{comp} = \left(C_N \times Y + \left(2^{|k_{in}\cup k_{out}|-f}\frac{N\times Y}{2^{c_{in}+c_{out}}} + N\times Y\right)C'_E + 2^{|K|}\mathcal{P}^Y\right)C_E. \tag{6.8}$$

Dans les sections qui suivent, nous appliquons cette technique pour réaliser une cryptanalyse différentielle impossible de CLEFIA-128 réduit à 13 tours (section 6.5), ainsi qu'à Camellia-256 réduit à 14 tours (section 6.6).

6.5. Applications sur CLEFIA

CLEFIA est un chiffrement par blocs de 128 bits conçu par Shirai et al. pour SONY en 2007 [184]. Il a été adopté comme standard ISO/IEC 29192 en cryptographie à bas coût. C'est un schéma de Feistel généralisé à quatre branches dont les spécifications précises sont décrites à l'annexe B.1. Il admet trois tailles de clé différentes : 128, 192 et 256 bits et nous appellerons donc chaque version CLEFIA-128, CLEFIA-192 et CLEFIA-256 respectivement. Un nombre différent R d'itérations est prévu pour chaque version de CLEFIA, comme décrit à la table B.1.

Du fait de sa standardisation, CLEFIA est un chiffrement par blocs ayant attiré l'attention de beaucoup de chercheurs, et de nombreuses attaques ont été publiées sur des versions réduites [197, 196, 195, 151, 192, 27]. La plupart d'entre elles utilisent des différentielles impossibles. Cependant, comme nous l'ont signalés les concepteurs de CLEFIA, certaines cryptanalyses présentent des défauts (voir notamment table 6.1).

Dans ce qui suit, nous allons améliorer la cryptanalyse différentielle impossible de CLEFIA-128 réduit à 13 tours fournie par Mala et al. [151] qui est la meilleure attaque connue sur CLEFIA-128. Nous rappelons les différentielles et les paramètres qu'ils utilisent dans leur attaque. Notre but ici est de montrer que nous sommes capables d'améliorer les complexités de l'attaque en utilisant les techniques vues à la section précédente.

Dans [196], Tsunoo et al. remarquent qu'il existe une différentielle impossible sur 9 tours ($\delta_X \nrightarrow_9 \delta_Y$) de CLEFIA, avec $\delta_X = (0,0,0,A) \in (\mathbb{F}_2^{32})^4$ et $\delta_Y = (0,0,B,0) \in (\mathbb{F}_2^{32})^4$, où $A, B \in (\mathbb{F}_2^8)^4$ sont tels qu'un seul de leurs quatre octets est non nul à des positions distinctes. Dans notre attaque, nous utilisons la même différentielle impossible (voir [196, Section 3.1]) et la plaçons entre les tours 3 et 11.

Les chemins différentiels ($\delta_{in} \to \delta_X$) et ($\delta_{out} \to \delta_Y$) sont décrits à la figure 6.7. Dans cette figure, nous pouvons voir les paramètres de l'attaque. Plus précisément la probabilité de vérifier la différentielle ($\delta_{in} \to \delta_X$), avec

$$\delta_{in} = (0,0,0,0)|(*,0,0,0)|M_0(*,0,0,0)|(*,*,*,*) \in (\mathbb{F}_2^8)^4 \times (\mathbb{F}_2^8)^4 \times (\mathbb{F}_2^8)^4 \times (\mathbb{F}_2^8)^4,$$

est $2^{-c_{in}} = 2^{-40}$. De même, la probabilité de vérifier la différentielle ($\delta_{out} \to \delta_Y$), avec

$$\delta_{out} = (0,*,0,0)|M_1(*,0,0,0)|(*,*,*,*)|(0,0,0,0) \in (\mathbb{F}_2^8)^4 \times (\mathbb{F}_2^8)^4 \times (\mathbb{F}_2^8)^4 \times (\mathbb{F}_2^8)^4,$$

est $2^{-c_{out}} = 2^{-40}$. D'où $c_{in} + c_{out} = 40 + 40 = 80$ et $|\delta_{in}| = |\delta_{out}| = 48$.

En suivant l'analyse de complexité de la section 6.2, nous savons que nous devons avoir au moins $N_{\min} = 2^{c_{in}+c_{out}} = 2^{80}$ paires dont les entrées ont une différence δ_{in} et les sorties ont une différence δ_{out}. Le coût pour construire ces $N_{\min}$ paires, c'est-à-dire le nombre minimal de données qu'il faut chiffrer (ou déchiffrer), est

$$C_{N_{\min}} = \max\left\{\sqrt{2^{80}2^{129-48}}, 2^{80}2^{129-48-48}\right\} = 2^{113}.$$

6.5.1. Utilisation de la technique state-test

Nous allons maintenant utiliser la *state-test* technique, décrite à la sous-section 6.4.2, sur les 8 bits de l'état interne notés x dans la figure 6.7. Pour cela, nous devons fixer une partie des 32 bits de la branche la plus à gauche des clairs. Puisque le nombre minimal de données dont nous avons besoin pour construire les $N_{\min}$ paires est $C_{N_{\min}} = 2^{113}$, nous pouvons fixer

$$s - C_{N_{\min}} = 128 - 113 = 15 \quad \text{bits},$$

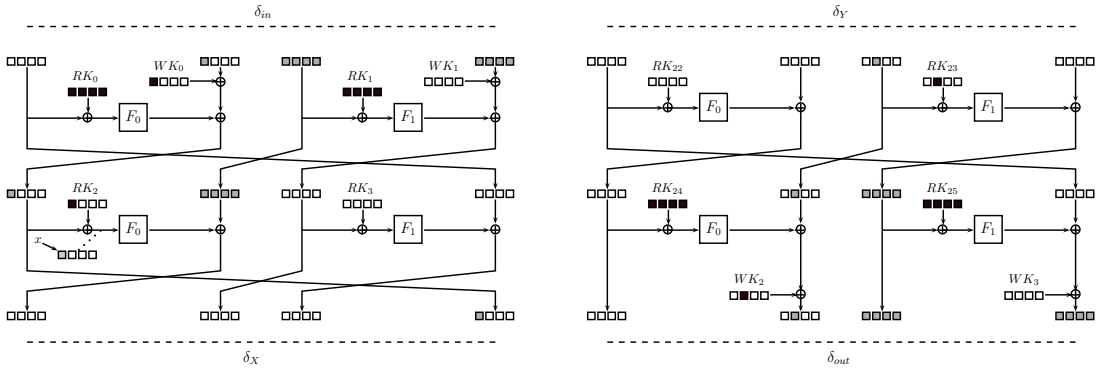


FIGURE 6.7. : Chemins différentiels sur CLEFIA-128.
■ : octet avec une différence non nulle. ■ : octet de sous-clés que nous devons tester.

sur les données. Cependant, puisque chaque boîte-S est une fonction $\mathbb{F}_2^8 \to \mathbb{F}_2^8$, nous ne pouvons fixer qu'un multiple de 8 bits si nous voulons que la technique *state-test* fonctionne. Nous fixons donc seulement un octet sur la branche la plus à gauche des données claires. Nous n'aurons donc besoin de tester que les 24 bits de la sous-clé RK_0 situés sur les autres octets.

Si nous n'utilisons pas la technique *state-test*, nous devrions tester :

		32 bits de RK_1				8 bits de $RK_{23} \oplus WK_2$
	+	32 bits de RK_0			+	32 bits de RK_{24}
	+	8 bits de $RK_2 \oplus WK_0$			+	32 bits de RK_{25}
k_{in}	=	72 bits		k_{out}	=	72 bits

Cependant, suite à l'analyse de l'algorithme de cadencement de clés de CLEFIA-128, nous remarquons que les sous-clés RK_1 et RK_{24} ont 22 bits en commun. Ainsi, sans utiliser la technique *state-test*, le nombre de bits de clé que nous devrions tester serait

$$|k_{in} \cup k_{out}| = 72 + 72 - 22 = 122 \quad \text{bits.}$$

Grâce aux 8 bits que nous avons fixés sur les clairs, nous pouvons dire que nous n'avons que $|k_{in} \cup k_{out}| - 8 = 122 - 8 = 114$ bits de clé à tester durant notre attaque avec la technique *state-test*.

Nous évaluons le coût d'un chiffrement partiel par $C'_E = 18/104$ puisqu'il n'y a que 18 boîtes-S actives dans les chemins différentiels $(\delta_{in} \to \delta_X)$ et $(\delta_{out} \to \delta_Y)$ et 13 tours de CLEFIA-128 comportent 104 boîtes-S. Nous en déduisons donc, à l'aide de la formule (6.2), que la complexité en temps de notre cryptanalyse différentielle de CLEFIA-128 réduit à 13 tours est donnée par

$$\left(C_N \times Y + \left(2^{114}\frac{N \times Y}{2^{80}} + N \times Y\right)\frac{18}{104} + 2^{128}\mathcal{P}^Y\right) C_E,$$

où $1 \leq Y \leq 2^8$ est le nombre de fois que nous appliquons la technique *state-test* en parallèle.

Notre attaque requiert un minimum de $C_{N_{\min}} = 2^{113}$ messages clairs, et puisque nous avons fixé 8 bits des clairs à une valeur constante, il ne nous reste que $128 - 113 - 8 = 7$

bits de liberté pour construire des structures. À l'aide de ces bits de liberté sur les données, la meilleure complexité en temps que nous obtenons est $2^{116.9}C_E$ en utilisant $2^{83.33}$ paires construites à partir de $2^{116.33}$ messages clairs.

6.5.2. Utilisation des différentielles impossibles multiples

Dans [196], les auteurs remarquent qu'il existe plusieurs différentielles impossibles distinctes sur 9 tours de CLEFIA (voir table 6.2) :

$\delta_X \nrightarrow_9 \delta_Y$	
δ_X	δ_Y
$(0,0,0,A)$	$(0,0,0,B)$
$(0,A,0,0)$	$(0,B,0,0)$

A	B		
$(0,0,0,\alpha)$	$(0,0,\beta,0)$	$(0,\beta,0,0)$	$(\beta,0,0,0)$
$(0,0,\alpha,0)$	$(0,0,0,\beta)$	$(0,\beta,0,0)$	$(\beta,0,0,0)$
$(0,\alpha,0,0)$	$(0,0,0,\beta)$	$(0,0,\beta,0)$	$(\beta,0,0,0)$
$(\alpha,0,0,0)$	$(0,0,0,\beta)$	$(0,0,\beta,0)$	$(0,\beta,0,0)$

TABLE 6.2. : Différentielles impossibles sur 9 tours de CLEFIA

Dans [197], les auteurs utilisent ces différentielles impossibles multiples pour cryptanalyser 12 tours de CLEFIA-128. Ici, nous allons utiliser notre approche formalisée (voir sous-section 6.4.1) de la cryptanalyse différentielle impossible utilisant les différentielles impossibles multiples, pour améliorer la complexité en données de l'attaque sur 13 tours de CLEFIA-128 de Mala et al. [151].

Nous utilisons ici les 24 différentielles impossibles qui sont décrites à la table 6.2. Nous avons donc $n_{in} = 2 \times 4$ différents choix pour la différence δ_X, et pour chacune d'entre elles $n_{out} = 3$ différents δ_Y telles que $\delta_X \nrightarrow_9 \delta_Y$.

Nous avons donc, en reprenant les notations de la sous-section 6.4.1,

$$|\delta'_{in}| = |\delta_{in}| + \log_2(8) = 48 + 3 \quad \text{et} \quad |\delta'_{out}| = |\delta_{out}| + \log_2(3) = 48 + 1.58.$$

Comme les probabilités pour vérifier les chemins différentiels restent inchangés, *i.e.* $c_{in} + c_{out} = 80$, le nombre minimal de paires vérifiant la différentielle $(\delta_{in} \to \delta_{out})$ est toujours $N_{\min} = 2^{80}$. Nous avons donc

$$C_{N_{\min}} = \max\left\{\sqrt{2^{80}2^{129-51}}, 2^{80}2^{129-52-49.58}\right\} = 2^{113-4.58}.$$

Nous avons aussi toujours $|k_{in} \cup k_{out}| = 122$ bits de clé à tester.

En conclusion, en faisant varier le nombre de données que nous voulons utiliser dans notre attaque, la meilleure complexité en temps, de cette cryptanalyse avec les différentielles impossibles multiples, que nous obtenons est $2^{122.26}C_E$ en utilisant $2^{82.6}$ paires construites à partir de $2^{111.02}$ messages clairs. Rappelons que le but de l'utilisation de cette technique est de diminuer la complexité en données plutôt que de diminuer la complexité en temps.

6.5.3. Combinaison de la technique state-test et des différentielles impossibles multiples

La technique *state-test* réduisant la complexité en temps, et les différentielles impossibles multiples réduisant la complexité en données, il parait naturel de combiner les deux pour obtenir un gain en temps *et* en données.

Dans ce paragraphe, nous prenons une seule possibilité parmi les δ_X, et deux δ_Y correspondants (voir table 6.2). En d'autres termes, nous choisissons $n_{in} = 1$ et $n_{out} = 2$. En considérant deux chemins possibles en sortie, le nombre de bits de sous-clés reste le même. Ainsi, nous avons

$$|\delta_{in}| = 48 \quad \text{et} \quad |\delta'_{out}| = |\delta_{out}| + \log_2(2) = 48 + 1.$$

Comme précédemment, la probabilité de chaque chemin différentiel ne varie pas, cela veut dire que $c_{in} + c_{out} = 80$, et le nombre minimal de paires pour que l'attaque soit valide est $N_{\min} = 2^{80}$. Pour ce nombre de paires, nous avons besoin de $C_{N_{\min}} = 2^{113-1} = 2^{112}$ messages clairs, ce qui nous permet de fixer 16 bits des clairs, pour pouvoir utiliser la technique *state-test*. Nous fixons donc 2 octets sur la branche la plus à gauche des clairs, ce qui nous autorise à ne tester que les 2 octets de RK_0 sur les autres positions.

Grâce aux 16 bits que nous avons fixés sur les clairs, nous pouvons dire que nous n'avons que $|k_{in} \cup k_{out}| - 16 = 122 - 16 = 106$ bits de clé à tester durant notre attaque avec la technique *state-test*.

En combinant les différentielles impossibles multiples et la technique *state-test*, nous calculons la complexité en temps de la cryptanalyse comme

$$\left(C_N + \left(2^{106}\frac{N}{2^{80}} + N\right)\frac{18}{104} + 2^{128}\mathcal{P}\right) C_E,$$

Si nous considérons $N = 2^{83.16}$ paires vérifiant les différentielles impossibles, nous avons besoin de $C_N = 2^{114.58}$ clairs pour les construire, et la complexité en temps est donc $2^{116.16} C_E$.

Nous résumons l'ensemble de ces attaques dans la table 6.3, et les comparons à la cryptanalyse différentielle impossible proposée par Mala et al. [151]. Notons tout de même, qu'avec l'utilisation de nos formules, de nombreux autres compromis de complexité sont possibles.

Version		# Tours	Temps	Données	Mémoire
CLEFIA-128	[151]	13	$2^{121.2}$	$2^{117.8}$	$2^{86.8}$
	state-test	13	$2^{116.90}$	$2^{116.33}$	$2^{83.33}$
	multiples	13	$2^{122.26}$	$\mathbf{2^{111.02}}$	$\mathbf{2^{82.60}}$
	multiple & *state-test*	13	$\mathbf{2^{116.16}}$	$\mathbf{2^{114.58}}$	$\mathbf{2^{83.16}}$

TABLE 6.3. : Résumé des différentes cryptanalyses différentielles impossibles contre CLEFIA-128, et comparaison avec la meilleure cryptanalyse différentielle impossible connue.

6.6. Applications sur Camellia

Camellia est un chiffrement par blocs de 128 bits conçu par Aoki et al. pour Mitsubishi et NTT en 2000 [7]. Il a été adopté comme standard ISO/IEC 18033 en cryptographie à bas coût en 2005 et a donc attiré l'attention de la communauté cryptographique. C'est un schéma de Feistel à deux branches un petit peu particulier, puisque des permutations FL et FL^{-1} sont appliquées sur les branches gauche et droite respectivement tous les six tours. Les spécifications complètes de Camellia sont décrites à l'annexe B.2. Il existe trois versions de Camellia, que nous notons Camellia-128, Camellia-192 et Camellia-256, dépendant de la taille de clé utilisée (128, 192 ou 256 bits respectivement). Un nombre différent R d'itérations est prévu pour chaque version de Camellia, comme décrit à la table B.3.

Tout comme pour CLEFIA, nombreuses sont les attaques sur des versions réduites de Camellia, et les plus efficaces utilisent des différentielles impossibles [211, 212, 152, 147, 149]. Dans certains travaux, des défauts dans les évaluations de complexité ont été découverts (voir table 6.1).

À cause de sa conception originale, due aux fonctions FL/FL^{-1} notamment, les cryptanalyses du chiffrement par blocs Camellia peuvent être classées suivant plusieurs catégories. Certaines cryptanalyses considèrent les fonctions FL/FL^{-1}, tandis que d'autres préfèrent s'en passer pour arriver à cryptanalyser plus de tours. Il en va de même pour les clés de blanchiments. De plus, de part la structure de Camellia, le tour où commence l'attaque a aussi son importance : les fonctions FL/FL^{-1}, rappelons-le, sont placées sur chaque branche tous les six tours d'un schéma de Feistel. Les meilleures attaques pour toutes les versions de Camellia, comprenant clés de blanchiments et fonctions FL/FL^{-1}, en termes de nombre de tours atteints et de complexité sont présentées par Liu et al. dans [147, Section 4.2]. Dans la sous-section suivante, nous fournissons des améliorations de leurs attaques au niveau de l'évaluation de la complexité. Ensuite, nous présenterons une cryptanalyse différentielle impossible de Camellia-256 sur 14 tours utilisant la technique *state-test*, mais qui ne tient compte ni des fonctions FL/FL^{-1}, ni des clés de blanchiments. Il s'agit de l'attaque sur le plus grand nombre de tours qui commence par le premier.

6.6.1. Améliorations des meilleures cryptanalyses différentielles impossibles sur Camellia

Nous reprenons exactement les mêmes paramètres ainsi que les mêmes chemins différentiels que ceux des attaques présentées dans [147]. Les paramètres pour l'attaque sur 11 tours de Camellia-128, 12 tours de Camellia-192 et 13 tours de Camellia-256 sont décrits à la table 6.4.

Version	$\|\delta_{in}\|$	$\|\delta_{out}\|$	r_{in}	r_{out}	r_δ	c_{in}	c_{out}	$\|k_{in} \cup k_{out}\|$	$C_{N_{\min}}$	C'_E
Camellia-128	23	80	1	2	8	32	57	96	2^{115}	7/88
Camellia-192	80	80	2	2	8	73	73	160	2^{115}	12/96
Camellia-256	80	128	2	3	8	73	121	224	2^{115}	20/104

TABLE 6.4. : Paramètres des cryptanalyses différentielles impossibles sur Camellia.

En faisant varier le nombre de données que nous utilisons, les meilleures complexités en temps obtenues à l'aide de l'analyse précise que nous avons fournie à la section 6.2 sont

décrites à la table 6.5. Nous les comparons avec les complexités données par les auteurs des attaques sur lesquels nous nous basons [147].

Version		# Tours	Temps	Données (C_N)	Mémoire (N)
Camellia-128	[147]	11	2^{122}	2^{122}	2^{98}
	cette section	11	$\mathbf{2^{118.43}}$	$\mathbf{2^{118.40}}$	$\mathbf{2^{92.4}}$
Camellia-192	[147]	12	$2^{187.2}$	2^{123}	$2^{155.41}$
	cette section	12	$\mathbf{2^{161.06}}$	$\mathbf{2^{119.70}}$	$\mathbf{2^{150.70}}$
Camellia-256	[147]	13	$2^{251.1}$	2^{123}	2^{203}
	cette section	13	$\mathbf{2^{225.06}}$	$\mathbf{2^{119.71}}$	$\mathbf{2^{198.71}}$

TABLE 6.5. : Résumé des cryptanalyses différentielles impossibles contre les différentes versions de Camellia, et comparaison avec la meilleure cryptanalyse différentielle impossible connue.

6.6.2. Utilisation de la technique state-test sur Camellia-256

Nous utilisons maintenant la technique *state-test* pour réaliser une cryptanalyse différentielle impossible sur 14 tours de Camellia-256, sans les fonctions FL/FL^{-1} et sans les clés de blanchiments. Cependant, contrairement à toutes les autres attaques de ce type qui ne commencent pas par le premier tour pour ne pas s'encombrer de l'asymétrie de l'algorithme du cadencement de clés de Camellia, notre attaque commence par le premier tour de chiffrement. La meilleure attaque de Camellia-256 de ce type (avec FL/FL^{-1} et sans clés de blanchiments) connue à ce jour est celle présentée par Liu et al. [147]. Cette attaque atteint aussi 14 tours mais ne commence pas par le premier tour de chiffrement pour exploiter une propriété des sous-clés sur les tours sur laquelle elle s'applique. Cette propriété est la raison principale pour laquelle ils arrivent à cryptanalyser 14 tours de chiffrement. Notre attaque est donc la première cryptanalyse (toutes cryptanalyses confondues) atteignant 14 tours de Camellia-256, sans les fonctions FL/FL^{-1}, en partant du premier tour.

Nous considérons donc la même différentielle impossible sur 8 tours que celle présentée par Mala et al. [152] mais la plaçons entre le quatrième tour et le douzième tour.

Nous résumons alors les paramètres de notre cryptanalyse différentielle impossible sur 14 tours de Camellia-256 dans la table 6.6.

$\lvert\delta_{in}\rvert$	$\lvert\delta_{out}\rvert$	r_{in}	r_{out}	r_δ	c_{in}	c_{out}	$\lvert k_{in} \cup k_{out}\rvert$	C'_E
128	56	4	2	8	120	48	227	20/112

TABLE 6.6. : Paramètres de l'attaque sur 14 tours de Camellia-256 (sans FL/FL^{-1} et sans clés de blanchiments).

Nous voyons que pour que notre attaque soit valide, il nous faut un minimum de $N_{\min} = 2^{c_{in}+c_{out}} = 2^{168}$ paires ayant la différentielle $(\delta_{in} \to \delta_{out})$. Le nombre de données nécessaires pour construire ces $N_{\min}$ paires est

$$C_{N_{\min}} = \max\left\{\sqrt{2^{168}2^{129-128}}, 2^{168}2^{129-184}\right\} = 2^{113}.$$

Nous disposons donc de $128 - 113 = 15$ bits de liberté pour utiliser la technique *state-test*. Puisque les boîtes-S de Camellia sont des fonctions $\mathbb{F}_2^8 \to \mathbb{F}_2^8$, nous fixons donc 8 bits sur la moitié de droite des chiffrés et appliquons la technique *state-test* sur 8 bits de l'état du pénultième tour. Le nombre de bits de clé que nous devons tester est donc $|k_{in} \cup k_{out}| = 227 - 8 = 219$ (en analysant l'algorithme de cadencement de clés de Camellia256, nous remarquons que 45 bits sont partagés entre les sous-clés).

La meilleure complexité en temps de notre cryptanalyse est $2^{221.34} C_E$ et est obtenue lorsque nous utilisons $N = 2^{172.78}$ paires formées à partir de $C_N = 2^{117.78}$ chiffrés. Nous résumons ces complexités à la table 6.7 et les comparons avec les complexités données par [147], même si notre attaque, rappelons-le, commence au premier tour de chiffrement de Camellia-256.

Version		# Tours	Temps	Données (C_N)	Mémoire (N)
Camellia-256	[147]	14	$2^{250.5}$	2^{120}	2^{125}
	cette section	14	$\mathbf{2^{220}}$	$\mathbf{2^{118}}$	2^{173}

TABLE 6.7. : Complexités de la cryptanalyse différentielle impossible sur 14 tours de Camellia-256 (sans FL/FL^{-1} et sans clés de blanchiments).

6.7. Application sur LBlock

LBlock est un chiffrement par blocs de 64 bits conçu par Wenling Wu et Lei Zhang [210] en 2011. Sa construction itérée sur 32 tours peut être vu comme une variante d'un schéma de Feistel à 2 branches. Une description précise de LBlock est présentée à l'annexe B.3. Ce chiffrement par blocs est paramétrée par une clé de taille 80 bits.

6.7.1. Cryptanalyse différentielle impossible sur 23 tours de LBlock

Nous remarquons premièrement qu'une différence

$$\delta_X = (0, A) \in (\mathbb{F}_2^{32})^2, \qquad \text{avec} \quad A = (0, 0, 0, 0, \alpha, 0, 0, 0) \in (\mathbb{F}_2^4)^8,$$

ne peut se propager en une différence

$$\delta_Y = (B, 0) \in (\mathbb{F}_2^{32})^2, \qquad \text{avec} \quad B = (0, 0, 0, 0, 0, \beta, 0, 0) \in (\mathbb{F}_2^4)^8,$$

après 14 tours de la fonction interne de LBlock. Cette différentielle impossible, que nous plaçons entre les tours 5 et 19, est représentée à la figure 6.8.

Les chemins différentiels $(\delta_{in} \to \delta_X)$ et $(\delta_{out} \to \delta_Y)$ sur $r_{in} = 5$ tours et $r_{out} = 4$ tours respectivement, sont décrits à la figure 6.9.

Les paramètres de notre cryptanalyse différentielle impossible sur 23 tours de LBlock sont donc résumés à la table 6.8.

Puisque $c_{in} + c_{out} = 44 + 28 = 72$, le nombre minimal de paires vérifiant la différentielle $(\delta_{in} \to \delta_{out})$, sur 23 tours de LBlock, qu'il nous faut pour que notre attaque soit valide est $N_{\min} = 2^{72}$. Le nombre de données dont nous avons besoin est alors

$$C_{N_{\min}} = \max\left\{\sqrt{2^{72} 2^{65-48}}, 2^{72} 2^{65-48-32}\right\} = 2^{57}.$$

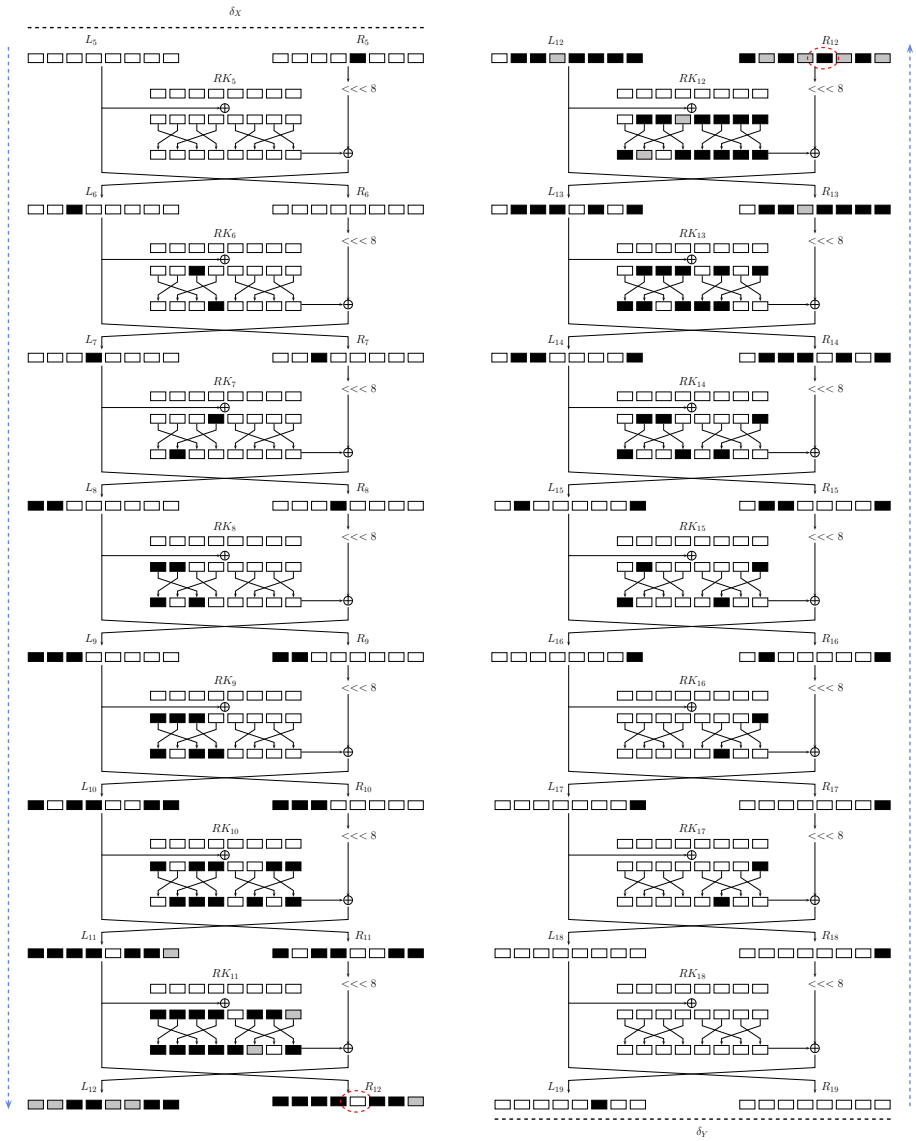
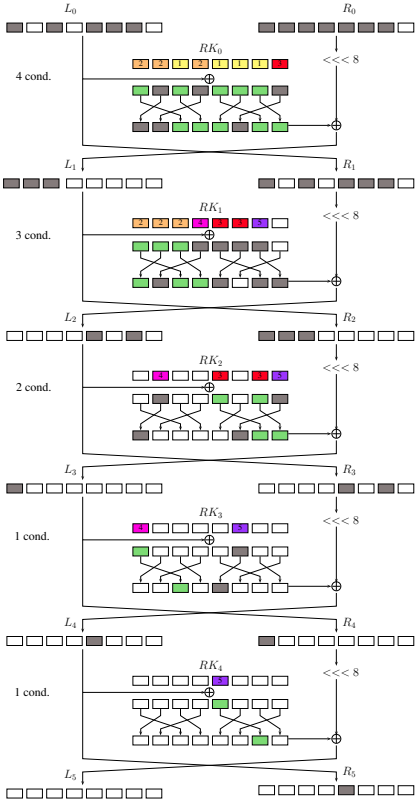


FIGURE 6.8. : Différentielle impossible sur 14 tours de LBlock.
■ : octet avec une différence non nulle. ■ : octet avec une différence inconnue.

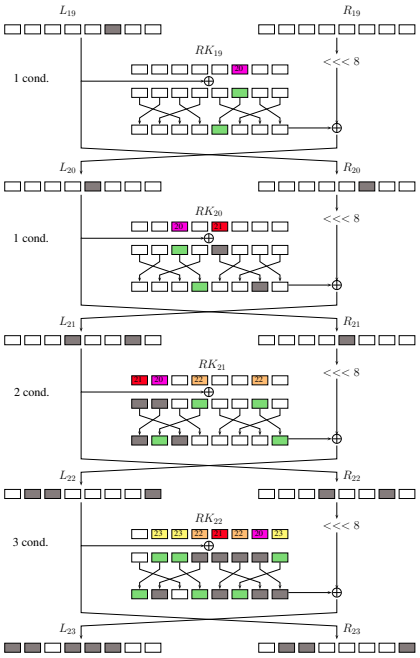
$\lvert\delta_{in}\rvert$	$\lvert\delta_{out}\rvert$	r_{in}	r_{out}	r_{δ}	c_{in}	c_{out}	$\lvert k_{in} \cup k_{out}\rvert$	C'_E
48	32	5	4	14	44	28	73	36/184

TABLE 6.8. : Paramètres de l'attaque sur 23 tours de LBlock.

En utilisant la formule (6.2), nous remarquons que la complexité en temps minimale que nous pouvons atteindre est $2^{74.06}C_E$ avec $N = 2^{74.6}$ paires vérifiant la différentielle $(\delta_{in} \rightarrow \delta_{out})$ construites à partir de $C_N = 2^{59.6}$ données claires.



(a) $(\delta_{in} \rightarrow \delta_X)$.



(b) $(\delta_{out} \rightarrow \delta_Y)$.

FIGURE 6.9. : Chemins différentiels sur LBlock. Les différentes couleurs correspondent aux octets de clés qu'il faut connaître suivant les tours de l'état.

6.7.2. Amélioration de la cryptanalyse différentielle sur 22 tours de LBlock

La meilleure cryptanalyse connue de LBlock était une cryptanalyse différentielle impossible atteignant 22 tours ayant une complexité en temps de $2^{79.28}C_E$ présentée par Ferhat Karakoç Hüsseyin Demirci et A. Emre Harmanci [119] en 2012.

Dans cette section, nous montrons une nouvelle fois l'efficacité de l'analyse générique présentée à la section 6.2 en reprenant exactement les mêmes paramètres pour monter la cryptanalyse [119], et en améliorant considérablement l'évaluation des complexités de cette attaque. Les résultats que nous obtenons sont décrits à la table 6.9 et sont comparés avec ceux de [119].

Version		# Tours	Temps	Données (C_N)	Mémoire (N)
LBlock	[119]	22	$2^{79.28}$	2^{58}	2^{57}
	cette section	22	$\mathbf{2^{71.53}}$	$\mathbf{2^{60}}$	2^{59}

TABLE 6.9. : Complexités de la cryptanalyse différentielle impossible sur 22 tours de LBlock.

6.8. Applications sur Simon

Simon est une famille de chiffrements par blocs à bas coût, optimisés pour les performances matérielles, récemment proposés par la NSA [12].

Il existe plusieurs versions de l'algorithme de chiffrement par blocs Simon, qui sont recensées à la table B.9. Une description complète des spécifications de Simon peut être trouvée à l'annexe B.4.

Dans le document de description de l'algorithme de chiffrement, les auteurs ne donnent aucun indice concernant la sécurité et la résistance aux cryptanalyses. C'est dans ce contexte que sont apparus de multiples travaux [4, 6, 5, 3] tentant d'évaluer la résistance de la famille de chiffrements par blocs Simon aux cryptanalyses impossibles différentielles, linéaires, différentielles, et les attaques dites *rectangles*. Les meilleurs résultats, en nombre de tours et complexité en temps, sont pour le moment dus aux cryptanalyses différentielles présentées par Abed et al. [4] et Biruykov et al. [5].

De plus, les cryptanalyses différentielles impossibles qui ont été proposées présentent des défauts au niveau de l'évaluation de la complexité. En effet, l'attaque présentée par Hoda AlKhzaimi et Martin Lauridsen [6] demande plus de données qu'il n'est possible d'en générer, l'invalidant donc. Dans les attaques présentées par Abed et al. [4, 3], la probabilité du chemin différentiel en entrée n'est pas calculée correctement. Nous verrons que, puisque nous utilisons les mêmes différentielles qu'eux, c_{in} devrait être 22, et non 10 comme ils le prétendent.

Dans ce qui suit, nous présentons donc les cryptanalyses différentielles impossibles pour toutes les versions de Simon. Nous appliquons notre analyse et nos formules génériques présentées à la section 6.2, ainsi que les différentielles impossibles multiples, pour fournir les meilleures cryptanalyses différentielles impossibles pour toutes les versions de Simon. Pour les variantes travaillant sur des plus petits blocs, ces attaques sont les meilleures connues, toutes cryptanalyses confondues.

Notre approche sera la même pour toutes les versions de Simon, excepté pour Simon-96/96 et Simon-128/128, c'est pourquoi nous ne détaillerons que la cryptanalyse différentielle impossible sur Simon-32/64 tandis que nous ne donnerons que les paramètres et les évaluations de complexités pour les autres versions.

Nous remarquons tout d'abord que sans les améliorations de la section 6.4.1, nous ne pourrions avoir de cryptanalyses différentielles impossibles sur aucune des versions de Simon. En effet, pour toutes les versions de Simon, et quelques soient les nombres de tours, r_{in}, r_{out}, des chemins différentiels, nous avons que $c_{in} = |\delta_{in}|$ et $c_{out} = |\delta_{out}|$ impliquant donc $C_{N_{\min}} \geq 2^{s+1}$. C'est pourquoi nous utilisons les différentielles impossibles multiples et utilisons le fait qu'à partir de différences δ_X et δ_Y de poids 1 bit, nous avons les deux différentielles impossibles

$$\delta_X \not\rightarrow_{r_\delta} \begin{cases} \delta_Y \\ \delta_Y \lll_2, \end{cases}$$

comme nous pouvons le voir à la table 6.10 pour la version Simon-32/64. Ainsi, pour toutes les versions, nous pourrons prendre $|\delta'_{out}| = |\delta_{out}| + 1$, nous assurant ainsi d'avoir suffisamment de données disponibles pour que nos attaques soient valides.

Les versions Simon-96/96 et Simon-128/128 bénéficieront d'une attention particulière, puisque l'usage de plus de différentielles impossibles multiples est une nécessité pour avoir une cryptanalyse différentielle impossible valide. Nous y reviendrons dans un paragraphe dédié.

Cryptanalyse différentielle impossible de SIMON-32/64

En cherchant exhaustivement les différentielles impossibles de poids faible, nous avons trouvé que les différentielles impossibles de poids 1 (en entrée et en sortie) couvrent 11 tours, et sont toutes des rotations les unes des autres. La différentielle impossible que nous avons choisi d'utiliser dans notre attaque peut être visualisée en détail à la table 6.10.

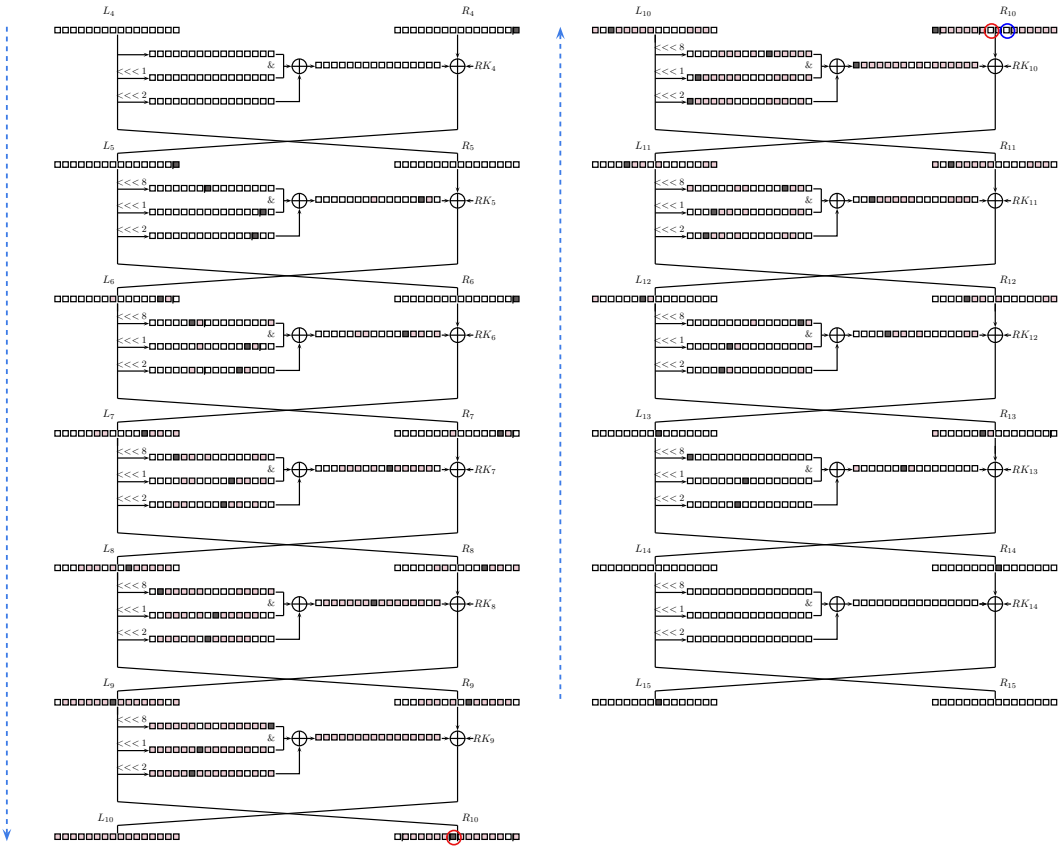


FIGURE 6.10. : Différentielle impossible sur 11 tours de SIMON-32/64.
■ : bit avec une différence non nulle. ■ : bit avec une différence inconnue.

Nous la plaçons entre les tours 5 et 16, et ajoutons des chemins différentiels (décrits par la figure 6.11) de telle manière que $r_{in} = 4$ et $r_{out} = 4$. Nous avons donc que le nombre de conditions à vérifier est $c_{in} = c_{out} = 22$ et que $|\delta_{in}| = 22$ et $|\delta'_{out}| = 23$.

Nous pouvons donc calculer $C_{N_{\min}}$ comme :

$$C_{N_{\min}} = \max\left\{\sqrt{2^{44}2^{33-23}}, 2^{44}2^{33-45}\right\} = 2^{32}.$$

Nous n'utilisons pas d'information de l'algorithme de cadencement de clés[4] de SIMON-32/64, et nous avons donc $|k_{in} \cup k_{out}| = 54$.

Nous évaluons le coût d'un chiffrement partiel de SIMON-32/64 par le ratio entre le nombre d'opérations non-linéaires (ici le & bit-à-bit) dans les chemins différentiels et le nombre total d'opérations non-linéaires dans la fonction de chiffrement : $C'_E = 54/304$.

[4]cela revient à considérer que les sous-clés sont indépendantes.

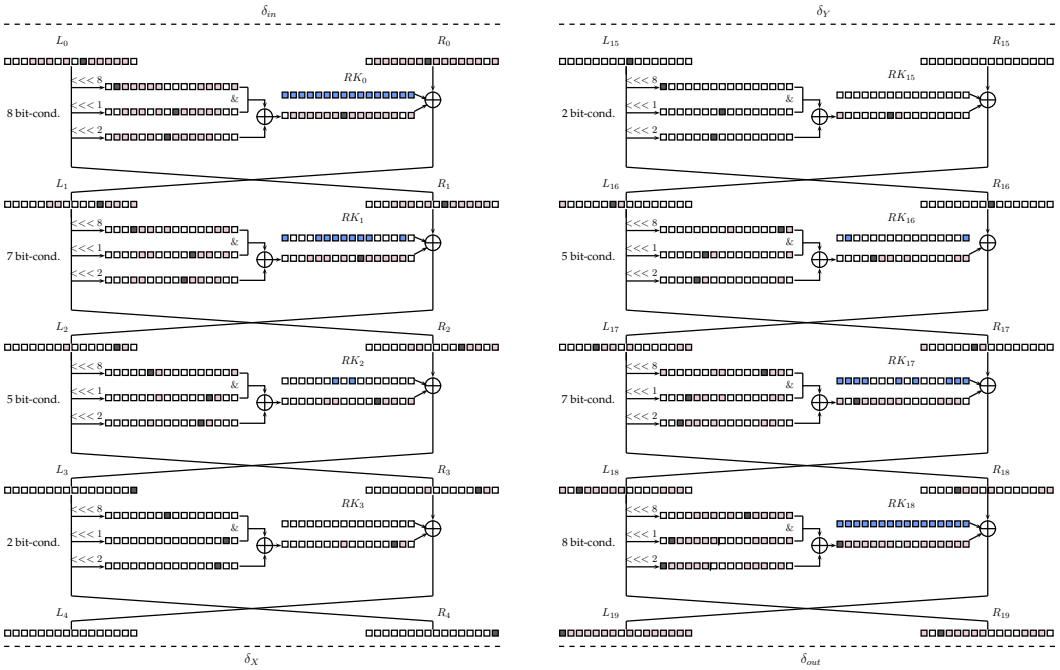


FIGURE 6.11. : Chemins différentiels sur SIMON-32/64.
■ : bit de sous-clé que nous devons tester. ■ : bit avec une différence inconnue. ■ : bit avec une différence non nulle.

En conclusion, notre cryptanalyse différentielle impossible de SIMON-32/64 sur 19 tours demande une complexité en temps $2^{62.56}C_E$, en utilisant $N = 2^{44}$ paires construites à partir de $C_N = 2^{32}$ données claires.

Résumé des attaques sur les autres versions de SIMON

Les attaques sur les autres versions de SIMON sont données à la table 6.10.

	SIMON 32/64	SIMON 48/72	SIMON 48/96	SIMON 64/96	SIMON 64/128	SIMON 96/144	SIMON 128/192	SIMON 128/256
# Tours	19	20	21	21	22	25	28	30
Temps	$2^{62.56}$	$2^{70.69}$	$2^{94.73}$	$2^{94.56}$	$2^{126.56}$	$2^{142.59}$	$2^{190.56}$	$2^{254.68}$
Données	2^{32}	2^{48}	2^{48}	2^{64}	2^{64}	2^{96}	2^{128}	2^{128}
Mémoire	2^{44}	2^{58}	2^{70}	2^{60}	2^{75}	2^{77}	2^{77}	2^{111}

TABLE 6.10. : Résumé des complexités des cryptanalyses différentielles impossibles sur les versions de SIMON avec la meilleure complexité en temps possible.

Utilisation de plus de différentielles impossibles multiples pour attaquer SIMON-96/96 et SIMON-128/128

Les cryptanalyses différentielles impossibles des versions de SIMON ayant la même taille pour la clé et les blocs ne peuvent être valides sans utiliser plus de différentielles impos-

sibles.

Nous ne détaillons que la version SIMON-96/96, mais la même méthode s'applique pour SIMON-128/128.

Nous considérons donc une cryptanalyse différentielle impossible sur 24 tours de SIMON-96/96, avec $r_{in} = r_{out} = 4$ et $r_\delta = 16$. Nous utilisons $n_{in} = 8$ différentielles impossibles différentes ayant chacune $n_{out} = 2$ incompatibilités comme décrites ci dessous :

$$\delta_X \lll i \nrightarrow_{16} \delta_Y = \begin{cases} \delta_X \lll i+1 \\ \delta_X \lll i+3 \end{cases}, \quad \text{avec } i \in \{0, 7, 8, 16, 19, 25, 31, 37\},$$

où $\delta_X \in \mathbb{F}_2^{48}$ est une différence de poids un. Ainsi $|\delta'_{in}| = 33$ et $|\delta'_{out}| = 31$. Si nous considérons $N = 2^{61}$ paires construites à partir de $C_N = 2^{94}$ données, la complexité en temps de notre attaque est de $2^{94.62} C_E$. Nous utilisons le même nombre de différentielles impossibles pour l'attaque sur 27 tours de SIMON-128/128 ($r_{in} = r_{out} = 4$). Le résumé des complexités pour les cryptanalyses différentielles impossibles sur SIMON-96/96 et SIMON-128/128 sont données à la table 6.11.

Version	# Tours	Temps	Données (C_N)	Mémoire (N)
SIMON-96/96	24	$2^{94.62}$	2^{94}	2^{61}
SIMON-128/128	27	$2^{126.6}$	2^{126}	2^{61}

TABLE 6.11. : Complexités de la cryptanalyse différentielle impossible sur 24 tours de SIMON-96/96 et 27 tours de SIMON-128/128.

6.9. Résumé des résultats

Algorithme	# Tours	Temps	Données	Mémoire	Référence
CLEFIA-128	13	$2^{121.2}$	$2^{117.8}$	$2^{86.8}$	[151]
state-test	13	$2^{116.90}$	$2^{116.33}$	$2^{83.33}$	section 6.5.1*
multiples	13	$2^{122.26}$	$\mathbf{2^{111.02}}$	$\mathbf{2^{82.60}}$	section 6.5.2*
multiples & *state-test*	13	$\mathbf{2^{116.16}}$	$\mathbf{2^{114.58}}$	$\mathbf{2^{83.16}}$	section 6.5.3*
Camellia-128	11	2^{122}	2^{122}	2^{98}	[147]
	11	$\mathbf{2^{118.43}}$	$\mathbf{2^{118.4}}$	$\mathbf{2^{92.4}}$	section 6.6.1*
Camellia-192	12	$2^{187.2}$	2^{123}	$2^{155.41}$	[147]
	12	$\mathbf{2^{161.06}}$	$\mathbf{2^{119.70}}$	$\mathbf{2^{150.70}}$	section 6.6.1*
Camellia-256	13	$2^{251.1}$	2^{123}	2^{203}	[147]
	13	$\mathbf{2^{225.06}}$	$\mathbf{2^{119.71}}$	$\mathbf{2^{198.71}}$	section 6.6.1*
Camellia-256 †	14	$2^{250.5}$	2^{120}	2^{125}	[147]
state-test	14	$\mathbf{2^{220}}$	$\mathbf{2^{118}}$	2^{173}	section 6.6.2
LBlock	22	$2^{79.28}$	2^{58}	$2^{72.67}$	[119]
	22	$\mathbf{2^{71.53}}$	2^{60}	$\mathbf{2^{59}}$	section 6.7
	23	$2^{74.06}$	$2^{59.6}$	$2^{74.6}$	section 6.7*
Simon-32/64	**19**	$2^{62.56}$	2^{32}	2^{44}	section 6.8*
Simon-48/72	**20**	$2^{70.69}$	2^{48}	2^{58}	section 6.8*
Simon-48/96	**21**	$2^{94.73}$	2^{48}	2^{70}	section 6.8*
Simon-64/96	**21**	$2^{94.56}$	2^{64}	2^{60}	section 6.8
Simon-64/128	**22**	$2^{126.56}$	2^{64}	2^{75}	section 6.8
Simon-96/96	**24**	$2^{94.62}$	2^{94}	2^{61}	section 6.8
Simon-96/144	**25**	$2^{142.59}$	2^{96}	2^{77}	section 6.8
Simon-128/128	**27**	$2^{126.6}$	2^{126}	2^{61}	section 6.8
Simon-128/192	**28**	$2^{190.56}$	2^{128}	2^{77}	section 6.8
Simon-128/256	**30**	$2^{254.68}$	2^{128}	2^{111}	section 6.8

Table 6.12. : Résumé des meilleures *cryptanalyses différentielles impossibles* sur CLEFIA-128, Camellia, LBlock et Simon, et présentations de nos résultats. '*' mentionne s'il s'agit de la meilleure attaque connue contre le chiffrement (plus petites complexités temps/données/mémoire pour le plus grand nombre de tours) en question. † mentionne que les fonctions FL/FL^{-1} et les clés de blanchiments ne sont pas considérées.

6.10. Conclusion

La cryptanalyse différentielle impossible fait partie des attaques les plus performantes pour cryptanalyser des systèmes de cryptographie symétrique, mais la haute technicité qu'elle demande pour être mise en place rend souvent son application délicate. De plus, la vérification de ces applications contre des systèmes est souvent difficile, et bien souvent, des défauts sont trouvés dans les calculs d'évaluation de complexité ou les parties les plus techniques visant à trouver les paramètres des attaques. Notre compréhension des problèmes posés par la cryptanalyse différentielle impossible, les formules génériques, l'automatisation de la recherche des paramètres et les nouvelles techniques d'amélioration nous ont permis de proposer une unification et une optimisation significative de la cryptanalyse différentielle impossible.

Nous avons tout d'abord recensé quelques problèmes rencontrés par d'autres auteurs pour monter de telles attaques contre des chiffrements par blocs. Nous avons aussi proposé des formules génériques qui uniformisent l'évaluation de complexité de ces attaques et qui évitent ainsi les erreurs de calcul et d'approximation. Grâce à l'outil automatique que nous avons conçu et développé, nous proposons un moyen efficace de faciliter la recherche de chemins différentiels et le calcul des paramètres, qui est sans doute la partie la plus technique d'une cryptanalyse différentielle impossible.

Cette approche clarifiée et simplifiée de la cryptanalyse différentielle impossible nous a permis de proposer de nouvelles idées, de nouvelles techniques qui permettent d'optimiser ces attaques. La technique *state-test*, que nous avons conçue, permet de diminuer la complexité en temps d'une attaque, en diminuant le nombre de bits d'information de clé qu'il nous faut tester dans les chemins différentiels. D'un autre côté, nous avons formalisé et adapté à notre scénario d'attaque générique, l'utilisation des différentielles impossibles multiples [197] pour diminuer la complexité en données.

Nous avons finalement prouvé l'efficacité de notre méthode en cryptanalysant un panel de schémas de Feistel généralisés, fournissant pour certains d'entre eux les meilleures attaques connues (tout type confondu) à ce jour.

Nous espérons que, dans le futur, ces résultats simplifieront et amélioreront les cryptanalyses différentielles impossibles contre les chiffrements par blocs, pas seulement les schémas de Feistel, et leurs possibles combinaisons avec d'autres attaques. Par exemple, dans [216], Zheng Yuan, Xian Li et Haixia Liu proposent une combinaison des cryptanalyses différentielles impossibles et linéaires. De plus, l'application de notre technique pourrait très bien être tout aussi efficace contre les chiffrements par blocs de type réseaux de permutation-substitution. Ces voies sont prometteuses et méritent des études approfondies.

Annexes

A. Preuves

A.1. Preuves du Chapitre 3

A.1.1. Preuve du Lemme 3.3.8

Lemme 3.3.8. Soit la fonction $F_{-1,-3,1} : x \mapsto x^{-1} + \mathrm{Tr}(x^{-3})$ sur $\mathbb{F}_{2^n}$. Nous avons alors que :

- Si n est pair, alors la fonction $F_{-1,-3,1}$ est bijective sur $\mathbb{F}_{2^n}$.
  Elle satisfait $\delta(F_{-1,-3,1}) = 6$ si et seulement s'il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que
  $$\mathrm{Tr}(\alpha^{-3}) = 0, \qquad \mathrm{Tr}((\alpha+1)^{-1}) = 0 \qquad \text{et } \mathrm{Tr}(\alpha^{-1}) = 1 \ ;$$
- Si n est impair, alors la fonction $F_{-1,-3,1}$ est 2-to-1 sur $\mathbb{F}_{2^n}$.
  Elle satisfait $\delta(F_{-1,-3,1}) = 4$ si et seulement s'il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que
  $$\mathrm{Tr}(\alpha^{-3}) = 0, \qquad \mathrm{Tr}((\alpha+1)^{-1}) = 0 \qquad \text{et } \mathrm{Tr}(\alpha^{-1}) = 1.$$

De plus, pour tout $n \geq 38$, il existe toujours au moins un élément $\alpha \in \mathbb{F}_{2^n}^*$ vérifiant ces conditions.

En premier lieu, nous appliquons la Proposition 3.3.7 : la fonction $F_{-1,-3,1}$ est une permutation sur $\mathbb{F}_{2^n}$ lorsque n est pair, et est une fonction 2-to-1 lorsque n est impair. Cette même proposition nous donne une borne supérieur sur l'uniformité différentielle de la fonction $F_{-1,-3,1}$.

Nous allons maintenant utiliser les notations, les formules ainsi que les Propriétés $(p1)$-$(p4)$ de la sous-section3.3.2. Pour tout élément $\alpha \in \mathbb{F}_{2^n}^*$, si $x \notin \{0, \alpha\}$ nous pouvons écrire que $f_\alpha(x) = \alpha(x^2+\alpha x)^{-1}$ et

$$\begin{aligned}\Delta_\alpha H(x) &= \frac{1}{x^3} + \frac{1}{(x+\alpha)^3} = \frac{x^3 + (x+\alpha)^3}{(x^2+\alpha x)^3} = \frac{\alpha x^2 + x\alpha^2 + \alpha^3}{(x^2+\alpha x)^3} \\ &= \frac{\alpha}{(x^2+\alpha)^2} + (f_\alpha(x))^3 = \frac{(f_\alpha(x))^2}{\alpha} + (f_\alpha(x))^3 . \end{aligned} \tag{A.1}$$

Notons aussi que

$$f_\alpha(x) = \frac{1}{\alpha}, \quad x \notin \{0, \alpha\} \qquad \Rightarrow \qquad \mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = \mathrm{Tr}(2/\alpha^3) = 0. \tag{A.2}$$

Cela signifie donc que l'équation $E(\alpha, 1/\alpha + 1)$ a au plus quatre solutions possibles, puisque $f_\alpha(x) + \mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1/\alpha + 1$ avec $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1$ est impossible.

Posons le vecteur binaire $u = (u_0, u_1, u_2) \in \mathbb{F}_2^3$, et définissons la propriété $(\mathcal{E}_u)$ sur $\alpha \in \mathbb{F}_{2^n}^*$ par :

$$(\mathcal{E}_u) \ : \quad \mathrm{Tr}(\alpha^{-3}) = u_0, \qquad \mathrm{Tr}\left((\alpha+1)^{-1}\right) = u_1, \qquad \mathrm{Tr}\left(\alpha^{-1}\right) = u_2. \tag{A.3}$$

1. Supposons que n est pair. Nous prenons donc $u = (0, 0, 1)$. Notre but est de démontrer que $\delta(F_{-1,-3,1}) = 6$ si et seulement s'il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ telle que la Proriété $(\mathcal{E}_u)$ est vérifiée. Conformément à la relation (A.2), l'équation $E(\alpha, 1/\alpha)$ possède six solutions x si et seulement si l'équation

$$f_\alpha(x) = \frac{1}{\alpha} \qquad \text{avec } \mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 0, \tag{A.4}$$

possède quatre solutions et l'équation

$$f_\alpha(x) = \frac{1+\alpha}{\alpha} \qquad \text{avec } \mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1, \tag{A.5}$$

possède 2 solutions.

Les quatre solutions de l'équation (A.4) sont les deux solutions de l'équation $x^2 + \alpha x + \alpha^2 = 0$ ainsi que $\{0, \alpha\}$. Cependant, $x = 0$ est une solution si et seulement si $\mathrm{Tr}\left((\Delta_\alpha H)(0)\right) = \mathrm{Tr}\left(\alpha^{-3}\right) = 0$ qui est la première conidition de $(\mathcal{E}_u)$. Si l'équation (A.5) est satisfaite aussi, alors $f_\alpha(x) = (1+\alpha)/\alpha$ où $\alpha \neq 1$ car $f_\alpha(x) \neq 0$ pour tout x puisque la fonction $x \mapsto x^{-1}$ est bijective sur $\mathbb{F}_{2^n}$. L'équation (A.5) est donc satisfaite si et seulement si $\mathrm{Tr}(1/(\alpha+1)) = 0$ et $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1$, qui est

$$\begin{aligned}\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) &= \mathrm{Tr}\left(\frac{(\alpha+1)^3}{\alpha^3} + \frac{(\alpha+1)^2}{\alpha^3}\right) = \mathrm{Tr}\left(\frac{\alpha^3+\alpha}{\alpha^3}\right) \\ &= \mathrm{Tr}(1+\alpha^{-2}) = \mathrm{Tr}(\alpha^{-1}) = 1.\end{aligned}$$

Nous obtenons donc les deuxième et troisième conditions de $(\mathcal{E}_u)$ respectivement.

2. Supposons que n est impair. L'équation $E(\alpha, 1/\alpha)$ possède quatre solutions lorsque les équations (A.4) et (A.5) ont chacune deux solutions. Nous allons prouver c'est le cas lorsqu'il existe un élément $\alpha \in \mathbb{F}_{2^n}^*$ tel que la Propriété $\mathcal{E}_u$ est satisfaite pour $u = (0, 0, 0)$. Nous ne donnons qu'une idée de cette preuve puisque la méthode est similaire à celle où n est pair.

Les solutions de l'équation (A.4) sont $\{0, \alpha\}$ avec $\mathrm{Tr}(\alpha^{-3}) = 0$. De plus, l'équation $f_{aaa}(x) = (\alpha+1)/\alpha$ doit avoir deux solutions $\{x, x+\alpha\}$ avec $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1$. Les conditions sont donc

$$\mathrm{Tr}\left((\alpha+1)^{-1}\right) = 0 \qquad \text{et} \qquad \mathrm{Tr}(1+\alpha^{-2}) = \mathrm{Tr}(\alpha^{-1}) + 1 = 1, \quad \textit{i.e.} \quad \mathrm{Tr}(\alpha^{-1}) = 0.$$

L'existence d'un triplet $u = (u_0, u_1, u_2) \in \mathbb{F}_2^3$ pour un entier n suffisamment large, qui satisfait la Propriété $(\mathcal{E}_u)$ provient d'un résultat de Stephen Cohen [76, Théorème 1.1]. Nous avons d'abord besoin de la définition suivante.

Définition A.1.1. Soient des polynômes $f_0(x)$, $f_1(x)$ et $f_2(x) \in \mathbb{F}_{2^n}[x]$. Nous disons que leurs fonctions respectives sur $\mathbb{F}_{2^n}$, f_i, forment un ensemble *fortement linéairement indépendant* sur $\mathbb{F}_2$ s'il n'existe pas de vecteur $(v_0, v_1, v_2) \in \mathbb{F}_2^3$ tel que pour tout $x \in \mathbb{F}_{2^n}$

$$v_0 f_0(x) + v_1 f_1(x) + v_2 f_2(x) = h^2(x) + h(x) + \beta, \tag{A.6}$$

pour $\beta \in \mathbb{F}_{2^n}$ et un polynôme $h(x)\mathbb{F}_{2^n}[x]$.

Théorème A.1.2 ([76, Théorème 1.1])**.** *Soient des polynômes $f_0(x)$, $f_1(x)$ et $f_2(x) \in \mathbb{F}_{2^n}[x]$ dont les fonctions respectives sur $\mathbb{F}_{2^n}$ forment un ensemble* fortement linéairement indépendant *sur $\mathbb{F}_2$. De plus, pour $i = 0, 1, 2$, soit*

$$f_i(x) = \frac{c_i(x)}{d_i(x)}, \qquad \text{où } c_i(x), d_i(x) \in \mathbb{F}_{2^n}[x] \text{ sont premiers entre eux.}$$

Soit $m = \max\{\deg(c_i), \deg(d_i),\ i = 0, 1, 2\}$. Alors il existe un élément $\alpha \in \mathbb{F}_{2^n}$, primitif, avec

$$Tr(f_0(\alpha)) = u_0, \qquad Tr(f_1(\alpha)) = u_1, \qquad Tr(f_2(\alpha)) = u_2,$$

pour tout triplet $(u_0, u_1, u_2)\mathbb{F}_2^3$ fixé, tant que

$$n > 4\left(3 + \log_2(\ell m)\right), \quad \ell = 9 \times 8 \times 3. \tag{A.7}$$

En prenant des polynômes $f_0(x) = x^{-3}$, $f_1(x) = (x+1)^{-1}$ et $f_2(x) = x^{-1}$, nous pouvons montrer qu'ils forment un ensemble fortement linéairement indépendant sur $\mathbb{F}_2$. En effet, supposons que l'équation (A.6) est vérifiée pour un certain triplet $(v_0, v_1, v_2) \in \mathbb{F}_2^3$ et écrivons $h(x) = h_0(x)/h_1(x)$ où $h_0(x), h_1(x) \in \mathbb{F}_{2^n}[x]$ sont des polynômes premiers entre eux. Nous avons alors

$$\frac{v_0(x+1) + v_1x^3 + v_2x^2(x+1) + \beta x^3(x+1)}{x^3(x+1)} = \frac{h_0^2(x) + h_0(x)h_1(x)}{h_1^2(x)},$$

où $h_1^2(x)$ et $h_0^2(x) + h_0(x)h_1(x)$ sont premiers entre eux. Donc nous devons avoir $h_1(x) \in \{1, x\}$, puisque $h_1^2(x)$ doit diviser $x^3(x+1)$. D'où, $x(x+1)$ divise

$$B(x) = v_0(x+1) + v_1x^3 + v_2x^2(x+1) + \beta x^3(x+1),$$

impliquant que $B(0) = 0$ et $B(1) = 0$. Cela nous amène à dire que $v_0 = 0$ et $v_1 = 0$. Notons que l'équation (A.6) implique maintenant que $\mathrm{Tr}(v_2 f_2(x)) = \mathrm{Tr}(\beta)$ pour tou x, ce qui est impossible si $v_3 \neq 0$.

En appliquant maintenant l'inégalité (A.7) avec $m = 3$, nous trouvons que $n \geq 38$, ce qui conclut la preuve.

A.1.2. Preuve du Théorème 3.3.10

Théorème 3.3.10. Soit un entier positif n pair et un élément $\gamma \in \mathbb{F}_{2^n}^*$. Alors les fonctions sur $\mathbb{F}_{2^n}$

$$F_{-1,t,\gamma} : x \mapsto x^{-1} + \gamma \mathrm{Tr}(x^t), \qquad \text{avec } t \in \{3, 5\},$$

vérifient $\delta(F_{-1,t,\gamma}) = 4$ pour n'importe quel élément γ tel que $\gamma^t = 1$, $t \in \{3, 5, \}$.

Nous utilisons dans cette preuve les mêmes notations et les mêmes méthodes que dans la sous-section précédente. Nous posons la fonction $H : x \mapsto x^t$ sur $\mathbb{F}_{2^n}$ et nous commençons par calculer $(\Delta_\alpha H)(x)$ pour $x \notin \{0, \alpha\}$. Rappelons que dans ce cas précis, $f_\alpha(x) = \alpha(x^2 + \alpha x)^{-1}$. Si $t = 3$ alors

$$(\Delta_\alpha H)(x) = \alpha x^2 + x\alpha^2 + \alpha^3 = \frac{\alpha^2}{f_\alpha(x)} + \alpha^3. \tag{A.8}$$

Si $t = 5$ alors

$$(\Delta_\alpha H)(x) = \alpha x^4 + x\alpha^4 + \alpha^5 = \frac{\alpha^3}{(f_\alpha(x))^2} + \frac{\alpha^4}{f_\alpha(x)} + \alpha^5, \tag{A.9}$$

puisque $\alpha x^4 + x\alpha^4 = \alpha^5 \left(\frac{(x^2+\alpha x)^2}{\alpha^4} + \frac{x^2+\alpha x}{\alpha^2} \right)$.

Maintenant, conformément aux équations (3.12), (3.13) et aux propriétés $(p1)$-$(p4)$, nous regardons les solutions x de

$$F_\alpha(x) = f_\alpha(x) + \gamma \mathrm{Tr}\left(h_\alpha(x)\right) = \beta, \qquad \beta \in \{1/\alpha, 1/\alpha + \gamma\}. \tag{A.10}$$

Notons que si $x \in \{0, \alpha\}$ alors $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = \mathrm{Tr}(\alpha^t)$. Rappelons que $\delta(F_{-1,t,\gamma}) = 6$ si et seulement si l'une des deux équations $E(\alpha, 1/\alpha)$ ou $E(\alpha, 1/\alpha + \gamma)$ possède six solutions. Nous allons montrer que cela est impossible.

Assumons pour le moment que $t = 3$. Notons que lorsque $f_\alpha(x) = 1/\alpha$, $x \notin \{0, \alpha\}$, nous obtenons $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 0$, d'après la relation (A.8). Cela implique que l'équation $E(\alpha, 1/\alpha + \gamma)$ possède au plus quatre solutions, car il est impossible d'avoir $F_\alpha(x) = 1/\alpha + \gamma$ avec $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1$. Maintenant, l'équation $E(\alpha, 1/\alpha)$ possède quatre solutions lorsque $\mathrm{Tr}(\alpha^3) = 0$. Elles sont $\{0, \alpha\}$, ainsi que les deux solutions de l'équation $x^2 + \alpha x + \alpha^2 = 0$.

Il y a deux autres solutions si et seulement s'il existe $x \notin \{0, \alpha\}$ tel que $f_\alpha(x) = 1/\alpha + \gamma$ avec $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1$. Remarquons que, $f_\alpha(x) = 1/\alpha + \gamma$ si et seulement si

$$x^2 + \alpha x + \frac{\alpha^2}{\alpha\gamma + 1} = 0, \qquad \textit{i.e.} \quad \mathrm{Tr}\left(\frac{1}{\alpha\gamma + 1}\right) = 0. \tag{A.11}$$

Cependant, d'après la relation (A.8)

$$\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = \mathrm{Tr}\left(\frac{\alpha^3}{\alpha\gamma + 1} + \alpha^3\right) = \mathrm{Tr}\left(\frac{1}{\alpha\gamma + 1} + \alpha^3\right) = 0. \tag{A.12}$$

Cela est dû (lorsque $\gamma^3 = 1$) à

$$\frac{\alpha^3}{\alpha\gamma + 1} = \frac{((\alpha\gamma + 1) + 1)^3}{\alpha\gamma + 1} = (\alpha\gamma + 1)^2 + (\alpha\gamma + 1) + 1 + \frac{1}{\alpha\gamma + 1}.$$

Il est donc impossible d'avoir $f_\alpha(x) = 1/\alpha + \gamma$, avec $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 1$, $x \notin \{0, \alpha\}$. D'où le fait que l'équation $E(\alpha, 1/\alpha)$ possède au plus quatre solutions.

Assumons maintenant que $t = 5$. Lorsque $f_\alpha(x) = 1/\alpha$, $x \notin \{0, \alpha\}$, nous avons que $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = \mathrm{Tr}(\alpha^5)$, d'après la relation (A.9). Supposons que $\mathrm{Tr}(\alpha^5) = 0$. Dans ce cas là, nous obtenons quatre solutions, qui sont $\{0, \alpha\}$ et les deux solutions de l'équation $x^2 + \alpha x + \alpha^2 = 0$, $x \notin \{0, \alpha\}$. Nous procédons comme dans le cas où $t = 3$ pour prouver qu'il n'y a pas d'autres solutions. Lorsque $f_\alpha(x) = 1/\alpha + \gamma$, et en supposans que nous avons la relation (A.11), nous obtenons :

$$\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = \mathrm{Tr}\left(\frac{\alpha^5}{(\alpha\gamma + 1)^2} + \frac{\alpha^5}{(\alpha\gamma + 1)} + \alpha^5\right) = \mathrm{Tr}\left(\frac{1}{\alpha\gamma + 1} + \alpha^5\right) = 0,$$

ce qui donne en développant

$$\frac{\alpha^5}{(\alpha\gamma + 1)^\ell} = \frac{((\alpha\gamma + 1) + 1)^5}{(\alpha\gamma + 1)^\ell}, \qquad \text{pour } \ell = 1, 2, \quad \text{où } \gamma^5 = 1.$$

Cela implique donc que $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) \neq 1$, qui est une contradiction.

Supposons maintenant que $f_\alpha(x) = 1/\alpha$ avec $\mathrm{Tr}(\alpha^5) = 1$, et considérons le cas où $\beta = 1/\alpha + \gamma$. Nous avons quatre solutions, comme ci-dessus. Il y a deux autres solutions lorsque $f_\alpha(x) = 1/\alpha + \gamma$, en supposans que nous avons la relation (A.11) et que $\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = 0$. Cependant, dans ce cas nous avons

$$\mathrm{Tr}\left((\Delta_\alpha H)(x)\right) = \mathrm{Tr}\left(\frac{1}{\alpha\gamma + 1} + \alpha^5\right) = 0 + 1 = 1,$$

qui est une contradiction.

A.2. Preuve du Chapitre 4

A.2.1. Preuve du Lemme 4.3.11

Nous prouvons le Lemme 4.3.11 dans une version légèrement plus générale, les espaces vectoriels $Ker(\alpha_i)$ sont remplacés par des espaces vectoriels A_i avec la même propriété de distributivité de l'intersection sur la somme de ces espaces vectoriels.

Proposition A.2.1 (Principe d'inclusion-exclusion)**.** *Soient m sous-espaces vectoriels de dimension finie, $A_1, \ldots, A_m$, de l'espace vectoriel A, tels que l'intersection d'espaces vectoriels est distributive sur la somme de ces espaces vectoriels. Alors,*

$$\dim(\sum_{i=1}^{m} A_i) = \sum_{k=1}^{m}(-1)^{k+1}\left(\sum_{1\leq i_1<\cdots<i_k\leq m} \dim(A_{i_1} \cap \cdots \cap A_{i_k})\right). \tag{A.13}$$

Démonstration. Nous allons prouver ce résultat par récurrence sur m. L'initialisation de cette récurrence est un résultat classique que nous avons rappelé au Théorème 4.1.6 :

$$\dim(A_1 + A_2) = \dim(A_1) + \dim(A_2) - \dim(A_1 \cap A_2).$$

L'hypothèse de récurrence est donnée par l'équation (A.13). Soit maintenant un autre sous-espace vectoriel A_{m+1} de A vérifiant aussi la propriété de distributivité de l'intersection sur la somme avec les espaces vectoriels A_i. Alors,

$$\begin{aligned}\dim(\sum_{i=1}^{m+1} A_i) &= \dim(\sum_{i=1}^{m} A_i + A_{m+1}) \\ &= \dim\left(\sum_{i=1}^{m} A_i\right) + \dim(A_{m+1}) - \dim\left((\sum_{i=1}^{m} A_i) \cap A_{m+1}\right),\end{aligned}$$

où la denière égalité est obtenue en utilisant le Théorème 4.1.6. Nous pouvons utiliser le fait que l'intersection se distribue sur ces espaces vectoriels :

$$\left(\sum_{i=1}^{m} A_i\right) \cap A_{m+1} = \sum_{i=1}^{m}(A_i \cap A_{m+1}).$$

Nous appliquons maintenant l'hypothèse de récurrence :

$$\begin{aligned}\dim\left(\sum_{1\le i\le m}(A_i\cap A_{m+1})\right) &= \sum_{1\le i\le m}\dim(A_i\cap A_{m+1})\\ &\quad-\sum_{1\le i_1<i_2\le m}\dim(A_{i_1}\cap A_{i_2}\cap A_{m+1})\\ &\quad+\sum_{1\le i_1<i_2<i_3\le m}\dim(A_{i_1}\cap A_{i_2}\cap A_{i_3}\cap A_{m+1})\\ &\quad\vdots\\ &\quad+(-1)^{m+1}\dim(A_1\cap\cdots\cap A_{m+1})\\ &= \sum_{1\le k\le m}(-1)^{k+1}\left(\sum_{1\le i_1<\cdots<i_k\le m}\dim(A_{i_1}\cap\cdots\cap A_{i_k}\cap A_{m+1})\right).\end{aligned}$$

Nous appliquons aussi l'hypothèse de récurrence au terme $\dim(\sum_{i=1}^{m}A_i)$, ce qui nous donne

$$\begin{aligned}\dim(\sum_{i=1}^{m+1}A_i) &= \sum_{1\le i\le m}(-1)^{k+1}\left(\sum_{1\le i_1<\cdots<i_k\le m}\dim(A_{i_1}\cap\cdots\cap A_{i_m})\right)\\ &\quad+\dim(A_{m+1})\\ &\quad+\sum_{1\le i\le m}(-1)^{k+2}\left(\sum_{1\le i_1<\cdots<i_k\le m}\dim(A_{i_1}\cap\cdots\cap A_{i_m}\cap A_{m+1})\right)\\ &= \sum_{i=1}^{m}\dim(A_i)\\ &\quad+\dim(A_{m+1})\\ &\qquad-\sum_{1\le i_1<i_2\le m}\dim(A_{i_1}\cap A_{i_2})\\ &\qquad-\sum_{1\le i\le m}\dim(A_i\cap A_{m+1})\\ &\qquad\qquad\vdots\\ &\qquad+(-1)^{m+1}\dim(A_{i_1}\cap\cdots\cap A_{i_m})\\ &\qquad+(-1)^{m+1}\sum_{1\le i_1<\cdots<i_{m-1}\le m}\dim(A_{i_1}\cap\ldots A_{i_{m-1}}\cap A_{m+1})\\ &\qquad\quad+(-1)^{m+2}\dim(\bigcap_{1\le i\le m+1}A_i)\end{aligned}$$

$$
\begin{aligned}
&= \sum_{1\leq i\leq m+1} \dim(A_i) \\
&\qquad - \sum_{1\leq i_1<i_2\leq m+1} \dim(A_{i_1}\cap A_{i_2}) \\
&\qquad\qquad \vdots \\
&\qquad + (-1)^{m+1} \sum_{1\leq i_1<\cdots<i_{i_m}\leq m+1} \dim(A_{i_1}\cap\cdots\cap A_{i_m}) \\
&\qquad\qquad + (-1)^{m+2} \dim\left(\bigcap_{1\leq i\leq m+1} A_i\right) \\
&= \sum_{1\leq k\leq m+1} (-1)^{k+1} \left(\sum_{1\leq i_1<\cdots<i_k\leq m+1} \dim(A_{i_1}\cap\cdots\cap A_{i_k})\right),
\end{aligned}
$$

qui conclut la preuve. □

B. Spécifications des chiffrements par blocs du chapitre 6

B.1. Spécifications de CLEFIA

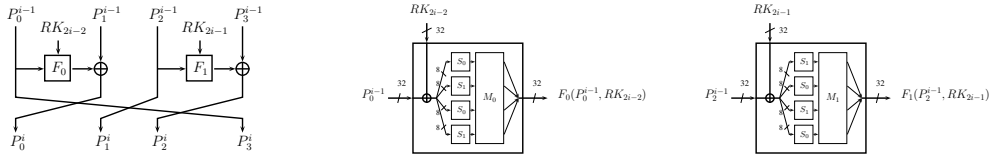
Version	Taille bloc	Taille clé	Nombre de tours
CLEFIA-128	128	128	18
CLEFIA-192	128	192	22
CLEFIA-256	128	256	26

TABLE B.1. : Paramètres et versions du chiffrement par blocs CLEFIA.

Algorithme 6 Algorithme de chiffrement de CLEFIA.

Entrée :
un clair $P = P_0 \mid P_1 \mid P_2 \mid P_3 \in \mathbb{F}_2^{128}$
des sous-clés $RK_i \in \mathbb{F}_2^x$, $0 \leq i < 2R$, et des clés de blanchiments $WK_0, WK_1, WK_2, WK_3 \in \mathbb{F}_2^{32}$.
Sortie :
un chiffré $C \in \mathbb{F}_2^{128}$.

Fonction CLEFIA-X(P, K)
1: $P_0^0 \mid P_1^0 \mid P_2^0 \mid P_3^0 \leftarrow P_0 \mid P1 \oplus WK_0 \mid P_2 \mid P_3 \oplus WK_1$
2: **Pour** $i = 1, \ldots, R$
3: $\quad P_0^i \leftarrow F_0(P_0^{i-1}, RK_{2i-2}) \oplus P_1^{i-1}$
4: $\quad P_1^i \leftarrow P_2^{i-1}$
5: $\quad P_2^i \leftarrow F_1(P_2^{i-1}, RK_{2i-1}) \oplus P_3^{-1}$
6: $\quad P_3^i \leftarrow P_0^{i-1}$
7: $C \leftarrow P_0^R \mid P_1^R \oplus WK_2 \mid P_2^R \mid P_3^R \oplus WK_3$
Retourner C.



(a) Un tour de CLEFIA. (b) F_0 (c) F_1

FIGURE B.1. : Les fonctions internes F_0 et F_1 de CLEFIA.

$$M_0 = \begin{pmatrix} 0\text{x}01 & 0\text{x}02 & 0\text{x}04 & 0\text{x}06 \\ 0\text{x}02 & 0\text{x}01 & 0\text{x}06 & 0\text{x}04 \\ 0\text{x}04 & 0\text{x}06 & 0\text{x}01 & 0\text{x}02 \\ 0\text{x}06 & 0\text{x}04 & 0\text{x}02 & 0\text{x}01 \end{pmatrix} \quad \text{et} \quad M_1 = \begin{pmatrix} 0\text{x}01 & 0\text{x}08 & 0\text{x}02 & 0\text{x}0a \\ 0\text{x}08 & 0\text{x}01 & 0\text{x}0a & 0\text{x}02 \\ 0\text{x}02 & 0\text{x}0a & 0\text{x}01 & 0\text{x}08 \\ 0\text{x}0a & 0\text{x}02 & 0\text{x}08 & 0\text{x}01 \end{pmatrix}.$$

B.1.1. Algorithme de cadencement de clés de CLEFIA

Même s'il existe trois versions différentes de cet algorithme de cadencement de clés, une pour chaque version de chiffrement de CLEFIA, nous ne verrons que la version traitant une clé de 128 bits, puisqu'il s'agit de la version qui nous intéresse pour la cryptanalyse que nous fournissons au chapitre 6. Les constantes utilisées dans l'algorithme 7 sont précisées dans [184]. Nous commençons tout d'abord par introduire une fonction essentielle au bon fonctionnement de cet algorithme, la fonction *double swap*.

Définition B.1.1 (Double swap)**.** La fonction *double swap*, notée Σ, est donnée par

$$\begin{array}{rccl} \Sigma: & \mathbb{F}_2^{128} & \to & \mathbb{F}_2^{128} \\ & X & \mapsto & X[7-63] \mid X[121-127] \mid X[0-6] \mid X[64-120], \end{array}$$

où $X[a-b]$ dénote la sous-séquence de bits entre la position a et la position b de $X \in \mathbb{F}_2^{128}$. La fonction double swap est illustrée à la figure B.2.

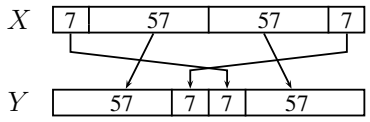


FIGURE B.2. : La fonction double swap Σ.

Algorithme 7 Algorithme de cadencement de clés de CLEFIA-128.

Entrée :
une clé $K = K_0 \mid K_1 \mid K_2 \mid K_3 \in (\mathbb{F}_2^{32})^4$ et 60 constantes $Con_i \in \mathbb{F}_2^{32}$, $0 \leq i < 24$.
Sortie :
des clés de blanchiments $WK_0, \ldots, WK_3$ et 36 clés de tour $RK_i \in \mathbb{F}_2^{32}$, $0 \leq i < 36$.

Fonction CADENCEMENT-CLEFIA-128(K, Con_i)
$L \leftarrow CLEFIA-128(K, Con_0, \ldots, Con_{23})_{12}$ ▷ 12 tours de CLEFIA-128
$WK_0 \mid WK_1 \mid WK_2 \mid WK_3 \leftarrow K_0 \mid K_1 \mid K_2 \mid K_3$
Pour $i = 1, \ldots, 8$
  $T \leftarrow L \oplus (Con_{24+4i} \mid Con_{24+4i+1} \mid Con_{24+4i+2} \mid Con_{24+4i+3})$
  $L \leftarrow \Sigma(L)$
  Si i est impair
    $T \leftarrow T \oplus K$
  $RK_{4i} \mid RK_{4i+1} \mid RK_{4i+2} \mid RK_{4i+3} \leftarrow T$.

B.1.2. Boîtes-S de CLEFIA

S_0	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	57	49	$d1$	$c6$	$2f$	33	74	fb	95	$6d$	82	ea	$0e$	$b0$	$a8$	$1c$
1.	28	$d0$	$4b$	92	$5c$	ee	85	$b1$	$c4$	$0a$	76	$3d$	63	$f9$	17	af
2.	bf	$a1$	19	65	$f7$	$7a$	32	20	06	ce	$e4$	83	$9d$	$5b$	$4c$	$d8$
3.	42	$5d$	$2e$	$e8$	$d4$	$9b$	$0f$	13	$3c$	89	67	$c0$	71	aa	$b6$	$f5$
4.	$a4$	be	fd	$8c$	12	00	97	da	78	$e1$	cf	$6b$	39	43	55	26
5.	30	98	cc	dd	eb	54	$b3$	$8f$	$4e$	16	fa	22	$a5$	77	09	61
6.	$d6$	$2a$	53	37	45	$c1$	$6c$	ae	ef	70	08	99	$8b$	$1d$	$f2$	$b4$
7.	$e9$	$c7$	$9f$	$4a$	31	25	fe	$7c$	$d3$	$a2$	bd	56	14	88	60	$0b$
8.	cd	$e2$	34	50	$9e$	dc	11	05	$2b$	$b7$	$a9$	48	ff	66	$8a$	73
9.	03	75	86	$f1$	$6a$	$a7$	40	$c2$	$b9$	$2c$	db	$1f$	58	94	$3e$	ed
$a.$	fc	$1b$	$a0$	04	$b8$	$8d$	$e6$	59	62	93	35	$7e$	ca	21	df	47
$b.$	15	$f3$	ba	$7f$	$a6$	69	$c8$	$4d$	87	$3b$	$9c$	01	$e0$	de	24	52
$c.$	$7b$	$0c$	68	$1e$	80	$b2$	$5a$	$e7$	ad	$d5$	23	$f4$	46	$3f$	91	$c9$
$d.$	$6e$	84	72	bb	$0d$	18	$d9$	96	$f0$	$5f$	41	ac	27	$c5$	$e3$	$3a$
$e.$	81	$6f$	07	$a3$	79	$f6$	$2d$	38	$1a$	44	$5e$	$b5$	$d2$	ec	cb	90
$f.$	$9a$	36	$e5$	29	$c3$	$4f$	ab	64	51	$f8$	10	$d7$	bc	02	$7d$	$8e$

S_1	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	$6c$	da	$c3$	$e9$	$4e$	$9d$	$0a$	$3d$	$b8$	36	$b4$	38	13	34	$0c$	$d9$
1.	bf	74	94	$8f$	$b7$	$9c$	$e5$	dc	$9e$	07	49	$4f$	98	$2c$	$b0$	93
2.	12	eb	cd	$b3$	92	$e7$	41	60	$e3$	21	27	$3b$	$e6$	19	$d2$	$0e$
3.	91	11	$c7$	$3f$	$2a$	$8e$	$a1$	bc	$2b$	$c8$	$c5$	$0f$	$5b$	$f3$	87	$8b$
4.	fb	$f5$	de	20	$c6$	$a7$	84	ce	$d8$	65	51	$c9$	$a4$	ef	43	53
5.	25	$5d$	$9b$	31	$e8$	$3e$	$0d$	$d7$	80	ff	69	$8a$	ba	$0b$	73	$5c$
6.	$6e$	54	15	62	$f6$	35	30	52	$a3$	16	$d3$	28	32	fa	aa	$5e$
7.	cf	ea	ed	78	33	58	09	$7b$	63	$c0$	$c1$	46	$1e$	df	$a9$	99
8.	55	04	$c4$	86	39	77	82	ec	40	18	90	97	59	dd	83	$1f$
9.	$9a$	37	06	24	64	$7c$	$a5$	56	48	08	85	$d0$	61	26	ca	$6f$
$a.$	$7e$	$6a$	$b6$	71	$a0$	70	05	$d1$	45	$8c$	23	$1c$	$f0$	ee	89	ad
$b.$	$7a$	$4b$	$c2$	$2f$	db	$5a$	$4d$	76	67	17	$2d$	$f4$	cb	$b1$	$4a$	$a8$
$c.$	$b5$	22	47	$3a$	$d5$	10	$4c$	72	cc	00	$f9$	$e0$	fd	$e2$	fe	ae
$d.$	$f8$	$5f$	ab	$f1$	$1b$	42	81	$d6$	be	44	29	$a6$	57	$b9$	af	$f2$
$e.$	$d4$	75	66	bb	68	$9f$	50	02	01	$3c$	$7f$	$8d$	$1a$	88	bd	ac
$f.$	$f7$	$e4$	79	96	$a2$	fc	$6d$	$b2$	$6b$	03	$e1$	$2e$	$7d$	14	95	$1d$

TABLE B.2. : Boîtes-S S_0 et S_1 de CLEFIA.

B.2. Spécifications de Camellia

Version	Taille bloc	Taille clé	Nombre de tours
Camellia-128	128	128	18
Camellia-192	128	192	24
Camellia-256	128	256	24

TABLE B.3. : Paramètres et versions du chiffrement par blocs Camellia.

Algorithme 8 Algorithme de chiffrement de Camellia.

Entrée :
un clair $P = L_0 \mid R_0 (\mathbb{F}_2^{64})^2$ et des sous-clés $RK_i \in \mathbb{F}_2^x$, $0 \leq i < R$.
Sortie :
un chiffré $C \in \mathbb{F}_2^{128}$.

Fonction CAMELLIA-X(P, K)
$L_0 \mid R_0 \leftarrow P \oplus (WK_0 \mid WK_1)$
Pour $i = 1, \ldots, R$
  $L_i \leftarrow R_{i-1} \oplus F(L_{i-1}, RK_i)$
  $R_i \leftarrow L_{i-1}$
  Si $i = 6, 12$ ou 18
    $L_i \leftarrow FL(L_i, kl_{i/3-1})$
    $R_i \leftarrow FL^{-1}(R_i, kl_{i/3})$
$C \leftarrow (L_R \mid R_R) \oplus (WK_2 \mid WK_3)$.
  Retourner C.

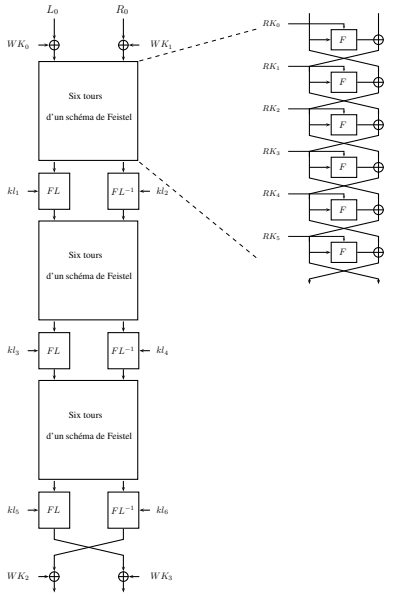
La fonction de diffusion est donnée par l'application d'une matrice binaire 8×8, P :

$$P = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

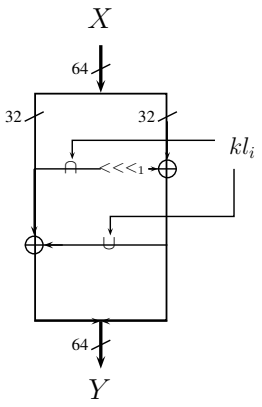
B.2.1. Algorithme de cadencement de clés de Camellia

Même si le chiffrement par blocs Camellia admet trois tailles de clé différentes, il n'existe que deux versions de l'algorithme de cadencement de clés : une pour les clés de 128 bits, et une pour les clés de 192 et 256 bits. Pour ces dernières, seule l'initialisation change. Notons K la clé principale. Nous définissons les variables KL et KR comme étant

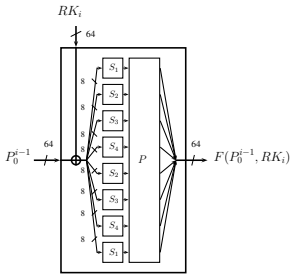
- $K = KL$ et $KR = 0$ si $K \in \mathbb{F}_2^{128}$,
- $K = KL \mid KR_L$ et $KR_R = \overline{KR_L}$ si $K \in \mathbb{F}_2^{192}$ et



(a) Description globale du schéma de Camellia-128.



(b) FL



(c) Fonction interne, F, de Camellia.

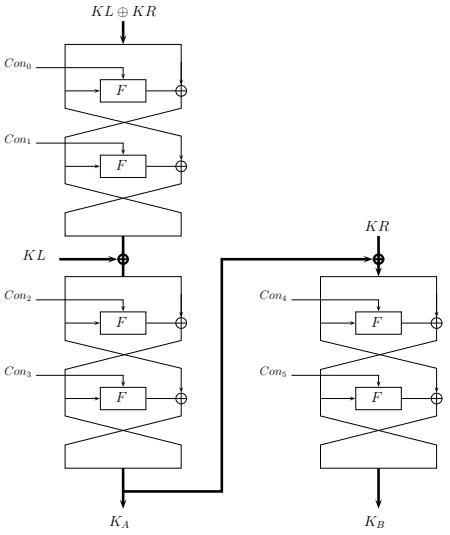
FIGURE B.3. : Description de la procédure de chiffrement de Camellia.

- $K = KL \mid KR$ si $K \in \mathbb{F}_2^{256}$,

où X_L et X_R signifient que nous prenons la moitié gauche ou droite de $X \in \mathbb{F}_2^x$ respectivement. La procédure pour obtenir deux nouvelles variables, $K_A, K_B \in \mathbb{F}_2^{128}$ est décrite à la figure B.4b. La fonction interne F est la même que celle utilisée lors du chiffrement avec Camellia. Les constantes $Con_i \in \mathbb{F}_2^{64}$, $0 \leq i \leq 5$ sont données à la table B.4a.

Con_0	$a09e667f3bcc908b$
Con_1	$b67ae8584caa73b2$
Con_2	$c6ef372fe94f82be$
Con_3	$54ff53a5f1d36f1c$
Con_4	$10e527fade682d1d$
Con_5	$b05688c2b3e6c1fd$

(a) Constantes utilisées dans le cadencement de clés de Camellia.



(b) Génération des variables pour le cadencement de clés de Camellia.

FIGURE B.4. : Schéma du cadencement de clé de Camellia.

Les sous-clés pour le chiffrement sont alors extraites suivant les tables B.4.

Clés de blanchiments	WK_0 WK_1	$(KL \lll 0)_L$ $(KL \lll 0)_R$
F (Tour 1)	RK_1	$(K_A \lll 0)_L$
F (Tour 2)	RK_2	$(K_A \lll 0)_R$
F (Tour 3)	RK_3	$(KL \lll 15)_L$
F (Tour 4)	RK_4	$(KL \lll 15)_R$
F (Tour 5)	RK_5	$(K_A \lll 15)_L$
F (Tour 6)	RK_6	$(K_A \lll 15)_R$
FL	kl_1	$(K_A \lll 30)_L$
FL^{-1}	kl_2	$(K_A \lll 30)_R$
F (Tour 7)	RK_7	$(KL \lll 45)_L$
F (Tour 8)	RK_8	$(KL \lll 45)_R$
F (Tour 9)	RK_9	$(K_A \lll 45)_L$
F (Tour 10)	RK_{10}	$(KL \lll 60)_R$
F (Tour 11)	RK_{11}	$(K_A \lll 60)_L$
F (Tour 12)	RK_{12}	$(K_A \lll 60)_R$
FL	kl_3	$(KL \lll 77)_L$
FL^{-1}	$kl4$	$(KL \lll 77)_R$
F (Tour 13)	RK_{13}	$(KL \lll 94)_L$
F (Tour 14)	RK_{14}	$(KL \lll 94)_R$
F (Tour 15)	RK_{15}	$(K_A \lll 94)_L$
F (Tour 16)	RK_{16}	$(K_A \lll 94)_R$
F (Tour 17)	RK_{17}	$(KL \lll 111)_L$
F (Tour 18)	RK_{18}	$(KL \lll 111)_R$
Clés de blanchiments	WK_2 WK_3	$(K_A \lll 111)_L$ $(K_A \lll 111)_R$

(a) Camellia-128.

Clés de blanchiments	WK_0 WK_1	$(KL \lll 0)_L$ $(KL \lll 0)_R$
F (Tour 1)	RK_1	$(K_B \lll 0)_L$
F (Tour 2)	RK_2	$(K_B \lll 0)_R$
F (Tour 3)	RK_3	$(KR \lll 15)_L$
F (Tour 4)	RK_4	$(KR \lll 15)_R$
F (Tour 5)	RK_5	$(K_A \lll 15)_L$
F (Tour 6)	RK_6	$(K_A \lll 15)_R$
FL	kl_1	$(KR \lll 30)_L$
FL^{-1}	kl_2	$(KR \lll 30)_R$
F (Tour 7)	RK_7	$(K_B \lll 30)_L$
F (Tour 8)	RK_8	$(K_B \lll 30)_R$
F (Tour 9)	RK_9	$(KL \lll 45)_L$
F (Tour 10)	RK_{10}	$(KL \lll 45)_R$
F (Tour 11)	RK_{11}	$(K_A \lll 45)_L$
F (Tour 12)	RK_{12}	$(K_A \lll 45)_R$
FL	kl_3	$(KL \lll 60)_L$
FL^{-1}	$kl4$	$(KL \lll 60)_R$
F (Tour 13)	RK_{13}	$(KR \lll 60)_L$
F (Tour 14)	RK_{14}	$(KR \lll 60)_R$
F (Tour 15)	RK_{15}	$(K_B \lll 60)_L$
F (Tour 16)	RK_{16}	$(K_B \lll 60)_R$
F (Tour 17)	RK_{17}	$(K_L \lll 77)_L$
F (Tour 18)	RK_{18}	$(K_L \lll 77)_R$
FL	kl_5	$(K_A \lll 77)_L$
FL^{-1}	kl_6	$(K_A \lll 77)_R$
F (Tour 19)	RK_{13}	$(KR \lll 94)_L$
F (Tour 20)	RK_{14}	$(KR \lll 94)_R$
F (Tour 21)	RK_{15}	$(K_A \lll 94)_L$
F (Tour 22)	RK_{16}	$(K_A \lll 94)_R$
F (Tour 23)	RK_{17}	$(KL \lll 111)_L$
F (Tour 24)	RK_{18}	$(KL \lll 111)_R$
Clés de blanchiments	WK_2 WK_3	$(K_B \lll 111)_L$ $(K_B \lll 111)_R$

(b) Camellia-192 et Camellia-256.

Table B.4. : Procédure du cadencement de clés pour Camellia

B.2.2. Boîtes-S de Camellia

S_0	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	70	82	2c	ec	b3	27	c0	e5	e4	85	57	35	ea	0c	ae	41
1.	23	ef	6b	93	45	19	a5	21	ed	0e	4f	4e	1d	65	92	bd
2.	86	b8	af	8f	7c	eb	1f	ce	3e	30	dc	5f	5e	c5	0b	1a
3.	a6	e1	39	ca	d5	47	5d	3d	d9	01	5a	d6	51	56	6c	4d
4.	8b	0d	9a	66	fb	cc	b0	2d	74	12	2b	20	f0	b1	84	99
5.	df	4c	cb	c2	34	7e	76	05	6d	b7	a9	31	d1	17	04	d7
6.	14	58	3a	61	de	1b	11	1c	32	0f	9c	16	53	18	f2	22
7.	fe	44	cf	b2	c3	b5	7a	91	24	08	e8	a8	60	fc	69	50
8.	aa	d0	a0	7d	a1	89	62	97	54	5b	1e	95	e0	ff	64	d2
9.	10	c4	00	48	a3	f7	75	db	8a	03	e6	da	09	3f	dd	94
a.	87	5c	83	02	cd	4a	90	33	73	67	f6	f3	9d	7f	bf	e2
b.	52	9b	d8	26	c8	37	c6	3b	81	96	6f	4b	13	be	63	2e
c.	e9	79	a7	8c	9f	6e	bc	8e	29	f5	f9	b6	2f	fd	b4	59
d.	78	98	06	6a	e7	46	71	ba	d4	25	ab	42	88	a2	8d	fa
e.	72	07	b9	55	f8	ee	ac	0a	36	49	2a	68	3c	38	f1	a4
f.	40	28	d3	7b	bb	c9	43	c1	15	e3	ad	f4	77	c7	80	9e

S_1	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	e0	05	58	d9	67	4e	81	cb	c9	0b	ae	6a	d5	18	5d	82
1.	46	df	d6	27	8a	32	4b	42	db	1c	9e	9c	3a	ca	25	7b
2.	0d	71	5f	1f	f8	d7	3e	9d	7c	60	b9	be	bc	8b	16	34
3.	4d	c3	72	95	ab	8e	ba	7a	b3	02	b4	ad	a2	ac	d8	9a
4.	17	1a	35	cc	f7	99	61	5a	e8	24	56	40	e1	63	09	33
5.	bf	98	97	85	68	fc	ec	0a	da	6f	53	62	a3	2e	08	af
6.	28	b0	74	c2	bd	36	22	38	64	1e	39	2c	a6	30	e5	44
7.	fd	88	9f	65	87	6b	f4	23	48	10	d1	51	c0	f9	d2	a0
8.	55	a1	41	fa	43	13	c4	2f	a8	b6	3c	2b	c1	ff	c8	a5
9.	20	89	00	90	47	ef	ea	b7	15	06	cd	b5	12	7e	bb	29
a.	0f	b8	07	04	9b	94	21	66	e6	ce	ed	e7	3b	fe	7f	c5
b.	a4	37	b1	4c	91	6e	8d	76	03	2d	de	96	26	7d	c6	5c
c.	d3	f2	4f	19	3f	dc	79	1d	52	eb	f3	6d	5e	fb	69	b2
d.	f0	31	0c	d4	cf	8c	e2	75	a9	4a	57	84	11	45	1b	f5
e.	e4	0e	73	aa	f1	dd	59	14	6c	92	54	d0	78	70	e3	49
f.	80	50	a7	f6	77	93	86	83	2a	c7	5b	e9	ee	8f	01	3d

S_2	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	38	41	16	76	$d9$	93	60	$f2$	72	$c2$	ab	$9a$	75	06	57	$a0$
1.	91	$f7$	$b5$	$c9$	$a2$	$8c$	$d2$	90	$f6$	07	$a7$	27	$8e$	$b2$	49	de
2.	43	$5c$	$d7$	$c7$	$3e$	$f5$	$8f$	67	$1f$	18	$6e$	af	$2f$	$e2$	85	$0d$
3.	53	$f0$	$9c$	65	ea	$a3$	ae	$9e$	ec	80	$2d$	$6b$	$a8$	$2b$	36	$a6$
4.	$c5$	86	$4d$	33	fd	66	58	96	$3a$	09	95	10	78	$d8$	42	cc
5.	ef	26	$e5$	61	$1a$	$3f$	$3b$	82	$b6$	db	$d4$	98	$e8$	$8b$	02	eb
6.	$0a$	$2c$	$1d$	$b0$	$6f$	$8d$	88	$0e$	19	87	$4e$	$0b$	$a9$	$0c$	79	11
7.	$7f$	22	$e7$	59	$e1$	da	$3d$	$c8$	12	04	74	54	30	$7e$	$b4$	28
8.	55	68	50	be	$d0$	$c4$	31	cb	$2a$	ad	$0f$	ca	70	ff	32	69
9.	08	62	00	24	$d1$	fb	ba	ed	45	81	73	$6d$	84	$9f$	ee	$4a$
$a.$	$c3$	$2e$	$c1$	01	$e6$	25	48	99	$b9$	$b3$	$7b$	$f9$	ce	bf	df	71
$b.$	29	cd	$6c$	13	64	$9b$	63	$9d$	$c0$	$4b$	$b7$	$a5$	89	$5f$	$b1$	17
$c.$	$f4$	bc	$d3$	46	cf	37	$5e$	47	94	fa	fc	$5b$	97	fe	$5a$	ac
$d.$	$3c$	$4c$	03	35	$f3$	23	$b8$	$5d$	$6a$	92	$d5$	21	44	51	$c6$	$7d$
$e.$	39	83	dc	aa	$7c$	77	56	05	$1b$	$a4$	15	34	$1e$	$1c$	$f8$	52
$f.$	20	14	$e9$	bd	dd	$e4$	$a1$	$e0$	$8a$	$f1$	$d6$	$7a$	bb	$e3$	40	$4f$

S_3	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	70	$2c$	$b3$	$c0$	$e4$	57	ea	ae	23	$6b$	45	$a5$	ed	$4f$	$1d$	92
1.	86	af	$7c$	$1f$	$3e$	dc	$5e$	$0b$	$a6$	39	$d5$	$5d$	$d9$	$5a$	51	$6c$
2.	$8b$	$9a$	fb	$b0$	74	$2b$	$f0$	84	df	cb	34	76	$6d$	$a9$	$d1$	04
3.	14	$3a$	de	11	32	$9c$	53	$f2$	fe	cf	$c3$	$7a$	24	$e8$	60	69
4.	aa	$a0$	$a1$	62	54	$1e$	$e0$	64	10	00	$a3$	75	$8a$	$e6$	09	dd
5.	87	83	cd	90	73	$f6$	$9d$	bf	52	$d8$	$c8$	$c6$	81	$6f$	13	63
6.	$e9$	$a7$	$9f$	bc	29	$f9$	$2f$	$b4$	78	06	$e7$	71	$d4$	ab	88	$8d$
7.	72	$b9$	$f8$	ac	36	$2a$	$3c$	$f1$	40	$d3$	bb	43	15	ad	77	80
8.	82	ec	27	$e5$	85	35	$0c$	41	ef	93	19	21	$0e$	$4e$	65	bd
9.	$b8$	$8f$	eb	ce	30	$5f$	$c5$	$1a$	$e1$	ca	47	$3d$	01	$d6$	56	$4d$
$a.$	$0d$	66	cc	$2d$	12	20	$b1$	99	$4c$	$c2$	$7e$	05	$b7$	31	17	$d7$
$b.$	58	61	$1b$	$1c$	$0f$	16	18	22	44	$b2$	$b5$	91	08	$a8$	fc	50
$c.$	$d0$	$7d$	89	97	$5b$	95	ff	$d2$	$c4$	48	$f7$	db	03	da	$3f$	94
$d.$	$5c$	02	$4a$	33	67	$f3$	$7f$	$e2$	$9b$	26	37	$3b$	96	$4b$	be	$2e$
$e.$	79	$8c$	$6e$	$8e$	$f5$	$b6$	fd	59	98	$6a$	46	ba	25	42	$a2$	fa
$f.$	07	55	ee	$0a$	49	68	38	$a4$	28	$7b$	$c9$	$c1$	$e3$	$f4$	$c7$	$9e$

TABLE B.6. : Boîtes-S de Camellia.

B.3. Spécifications de LBlock

Version	Taille bloc	Taille clé	Nombre de tours
LBlock	64	80	32

TABLE B.7. : Paramètres et versions du chiffrement par blocs LBlock.

Algorithme 9 Algorithme de chiffrement de LBlock.

Entrée :
un clair $P = L_0|R_0 \in (\mathbb{F}_2^{32})^2$ et une clé $K \in \mathbb{F}_2^{80}$.
Sortie :
un chiffré $C \in \mathbb{F}_2^{64}$.

Fonction LBLOCK(P,K)
1: $RK_0, \dots RK_{31} \leftarrow$ CADENCEMENT-LBLOCK(K) ▷ Algo 10
2: **Pour** $i = 1, \dots, 32$
3: $\quad R_i \leftarrow L_{i-1}$
4: $\quad L_i \leftarrow F(L_{i-1}, RK_{i-1}) \oplus (R_{i-1} \lll 8)$
5: $C \leftarrow R_{32}|L_{32}$ **Retourner** C

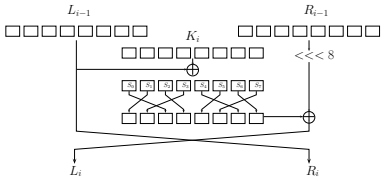


FIGURE B.5. : Un tour de LBlock.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S_0(x)$	e	9	f	0	d	4	a	b	1	2	8	3	7	6	c	5
$S_1(x)$	4	b	e	9	f	d	0	a	7	c	5	6	2	8	1	3
$S_2(x)$	1	e	7	c	f	d	0	6	b	5	9	3	2	4	8	a
$S_3(x)$	7	6	8	b	0	f	3	e	9	a	c	d	5	2	4	1
$S_4(x)$	e	5	f	0	7	2	c	d	1	8	4	9	b	a	6	3
$S_5(x)$	2	d	b	c	f	e	0	9	7	a	6	3	1	8	4	5
$S_6(x)$	b	9	4	e	0	f	a	d	6	c	5	7	3	8	1	2
$S_7(x)$	d	a	f	0	e	4	9	b	2	1	8	3	7	5	c	6
$S_8(x)$	8	7	e	5	f	d	0	6	b	c	9	a	2	4	1	3
$S_9(x)$	b	5	f	0	7	2	9	d	4	8	1	c	e	a	3	6

TABLE B.8. : Les dix boîtes-S de LBlock.

B.3.1. Algorithme de cadencement de clés de LBlock

Algorithme 10 Algorithme de cadencement de clés de LBlock.

Entrée :
une clé principale $K \in \mathbb{F}_2^{80}$.
Sortie :
32 sous-clés $RK_i \in \mathbb{F}_2^{32}$, $0 \leq i < 32$.

Fonction CADENCEMENT-LBLOCK(K)
1: $K_0 \leftarrow K$
2: $RK_0 \leftarrow K[79-48]$
3: **Pour** $i = 1, \ldots, 31$
4: $K_i \leftarrow K_{i-1} \lll 29$
5: $K_i[79-76] \leftarrow S_9(K_i[79-76])$
6: $K_i[75-72] \leftarrow S_8(K_i[75-72])$
7: $K_i[50-46] \leftarrow K_i[50-46] \oplus [\boldsymbol{i}]_5$ ▷ $[\boldsymbol{i}]_5$ est la repr. binaire de i sur 5 bits.
8: $RK_i \leftarrow K_i[79-48]$

B.4. Spécifications de SIMON

Version	Taille bloc	Taille clé	Nombre de tours
SIMON-32/64	32	64	32
SIMON-48/72	48	72	36
SIMON-48/96	48	96	36
SIMON-64/96	64	96	42
SIMON-64/128	64	128	44
SIMON-96/96	96	96	52
SIMON-96/144	96	144	54
SIMON-128/128	128	128	68
SIMON-128/192	128	192	69
SIMON-128/256	128	256	72

TABLE B.9. : Paramètres et versions du chiffrement par blocs SIMON.

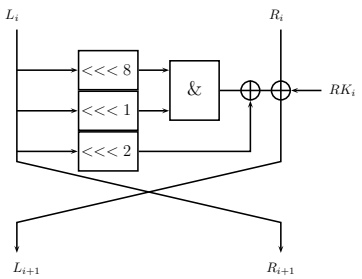


FIGURE B.6. : Un tour de SIMON.

B.4.1. Algorithme de cadencement de clés de SIMON

z	n	m	SIMON-$2n/nm$
4506230155203752166	16	2	SIMON-32/64
	24	3	SIMON-48/72
2575579794259089498	24	4	SIMON-48/96
3160415496042964403	32	3	SIMON-64/96
	48	2	SIMON-96/96
	64	2	SIMON-128/128
3957284701066611983	32	4	SIMON-64/128
	48	3	SIMON-96/144
	64	3	SIMON-128/192
3781244162168104175	64	4	SIMON-128/256

TABLE B.10. : Valeur de la constante z pour SIMON-$2n/nm$.

Algorithme 11 Algorithme de cadencement de clés de SIMON.

Entrée :
une clé $K = RK_0 \mid \cdots \mid RK_{m-1} \in (\mathbb{F}_2^n)^m$.

Sortie :
des sous-clés $RK_i \in \mathbb{F}_2^n$, $0 \leq i < R$.

Fonction CADENCEMENT-SIMON-$2n/nm(K)$

1: **Pour** $i = m, \ldots, R$
2: $\quad tmp \leftarrow RK_{i-1} \ggg 3$
3: $\quad$ **Si** $m = 4$
4: $\quad\quad tmp \leftarrow tmp \oplus RK_{i-3}$
5: $\quad tmp \leftarrow tmp \oplus (tmp \ggg 1)$
6: $\quad RK_i \leftarrow \overline{RK_{i-m}} \oplus tmp \oplus z[i - m \pmod{62}] \oplus 3$

Bibliographie

[1] *Proceedings of the 2010 IEEE International Symposium on Information Theory, ISIT 2010, Austin, Texas, USA, June 13-18, 2010*. IEEE, 2010.

[2] *Proceedings of the 2012 IEEE International Symposium on Information Theory, ISIT 2012, Cambridge, MA, USA, July 1-6, 2012*. IEEE, 2012.

[3] Farzaneh Abed, Eik List, Stefan Lucks et Jakob Wenzel : Differential and Linear Cryptanalysis of Reduced-Round Simon. *IACR Cryptology ePrint Archive*, 2013:526, 2013.

[4] Farzaneh Abed, Eik List, Jakob Wenzel et Stefan Lucks : Differential Cryptanalysis of round-reduced Simon and Speck. *In FSE 2014*, LNCS. Springer, 2014. À paraître.

[5] Javad Alizadeh, Nasour Bagheri, Praveen Gauravaram, Abhishek Kumar et Somitra Kumar Sanadhya : Linear Cryptanalysis of Round Reduced SIMON. *IACR Cryptology ePrint Archive*, 2013:663, 2013.

[6] Hoda AlKhzaimi et Martin M. Lauridsen : Cryptanalysis of the SIMON Family of Block Ciphers. *IACR Cryptology ePrint Archive*, 2013:543, 2013.

[7] Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima et Toshio Tokita : Camellia : A 128-Bit Block Cipher Suitable for Multiple Platforms - Design and Analysis. *In* Stinson et Tavares [188], pages 39–56.

[8] Ioannis G. Askoxylakis, Henrich Christopher Pöhls et Joachim Posegga, éditeurs. *Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems - 6th IFIP WG 11.2 International Workshop, WISTP 2012, Egham, UK, June 20-22, 2012. Proceedings*, volume 7322 de *Lecture Notes in Computer Science*. Springer, 2012.

[9] Yves Aubry, Gary McGuire et François Rodier : A few more functions that are not APN infinitely often. *Finite Fields : Theory and Applications-Selected Papers from the 9th International Conference Finite Fields ans Applications. Contemporary Mathematics*, 518:23–31, 2010.

[10] Jean Phillipe Aumasson et Willi Meier : Zero-sum distinguishers for reduced Keccak-f and for the core functions of Luffa and Hamsi, 2009. Rump Session de CHES'09.

[11] Roberto Maria Avanzi, Liam Keliher et Francesco Sica, éditeurs. *Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers*, volume 5381 de *Lecture Notes in Computer Science*. Springer, 2009.

[12] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks et Louis Wingers : The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013. http ://eprint.iacr.org/.

[13] T. D. Bending et Dmitry Fon-Der-Flaass : Crooked Functions, Bent Functions, and Distance Regular Graphs. *Electr. J. Comb.*, 5, 1998.

[14] Thierry P. Berger, Anne Canteaut, Pascale Charpin et Yann Laigle-Chapuy : On Almost Perfect Nonlinear Functions Over $\mathbb{F}_{2^n}$. *IEEE Transactions on Information Theory*, 52(9):4160–4170, 2006.

[15] Thomas Beth et Cunsheng Ding : On Almost Perfect Nonlinear Permutations. *In* Helleseth [107], pages 65–76.

[16] Jürgen Bierbrauer et Gohar M. M. Kyureghyan : Crooked binomials. *Des. Codes Cryptography*, 46(3):269–301, 2008.

[17] Eli Biham, Alex Biryukov et Adi Shamir : Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. *In* Stern [187], pages 12–23.

[18] Eli Biham et Adi Shamir : Differential Cryptanalysis of DES-like Cryptosystems. *J. Cryptology*, 4(1):3–72, 1991.

[19] Alex Biryukov, éditeur. *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, volume 4593 de *Lecture Notes in Computer Science*. Springer, 2007.

[20] Alex Biryukov, Guang Gong et Douglas R. Stinson, éditeurs. *Selected Areas in Cryptography - 17th International Workshop, SAC 2010, Waterloo, Ontario, Canada, August 12-13, 2010, Revised Selected Papers*, volume 6544 de *Lecture Notes in Computer Science*. Springer, 2011.

[21] Céline Blondeau : *La Cryptanalyse Différentielle et ses Généralisations*. Thèse de doctorat, UPMC, 2011.

[22] Céline Blondeau : Improbable Differential from Impossible Differential : On the Validity of the Model. *In* Paul et Vaudenay [174], pages 149–160.

[23] Céline Blondeau, Anne Canteaut et Pascale Charpin : Differential properties of power functions. *IJICoT*, 1(2):149–170, 2010.

[24] Céline Blondeau, Anne Canteaut et Pascale Charpin : Differential properties of $x \mapsto x^{2^t-1}$. *IEEE Transactions on Information Theory*, 57(12):8127–8137, 2011.

[25] Céline Blondeau et Kaisa Nyberg : New Links between Differential and Linear Cryptanalysis. *In* Johansson et Nguyen [115], pages 388–404.

[26] Céline Blondeau et Benoit Gérard : Differential Cryptanalysis of PUFFIN and PUFFIN2, 11 2011.

[27] Andrey Bogdanov, Huizheng Geng, Meiqin Wang, Long Wen et Baudoin Collard : Zero-correlation linear cryptanalysis with fft and improved attacks on iso standards camellia and clefia. *In* Lange *et al.* [141], pages 306–323.

[28] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin et C. Vikkelsoe : PRESENT : An Ultra-Lightweight Block Cipher. *In* Paillier et Verbauwhede [171], pages 450–466.

[29] Christina Boura : *Analyse de Fonctions de Hachage Cryptographiques*. Thèse de doctorat, UPMC, 2012.

[30] Christina Boura et Anne Canteaut : A zero-sum property for the KECCAK-f permutation with 18 rounds. *In ISIT* [1], pages 2488–2492.

[31] Christina Boura et Anne Canteaut : Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak-f and Hamsi-256. *In* Biryukov *et al.* [20], pages 1–17.

[32] Christina Boura et Anne Canteaut : On the Influence of the Algebraic Degree of F^{-1} on the Algebraic Degree of $G \circ F$. *IEEE Transactions on Information Theory*, 59(1):691–702, 2013.

[33] Christina Boura, Anne Canteaut et Christophe De Cannière : Higher-Order Differential Properties of Keccak and *Luffa*. *In* Joux [117], pages 252–269.

[34] Christina Boura, Marine Minier, María Naya-Plasencia et Valentin Suder : Improved Impossible Differential Attacks against Round-Reduced LBlock. *IACR Cryptology ePrint Archive*, 2014:279, 2014.

[35] Christina Boura, María Naya-Plasencia et Valentin Suder : Scrutinizing and Improving Impossible Differentials Attacks : Applications to CLEFIA, Camellia, LBlock and Simon, 2014. ASIACRYPT, à paraître.

[36] Christina Boura, María Naya-Plasencia et Valentin Suder : Scrutinizing and Improving Impossible Differentials Attacks : Applications to CLEFIA, Camellia, LBlock and Simon (Full version). *IACR Cryptology ePrint Archive*, 2014:699, 2014.

[37] Serdar Boztas et Hsiao-Feng Lu, éditeurs. *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, 17th International Symposium, AAECC-17, Bangalore, India, December 16-20, 2007, Proceedings*, volume 4851 de *Lecture Notes in Computer Science*. Springer, 2007.

[38] Carl Bracken, Eimear Byrne, Nadya Markin et Gary McGuire : New families of quadratic almost perfect nonlinear trinomials and multinomials. *Finite Fields and Their Applications*, 14(3):703–714, 2008.

[39] Carl Bracken, Eimear Byrne, Nadya Markin et Gary McGuire : A few more quadratic APN functions. *Cryptography and Communications*, 3(1):43–53, 2011.

[40] Carl Bracken, Eimear Byrne, Gary McGuire et Gabriele Nebe : On the equivalence of quadratic APN functions. *Des. Codes Cryptography*, 61(3):261–272, 2011.

[41] Carl Bracken et Gregor Leander : A Highly Nonlinear Differentially 4 Uniform Power Mapping That Permutes Fields of Even Degree. *Finite Fields and Their Applications*, 16(4):231–242, 2010.

[42] Carl Bracken, Chik How Tan et Yin Tan : Binomial differentially 4 uniform permutations with high nonlinearity. *Finite Fields and Their Applications*, 18(3):537–546, 2012.

[43] Ernest F. Brickell, éditeur. *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 de *Lecture Notes in Computer Science*. Springer, 1993.

[44] K. A. Browning, J. F. Dillon, R. E. Kibler et M. T. McQuistan : APN polynomials and related codes. *Special Volume of Journal of Combinatorics, Information and System Sciences, honoring the 75-th birthday of Prof. D.K.Ray-Chaudhuri*, 2008.

[45] K. A. Browning, J. F. Dillon, M. T. McQuistan et A. J. Wolfe : An APN Permutation in Dimension Six. *Finite Fields : Theory and Applications-Selected Papers from the 9th International Conference Finite Fields ans Applications. Contemporary Mathematics*, 518:33–42, 2010.

[46] Lilya Budaghyan, Claude Carlet et Gregor Leander : Two Classes of Quadratic APN Binomials Inequivalent to Power Functions. *IEEE Transactions on Information Theory*, 54(9):4218–4229, 2008.

[47] Lilya Budaghyan, Claude Carlet et Gregor Leander : Constructing new APN functions from known ones. *Finite Fields and Their Applications*, 15(2):150–159, 2009.

[48] Lilya Budaghyan, Claude Carlet et Alexander Pott : New classes of almost bent and almost perfect nonlinear polynomials. *IEEE Transactions on Information Theory*, 52(3):1141–1152, 2006.

[49] Anne Canteaut : Analyse et Conception de Chiffrements à Clef Secrète, 2006. Habilitation à diriger des recherches.

[50] Anne Canteaut, éditeur. *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 de *Lecture Notes in Computer Science*. Springer, 2012.

[51] Anne Canteaut, Claude Carlet, Pascale Charpin et Caroline Fontaine : Propagation Characteristics and Correlation-Immunity of Highly Nonlinear Boolean Functions. *In* Preneel [177], pages 507–522.

[52] Anne Canteaut, Claude Carlet, Pascale Charpin et Caroline Fontaine : On cryptographic properties of the cosets of R(1, m). *IEEE Transactions on Information Theory*, 47(4):1494–1513, 2001.

[53] Anne Canteaut et Pascale Charpin : Decomposing bent functions. *IEEE Transactions on Information Theory*, 49(8):2004–2019, 2003.

[54] Anne Canteaut, Pascale Charpin et Hans Dobbertin : Binary m-sequences with three-valued crosscorrelation : A proof of Welch's conjecture. *IEEE Transactions on Information Theory*, 46(1):4–9, 2000.

[55] Anne Canteaut, Pascale Charpin et Hans Dobbertin : Weight Divisibility of Cyclic Codes, Highly Nonlinear Functions on $\mathbb{F}_{2^m}$, and Crosscorrelation of Maximum-Length Sequences. *SIAM J. Discrete Math.*, 13(1):105–138, 2000.

[56] Anne Canteaut, Pascale Charpin et Gohar M. M. Kyureghyan : A new class of monomial bent functions. *Finite Fields and Their Applications*, 14(1):221–241, 2008.

[57] Anne Canteaut et María Naya-Plasencia : Structural weaknesses of permutations with low differential uniformity and generalized crooked functions. *Finite Fields : Theory and Applications-Selected Papers from the 9th International Conference Finite Fields ans Applications. Contemporary Mathematics*, 518:55–71, 2010.

[58] Claude Carlet : Boolean Functions for Cryptography and Error Correcting Codes. *In* Crama et Hammer [78], pages 257–397.

[59] Claude Carlet : Vectorial Boolean Functions for Cryptography. *In* Crama et Hammer [78], pages 398–469.

[60] Claude Carlet, Pascale Charpin et Victor Zinoviev : Codes, Bent Functions and Permutations Suitable For DES-like Cryptosystems. *Des. Codes Cryptography*, 15(2):125–156, 1998.

[61] Claude Carlet et Berk Sunar, éditeurs. *Arithmetic of Finite Fields, First International Workshop, WAIFI 2007, Madrid, Spain, June 21-22, 2007, Proceedings*, volume 4547 de *Lecture Notes in Computer Science*. Springer, 2007.

[62] Florian Caullery : APN functions of degree $4e$ with $e \equiv 3 (mod 4)$. *The International Workshop on Coding and Cryptography, WCC 2013, Bergen, Norway*, pages 39–46.

[63] Florent Chabaud et Serge Vaudenay : Links between differential and linear cryptoanalysis. *In* Santis [182], pages 356–365.

[64] Pascale Charpin : Normal Boolean functions. *J. Complexity*, 20(2-3):245–265, 2004.

[65] Pascale Charpin et Gohar M. M. Kyureghyan : On a Class of Permutation Polynomials over $\mathbb{F}_{2^m}$. *In* Golomb *et al.* [103], pages 368–376.

[66] Pascale Charpin et Gohar M. M. Kyureghyan : When does $G(x) + \gamma Tr(H(x))$ permute $\mathbb{F}_{p^n}$? *Finite Fields and Their Applications*, 15(5):615–632, 2009.

[67] Pascale Charpin et Gohar M. M. Kyureghyan : Monomial functions with linear structure and permutation polynomials. *Finite Fields : Theory and Applications-Selected Papers from the 9th International Conference Finite Fields ans Applications. Contemporary Mathematics*, 518:99–111, 2010.

[68] Pascale Charpin et Gohar M. M. Kyureghyan : A note on verifying the APN property. *IACR Cryptology ePrint Archive*, 2013:475, 2013.

[69] Pascale Charpin, Gohar M. M. Kyureghyan et Valentin Suder : Sparse Permutations with Low Differential Uniformity. *Finite Fields and Their Applications*, 28:214–243, 2014.

[70] Pascale Charpin, Enes Pasalic et Cédric Tavernier : On bent and semi-bent quadratic Boolean functions. *IEEE Transactions on Information Theory*, 51(12):4286–4298, 2005.

[71] Pascale Charpin, Alexander Pott et Arne Winterhof, éditeurs. *Finite Fields and their Applications, Character Sums and Polynomials*, volume 11 de *RADON Series on Computanional and Applied Mathematics*. De Gruyter, 2013.

[72] Pascale Charpin et Sumanta Sarkar : Polynomials with linear structure and Maiorana-McFarland construction. *In ISIT* [1], pages 1138–1142.

[73] Pascale Charpin et Sumanta Sarkar : Polynomials With Linear Structure and Maiorana-McFarland Construction. *IEEE Transactions on Information Theory*, 57(6):3796–3804, 2011.

[74] David Chaum et Wyn L. Price, éditeurs. *Advances in Cryptology - EUROCRYPT '87, Workshop on the Theory and Application of of Cryptographic Techniques, Amsterdam, The Netherlands, April 13-15, 1987, Proceedings*, volume 304 de *Lecture Notes in Computer Science*. Springer, 1988.

[75] Gérard D. Cohen, Teo Mora et Oscar Moreno, éditeurs. *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, 10th International Symposium, AAECC-10, San Juan de Puerto Rico, Puerto Rico, May 10-14, 1993, Proceedings*, volume 673 de *Lecture Notes in Computer Science*. Springer, 1993.

[76] Stephen D. Cohen : Finite Field Elements with Specified Order and Traces. *Des. Codes Cryptography*, 36(3):331–340, 2005.

[77] Nicolas Courtois et Josef Pieprzyk : Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. *In* Zheng [220], pages 267–287.

[78] Yves Crama et Peter L. Hammer, éditeurs. *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, volume 134 de *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2010.

[79] Joan Daemen et Vincent Rijmen : Rijndael for AES. *In AES Candidate Conference*, pages 343–348, 2000.

[80] Joan Daemen et Vincent Rijmen : *The Design of Rijndael : AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[81] Donald W. Davies, éditeur. *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 de *Lecture Notes in Computer Science*. Springer, 1991.

[82] John F. Dillon : *Elementary Hadamard Difference sets*. Thèse de doctorat, University of Maryland, 1974.

[83] John F. Dillon : Geometry, codes and difference sets : exceptional connections. *Codes and Designs, Colombus, OH*, pages 73–85, 2000.

[84] John F. Dillon et Hans Dobbertin : New cyclic difference sets with Singer parameters. *Finite Fields and Their Applications*, 10(3):342–389, 2004.

[85] Itai Dinur et Adi Shamir : Cube Attacks on Tweakable Black Box Polynomials. *In* Joux [116], pages 278–299.

[86] Hans Dobbertin : Almost Perfect Nonlinear Power Functions on $GF(2^n)$: The Niho Case . *Information and Computation*, 151(1):57 – 72, 1999.

[87] Hans Dobbertin : Almost Perfect Nonlinear Power Functions on $GF(2^n)$: The Welch Case. *IEEE Transactions on Information Theory*, 45(4):1271–1275, 1999.

[88] Hans Dobbertin : Another Proof of Kasami's Theorem. *Des. Codes Cryptography*, 17(1-3):177–180, 1999.

[89] Hans Dobbertin : Almost Perfect Nonlinear Power Functions on $GF(2^n)$: a new class for n divisible by 5. *Finite Fields and Applications Fq*5, pages 113 – 121, 2000.

[90] Hans Dobbertin, Gregor Leander, Anne Canteaut, Claude Carlet, Patrick Felke et Philippe Gaborit : Construction of bent functions via Niho power functions. *J. Comb. Theory, Ser. A*, 113(5):779–798, 2006.

[91] Xiang dong Hou : Affinity of permutations of F_{2^n}. *Discrete Applied Mathematics*, 154(2):313–325, 2006.

[92] Orr Dunkelman, Gautham Sekar et Bart Preneel : Improved Meet-in-the-Middle Attacks on Reduced-Round DES. *In* Srinathan *et al.* [186], pages 86–100.

[93] Yves Edel, Gohar M. M. Kyureghyan et Alexander Pott : A new APN function which is not equivalent to a power mapping. *IEEE Transactions on Information Theory*, 52(2):744–747, 2006.

[94] Yves Edel et Alexander Pott : A new almost perfect nonlinear function which is not quadratic. *Advances in Mathematics of Communications*, 3(1):59–81, 2009.

[95] Yves Edel et Alexander Pott : On Designs and Multiplier Groups Constructed from Almost Perfect Nonlinear Functions. *In* Parker [172], pages 383–401.

[96] Jan-Hendrik Evertse : Linear structures in blockciphers. *In* Chaum et Price [74], pages 249–266.

[97] Joan Feigenbaum, éditeur. *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, volume 576 de *Lecture Notes in Computer Science*. Springer, 1992.

[98] Caroline Fontaine : *Contribution à la recherche de fonctions booléennes hautement non linéaires, et au marquage d'images en vue de la protection des droits d'auteur.* Thèse de doctorat, UPMC, 1998.

[99] Henri Gilbert et Thomas Peyrin : Super-Sbox Cryptanalysis : Improved Attacks for AES-Like Permutations. *In* Hong et Iwata [111], pages 365–383.

[100] Robert Gold : Maximal recursive sequences with 3-valued recursive cross-correlation functions (Corresp.). *IEEE Transactions on Information Theory*, 14(1):154–156, Jan 1968.

[101] Jovan Dj. Golic : On the Security of Nonlinear Filter Generators. *In* Gollmann [102], pages 173–188.

[102] Dieter Gollmann, éditeur. *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 21-23, 1996, Proceedings*, volume 1039 de *Lecture Notes in Computer Science*. Springer, 1996.

[103] Solomon W. Golomb, Matthew G. Parker, Alexander Pott et Arne Winterhof, éditeurs. *Sequences and Their Applications - SETA 2008, 5th International Conference, Lexington, KY, USA, September 14-18, 2008, Proceedings*, volume 5203 de *Lecture Notes in Computer Science*. Springer, 2008.

[104] Faruk Göloğlu et Gary McGuire : When is $x^{-1} + L(x)$ a permutation in odd characteristic ? *In Workshop on Coding and Cryptography, WCC 2013*, April 2006. Bergen, Norway.

[105] Guang Gong et Kishan Chand Gupta, éditeurs. *Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings*, volume 6498 de *Lecture Notes in Computer Science*. Springer, 2010.

[106] Paul R. Halmos : *Finite-Dimensional Vector Spaces*. The Unversity Series in Undergraduate Mathematics. D. Van Nostrand Company, second édition, 1958.

[107] Tor Helleseth, éditeur. *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 de *Lecture Notes in Computer Science*. Springer, 1994.

[108] Tor Helleseth et Jonathan Jedwab, éditeurs. *Sequences and Their Applications - SETA 2012 - 7th International Conference, Waterloo, ON, Canada, June 4-8, 2012. Proceedings*, volume 7280 de *Lecture Notes in Computer Science*. Springer, 2012.

[109] Fernando Hernando et Gary McGuire : Proof of a conjecture on the sequence of exceptional numbers, classifying cyclic codes and APN functions. *Journal of Algebra*, 343(1):78–92, Oct 2011.

[110] Doreen Hertel et Alexander Pott : Two results on maximum nonlinear functions. *Des. Codes Cryptography*, 47(1-3):225–235, 2008.

[111] Seokhie Hong et Tetsu Iwata, éditeurs. *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 de *Lecture Notes in Computer Science*. Springer, 2010.

[112] Heeralal Janwa, Gary Mcguire et Richard M. Wilson : Double-Error-Correcting Cyclic Codes and Absolutely Irreducible Polynomials over GF(2) . *Journal of Algebra*, 178(2):665 – 676, 1995.

[113] Heeralal Janwa et Richard M. Wilson : Hyperplane Sections of Fermat Varieties in P^3 in Char.2 and Some Applications to Cyclic Codes. *In* Cohen *et al.* [75], pages 180–194.

[114] Thomas Johansson et Subhamoy Maitra, éditeurs. *Progress in Cryptology - INDOCRYPT 2003, 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003, Proceedings*, volume 2904 de *Lecture Notes in Computer Science*. Springer, 2003.

[115] Thomas Johansson et Phong Q. Nguyen, éditeurs. *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 de *Lecture Notes in Computer Science*. Springer, 2013.

[116] Antoine Joux, éditeur. *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, volume 5479 de *Lecture Notes in Computer Science*. Springer, 2009.

[117] Antoine Joux, éditeur. *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 de *Lecture Notes in Computer Science*. Springer, 2011.

[118] Michael J. Jacobson Jr., Vincent Rijmen et Reihaneh Safavi-Naini, éditeurs. *Selected Areas in Cryptography, 16th Annual International Workshop, SAC 2009, Calgary, Alberta, Canada, August 13-14, 2009, Revised Selected Papers*, volume 5867 de *Lecture Notes in Computer Science*. Springer, 2009.

[119] Ferhat Karakoç, Hüseyin Demirci et A. Emre Harmanci : Impossible Differential Cryptanalysis of Reduced-Round LBlock. *In* Askoxylakis *et al.* [8], pages 179–188.

[120] Tadao Kasami : The Weight Enumerators for Several Clauses of Subcodes of the 2nd Order Binary Reed-Muller Codes. *Information and Control*, 18(4):369–394, 1971.

[121] Tadao Kasami et Nobuki Tokura : On the weight structure of Reed-Muller codes. *IEEE Transactions on Information Theory*, 16(6):752–759, 1970.

[122] Jongsung Kim, Seokhie Hong et Jongin Lim : Impossible differential cryptanalysis using matrix method. *Discrete Mathematics*, 310(5):988–1002, 2010.

[123] Jongsung Kim, Seokhie Hong, Jaechul Sung, Changhoon Lee et Sangjin Lee : Impossible Differential Cryptanalysis for Block Cipher Structures. *In* Johansson et Maitra [114], pages 82–96.

[124] Kwangjo Kim et Tsutomu Matsumoto, éditeurs. *Advances in Cryptology - ASIACRYPT '96, International Conference on the Theory and Applications of Cryptology and Information Security, Kyongju, Korea, November 3-7, 1996, Proceedings*, volume 1163 de *Lecture Notes in Computer Science*. Springer, 1996.

[125] Lars R. Knudsen : Truncated and Higher Order Differentials. *In* Preneel [176], pages 196–211.

[126] Lars R. Knudsen : DEAL – A 128-bit cipher, 1998. Technical Report (AES submission), Department of Informatics, University of Bergen, Norway.

[127] Lars R. Knudsen et Matthew J.B. Robshaw : *The Block Cipher Companion*. Information Security and Cryptography. Springer, 2011.

[128] Gohar M. M. Kyureghyan : Crooked maps in $\mathbb{F}_{2^n}$. *Finite Fields and Their Applications*, 13(3):713–726, 2007.

[129] Gohar M. M. Kyureghyan : The only crooked power functions are $x^{2^k+2^l}$. *Eur. J. Comb.*, 28(4):1345–1350, 2007.

[130] Gohar M. M. Kyureghyan : Constructing permutations of finite fields via linear translators. *J. Comb. Theory, Ser. A*, 118(3):1052–1061, 2011.

[131] Gohar M. M. Kyureghyan et Alexander Pott : Some Theorems on Planar Mappings. *In* von zur Gathen *et al.* [204], pages 117–122.

[132] Gohar M. M. Kyureghyan et Valentin Suder : On inverses of APN exponents. *In ISIT* [2], pages 1207–1211.

[133] Gohar M. M. Kyureghyan et Valentin Suder : On Inversion in $\mathbb{Z}_{2^n-1}$. *Finite Fields and Their Applications*, 25:234–254, 2014.

[134] Xuejia Lai : Higher Order Derivatives and Differential Cryptanalysis. *In Symposium on Communication, Coding and Cryptography, in honor of James L. Massey on the occasion of his 60'th birthday*, Monte-Verita, Ascona, Switzerland, February $10 - 13$ 1994.

[135] Xuejia Lai et James L. Massey : Markov Ciphers and Differential Cryptanalysis. *In* Davies [81], pages 17–38.

[136] Yann Laigle-Chapuy : A Note on a Class of Quadratic Permutations over $\mathbb{F}_{2^n}$. *In* Boztas et Lu [37], pages 130–137.

[137] Yann Laigle-Chapuy : Permutation polynomials and applications to coding theory. *Finite Fields and Their Applications*, 13(1):58–70, 2007.

[138] Yann Laigle-Chapuy : *Polynômes de Permutation et Application en cryptographie, Cryptanalyse de Registres Combinés*. Thèse de doctorat, UPMC, 2009.

[139] Kwok-Yan Lam, Eiji Okamoto et Chaoping Xing, éditeurs. *Advances in Cryptology - ASIACRYPT '99, International Conference on the Theory and Applications of Cryptology and Information Security, Singapore, November 14-18, 1999, Proceedings*, volume 1716 de *Lecture Notes in Computer Science*. Springer, 1999.

[140] Serge Lang : *Algebra*. Addison Wesley, third édition, 1993.

[141] Tanja Lange, Kristin Lauter et Petr Lisonek, éditeurs. *Selected Areas in Cryptography - SAC 2013 - 20th International Conference, Burnaby, BC, Canada, August 14-16, 2013, Revised Selected Papers*, volume 8282 de *Lecture Notes in Computer Science*. Springer, 2014.

[142] Gregor Leander et Axel Poschmann : On the classification of 4 bit s-boxes. *In* Carlet et Sunar [61], pages 159–176.

[143] Chunlei Li, Nian Li, Tor Helleseth et Cunsheng Ding : On the weight distributions of several classes of cyclic codes from APN monomials. *CoRR*, abs/1308.5885, 2013.

[144] Yongqiang Li et Mingsheng Wang : On EA-equivalence of certain permutations to power mappings. *Des. Codes Cryptography*, 58(3):259–269, 2011.

[145] Yongqiang Li et Mingsheng Wang : Permutation polynomials EA-equivalent to the inverse function over $GF(2^n)$. *Cryptography and Communications*, 3(3):175–186, 2011.

[146] Rudolf Lidl et Harald Niederreiter : *Finite Fields*, volume 20 de *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1997.

[147] Ya Liu, Leibo Li, Dawu Gu, Xiaoyun Wang, Zhiqiang Liu, Jiazhe Chen et Wei Li : New Observations on Impossible Differential Cryptanalysis of Reduced-Round Camellia. *In* Canteaut [50], pages 90–109.

[148] Javier Lopez et Gene Tsudik, éditeurs. *Applied Cryptography and Network Security - 9th International Conference, ACNS 2011, Nerja, Spain, June 7-10, 2011. Proceedings*, volume 6715 de *Lecture Notes in Computer Science*, 2011.

[149] Jiqiang Lu, Jongsung Kim, Nathan Keller et Orr Dunkelman : Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1. *In* Malkin [153], pages 370–386.

[150] Florence J. MacWilliams et Neil J. A. Sloane : *The Theory of Error-Correcting Codes*, volume 16 de *North-Holland Mathematical Library*. Elsevier Science Publishers, 1977.

[151] Hamid Mala, Mohammad Dakhilalian et Mohsen Shakiba : Impossible Differential Attacks on 13-Round CLEFIA-128. *J. Comput. Sci. Technol.*, 26(4):744–750, 2011.

[152] Hamid Mala, Mohsen Shakiba, Mohammad Dakhilalian et Ghadamali Bagherikaram : New Results on Impossible Differential Cryptanalysis of Reduced-Round Camellia-128. *In* Jr. *et al.* [118], pages 281–294.

[153] Tal Malkin, éditeur. *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings*, volume 4964 de *Lecture Notes in Computer Science*. Springer, 2008.

[154] Ariane M. Masuda et Michael E. Zieve : Permutation binomials over finite fields. *Trans. Amer. Math. Soc*, 2007.

[155] Mitsuru Matsui : Linear Cryptoanalysis Method for DES Cipher. *In* Helleseth [107], pages 386–397.

[156] Robert J. McEliece : *Finite Fields for Computer Scientists and Engineers*. Kluwer Academic Publishers, 1987.

[157] Marine Minier et María Naya-Plasencia : A related key impossible differential attack against 22 rounds of the lightweight block cipher LBlock. *Inf. Process. Lett.*, 112(16):624–629, 2012.

[158] Marine Minier et María Naya-Plasencia : Communication privée, Mai 2013.

[159] Gary Mullen : Permutation polynomials over finite fields. *Finite Fields, Coding Theory, and Advances in Commu- nication and Computing, Las Vegas, NY, Lecture Notes in Pure and Applied Math.*, 141:131–151.

[160] Gary L. Mullen : Permutation polynomials and nonsingular feedback shift registers over finite fields. *IEEE Transactions on Information Theory*, 35(4):900–902, 1989.

[161] Gary L. Mullen et Daniel Panario, éditeurs. *Handbook of Finite Fields*. Discrete Mathematics and its Applications. CRC Press, 2013.

[162] Amela Muratovic-Ribic : A note on the coefficients of inverse polynomials. *Finite Fields and Their Applications*, 13(4):977–980, 2007.

[163] Yassir Nawaz, Guang Gong et Kishan Chand Gupta : Upper Bounds on Algebraic Immunity of Boolean Power Functions. *In* Robshaw [178], pages 375–389.

[164] Yassir Nawaz, Kishan Chand Gupta et Guang Gong : Algebraic immunity of S-boxes based on power mappings : analysis and construction. *IEEE Transactions on Information Theory*, 55(9):4263–4273, 2009.

[165] Kaisa Nyberg : On the Construction of Highly Nonlinear Permutations. *In* Rueppel [181], pages 92–98.

[166] Kaisa Nyberg : Differentially Uniform Mappings for Cryptography. *In* Helleseth [107], pages 55–64.

[167] Kaisa Nyberg : S-boxes and Round Functions with Controllable Linearity and Differential Uniformity. *In* Preneel [176], pages 111–130.

[168] Kaisa Nyberg : Generalized Feistel Networks. *In* Kim et Matsumoto [124], pages 91–104.

[169] Kaisa Nyberg, éditeur. *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 de *Lecture Notes in Computer Science*. Springer, 2008.

[170] Kaisa Nyberg et Lars R. Knudsen : Provable Security Against Differential Cryptanalysis. *In* Brickell [43], pages 566–574.

[171] Pascal Paillier et Ingrid Verbauwhede, éditeurs. *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 de *Lecture Notes in Computer Science*. Springer, 2007.

[172] Matthew G. Parker, éditeur. *Cryptography and Coding, 12th IMA International Conference, Cryptography and Coding 2009, Cirencester, UK, December 15-17, 2009. Proceedings*, volume 5921 de *Lecture Notes in Computer Science*. Springer, 2009.

[173] Enes Pasalic et Pascale Charpin : Some results concerning cryptographically significant mappings over $GF(2^n)$. *Des. Codes Cryptography*, 57(3):257–269, 2010.

[174] Goutam Paul et Serge Vaudenay, éditeurs. *Progress in Cryptology - INDOCRYPT 2013 - 14th International Conference on Cryptology in India, Mumbai, India, December 7-10, 2013. Proceedings*, volume 8250 de *Lecture Notes in Computer Science*. Springer, 2013.

[175] Marius Portmann et Marc Rennhard : Almost Perfect Nonlinear Permutations. Semester project, Eidgenössische Technische Hochschule Zürich, 1997.

[176] Bart Preneel, éditeur. *Fast Software Encryption : Second International Workshop. Leuven, Belgium, 14-16 December 1994, Proceedings*, volume 1008 de *Lecture Notes in Computer Science*. Springer, 1995.

[177] Bart Preneel, éditeur. *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 de *Lecture Notes in Computer Science*. Springer, 2000.

[178] Matthew J. B. Robshaw, éditeur. *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 de *Lecture Notes in Computer Science*. Springer, 2006.

[179] François Rodier : Functions of degree $4e$ that are not APN infinitely often. *Cryptography and Communications*, 3(4):227–240, 2011.

[180] O. S. Rothaus : On "bent" functions. *J. Comb. Theory, Ser. A*, 20(3):300–305, 1976.

[181] Rainer A. Rueppel, éditeur. *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, volume 658 de *Lecture Notes in Computer Science*. Springer, 1993.

[182] Alfredo De Santis, éditeur. *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, volume 950 de *Lecture Notes in Computer Science*. Springer, 1995.

[183] Claude Shannon : Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, 1949.

[184] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai et Tetsu Iwata : The 128-Bit Blockcipher CLEFIA (Extended Abstract). *In* Biryukov [19], pages 181–195.

[185] Igor Shparlinski : A deterministic test for permutation polynomials. *Computational Complexity*, 2:129–132, 1992.

[186] K. Srinathan, C. Pandu Rangan et Moti Yung, éditeurs. *Progress in Cryptology - INDOCRYPT 2007, 8th International Conference on Cryptology in India, Chennai,*

India, December 9-13, 2007, Proceedings, volume 4859 de *Lecture Notes in Computer Science*. Springer, 2007.

[187] Jacques Stern, éditeur. *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 de *Lecture Notes in Computer Science*. Springer, 1999.

[188] Douglas R. Stinson et Stafford E. Tavares, éditeurs. *Selected Areas in Cryptography, 7th Annual International Workshop, SAC 2000, Waterloo, Ontario, Canada, August 14-15, 2000, Proceedings*, volume 2012 de *Lecture Notes in Computer Science*. Springer, 2001.

[189] Valentin Suder et Gohar Kyureghyan : On Inversion in Z_{2^n-1}. *In The 11th International Conference on Finite Fields and their Applications*, 2013.

[190] Yin Tan, Longjiang Qu, Chik How Tan et Chao Li : New Families of Differentially 4-Uniform Permutations over $\mathbb{F}_{2^{2k}}$. *In* Helleseth et Jedwab [108], pages 25–39.

[191] Deng Tang, Claude Carlet et Xiaohu Tang : Differentially 4-Uniform Bijections by Permuting the Inverse Function. *IACR Cryptology ePrint Archive*, 2013:639, 2013.

[192] Xuehai Tang, Bing Sun, Ruilin Li et Chao Li : Impossible differential cryptanalysis of 13-round CLEFIA-128. pages 1191–1196, 2011.

[193] Anne Tardy-Corfdir et Henri Gilbert : A Known Plaintext Attack of FEAL-4 and FEAL-6. *In* Feigenbaum [97], pages 172–181.

[194] Sony Corporation CLEFIA Design Team : Commentaires sur la cryptanalyse différentielle impossible de CLEFIA présentée à Inscrypt 2008, 2009. 8 Janvier.

[195] Cihangir Tezcan : The Improbable Differential Attack : Cryptanalysis of Reduced Round CLEFIA. *In* Gong et Gupta [105], pages 197–209.

[196] Yukiyasu Tsunoo, Etsuko Tsujihara, Maki Shigeri, Teruo Saito, Tomoyasu Suzaki et Hiroyasu Kubo : Impossible Differential Cryptanalysis of CLEFIA. *In* Nyberg [169], pages 398–411.

[197] Yukiyasu Tsunoo, Etsuko Tsujihara, Maki Shigeri, Tomoyasu Suzaki et Tsutomu Kawabata : Cryptanalysis of CLEFIA using multiple impossible differentials. *In Information Theory and Its Applications. ISITA 2008*, pages 1–6, 2008.

[198] Aleksandr Tuxanidy et Qiang Wang : On the inverses of some classes of permutations of finite fields. *Finite Fields and Their Applications*, 28(0):244 – 281, 2014.

[199] Edwin R. van Dam et Dmitry Fon-Der-Flaass : Codes, graphs, and schemes from nonlinear functions. pages 85–98, 2003.

[200] Serge Vaudenay : On the Lai-Massey Scheme. *In* Lam *et al.* [139], pages 8–19.

[201] Marion Videau : *Critères de Sécurité des Algorithmes de Chiffrement à Clé Secrète*. Thèse de doctorat, UPMC, 2005.

[202] Joachim von zur Gathen : Tests for permutation polynomials. *SIAM J. Comput.*, 20(3):591–602, 1991.

[203] Joachim von zur Gathen et Jürgen Gerhard : *Modern Computer Algebra*. Cambridge University Press, 1999.

[204] Joachim von zur Gathen, José Luis Imaña et Çetin Kaya Koç, éditeurs. *Arithmetic of Finite Fields, 2nd International Workshop, WAIFI 2008, Siena, Italy, July 6-9, 2008, Proceedings*, volume 5130 de *Lecture Notes in Computer Science*. Springer, 2008.

[205] Zhe-Xian Wan : *Lectures on Finite Fields and Galois Rings*. World Scientific, 2003.

[206] Qiang Wang : On inverse permutation polynomials. *Finite Fields and Their Applications*, 15(2):207–213, 2009.

[207] Guobiao Weng, Yin Tan et Guang Gong : On quadratic almost perfect nonlinear functions and their related algebraic object. *The International Workshop on Coding and Cryptography, WCC 2013, Bergen, Norway*, pages 57–68.

[208] Baofeng Wu : The compositional inverse of a class of linearized permutation polynomials over $\mathbb{F}_{2^n}$, n odd. *Finite Fields and Their Applications*, 29(0):34 – 48, 2014.

[209] Gaofei Wu, Nian Li, Tor Helleseth et Yuqing Zhang : More Classes of Complete Permutation Polynomials over F_q. *CoRR*, abs/1312.4716, 2013.

[210] Wenling Wu et Lei Zhang : LBlock : A Lightweight Block Cipher. *In* Lopez et Tsudik [148], pages 327–344.

[211] Wenling Wu, Lei Zhang et Wentao Zhang : Improved Impossible Differential Cryptanalysis of Reduced-Round Camellia. *In* Avanzi *et al.* [11], pages 442–456.

[212] Wenling Wu, Wentao Zhang et Dengguo Feng : Impossible Differential Cryptanalysis of Reduced-Round ARIA and Camellia. *J. Comput. Sci. Technol.*, 22(3):449–456, 2007.

[213] Satoshi Yoshiara : Equivalences of quadratic APN functions. *Journal of Algebraic Combinatorics*, 35(3):461–475, 2012.

[214] Yuyin Yu, Mingsheng Wang et Yongqiang Li : A matrix approach for constructing quadratic APN functions. *The International Workshop on Coding and Cryptography, WCC 2013, Bergen, Norway*, pages 47–56.

[215] Pingzhi Yuan et Cunsheng Ding : Permutation polynomials of the form $L(x) + S_{2k}^a + S_{2k}^b$ over $\mathbb{F}_{q^{3k}}$. *Finite Fields and Their Applications*, 29(0):106 – 117, 2014.

[216] Zheng Yuan, Xian Li et Haixia Liu : Impossible Differential-Linear Cryptanalysis of Full-Round CLEFIA-128. *IACR Cryptology ePrint Archive*, 2013:301, 2013.

[217] Moti Yung, Peng Liu et Dongdai Lin, éditeurs. *Information Security and Cryptology, 4th International Conference, Inscrypt 2008, Beijing, China, December 14-17, 2008, Revised Selected Papers*, volume 5487 de *Lecture Notes in Computer Science*. Springer, 2009.

[218] Zhengbang Zha, Gohar M. M. Kyureghyan et Xueli Wang : A New Family of Perfect Nonlinear Binomials. *IACR Cryptology ePrint Archive*, 2008:196, 2008.

[219] Wenying Zhang et Jing Han : Impossible Differential Analysis of Reduced Round CLEFIA. *In* Yung *et al.* [217], pages 181–191.

[220] Yuliang Zheng, éditeur. *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*, volume 2501 de *Lecture Notes in Computer Science*. Springer, 2002.