THÈSE DE DOCTORAT

de l'Université de recherche Paris Sciences et Lettres
PSL Research University

Préparée à l'École normale supérieure

Hybrid fully homomorphic framework

École doctorale n°386
Sciences Mathématiques de Paris Centre

Spécialité Informatique

Soutenue par **Pierrick MÉAUX**
le 8 décembre 2017

Dirigée par
Vadim LYUBASHEVSKY
et **David POINTCHEVAL**

COMPOSITION DU JURY

MME. CANTEAUT Anne
Inria de Paris
Rapporteur

M. CARLET Claude
Université Paris VIII
Président du jury

M. LYUBASHEVSKY Vadim
IBM Zürich
Co-directeur de thèse

MME. MESNAGER Sihem
Université Paris VIII
Examinateur

M. POINTCHEVAL David
CNRS, École normale supérieure
Directeur de thèse

M. SIRDEY Renaud
Commissariat à l'énergie atomique
Examinateur

M. STANDAERT François-Xavier
Université catholique de Louvain
Examinateur

M. STEHLÉ Damien
École normale supérieure de Lyon
Rapporteur

Hybrid Fully Homomorphic Framework

Pierrick Méaux

Thèse de doctorat dirigée par
Vadim Lyubashevsky et David Pointcheval

Abstract

Fully homomorphic encryption, firstly built in 2009, is a very powerful kind of encryption, allowing to compute any function on encrypted data, and to get an encrypted version of the result. Such encryption enables to securely delegate data to a cloud, ask for computations, recover the result, while keeping private the data during the whole process. However, today's inefficiency of fully homomorphic encryption, and its inadequateness to the outsourcing computation context, makes its use alone insufficient for this application. Both of these issues can be circumvented, using fully homomorphic encryption in a larger framework, by combining it with a symmetric encryption scheme. This combination gives a hybrid fully homomorphic framework, designed towards efficient outsourcing computation, providing both security and privacy.

In this thesis, we contribute to the study of hybrid fully homomorphic framework, through the analysis, and the design of symmetric primitives making efficient this hybrid construction. Through the examination of fully homomorphic encryption schemes, we develop tools to efficiently use the homomorphic properties in a more complex framework. By investigating various symmetric encryption schemes, and building blocks up to the circuit level, we determine good candidates for a hybrid context. Through evaluating the security of constructions optimizing the homomorphic evaluation, we contribute to a wide study within the cryptographic Boolean functions area.

More particularly, we introduce a new family of symmetric encryption schemes, with a new design, adapted to the hybrid fully homomorphic framework. We then investigate its behavior relatively to homomorphic evaluation, and we address the security of such design. Finally, particularities of this family of ciphers motivate specific cryptanalyses, therefore we develop and analyze new cryptographic Boolean criteria.

Résumé

Le chiffrement complètement homomorphe est une classe de chiffrement permettant de calculer n'importe quelle fonction sur des données chiffrées et de produire une version chiffrée du résultat. Il permet de déléguer des données à un cloud de façon sécurisée, faire effectuer des calculs, tout en gardant le caractère privé de ces données. Cependant, l'innéficacité actuelle des schémas de chiffrement complètement homomorphes, et leur inadéquation au contexte de délégation de calculs, rend son usage seul insuffisant pour cette application. Ces deux problèmes peuvent être résolus, en utilisant ce chiffrement dans un cadre plus large, en le combinant avec un schéma de chiffrement symétrique. Cette combinaison donne naissance au chiffrement complètement homomorphe hybride, conçu dans le but d'une délégation de calculs efficace, garantissant des notions de sécurité et de vie privée.

Dans cette thèse, nous étudions le chiffrement complètement homomorphe hybride et ses composantes, à travers la conception de primitives cryptographiques symétriques rendant efficace cette construction hybride. En examinant les schémas de chiffrement complètement homomorphes, nous developpons des outils pour utiliser efficacement leurs propriétés homomorphiques dans un cadre plus complexe. En analysant différents schémas symétriques, et leurs composantes, nous déterminons de bons candidats pour le contexte hybride. En étudiant la sécurité des constructions optimisant l'évaluation homomorphique, nous contribuons au domaine des fonctions booléennes utilisées en cryptologie.

Plus particulièrement, nous introduisons une nouvelle famille de schémas de chiffrement symétriques, avec une nouvelle construction, adaptée au contexte hybride. Ensuite, nous nous intéressons à son comportement homomorphique, et nous étudions la sécurité de cette construction. Finalement, les particularités de cette famille de schémas de chiffrement motivant des cryptanalyses spécifiques, nous développons et analysons de nouveaux critères cryptographiques booléens.

Acknowledgments

Différentes personnes ont contribué au bon déroulement de ma thèse, que ce soit par ce qu'elles m'ont appris, par le soutient qu'elles m'ont apporté, ou par les bons moments qu'elles m'ont fait passer, au cours de ces trois dernières années. Je remercie toutes ces personnes et je leur en suis reconnaissant. Ces remerciements sont la dernière partie de ce manuscrit que j'écris (après une rédaction un peu longue, voire douloureuse) ; je m'excuse par avance pour tous ceux que j'oublierai de mentionner, ou pour qui je ne trouverai pas les mots pour exprimer ma gratitude.

Je remercie tout d'abord mes deux directeurs de thèse, Vadim Lyubashevsky et David Pointcheval, pour avoir accepté d'encadrer ma thèse, et pour m'avoir permis de faire ma thèse au sein de l'équipe crypto/cascade de l'ENS (INRIA, CNRS, PSL, choisissez votre affiliation préférée). I thank Vadim for sharing his insight on the important problems in lattice-based cryptography (and on research in general), for inviting me at IBM Zurich last year, and for inciting me to visit François-Xavier Standaert at UCL. Je remercie David pour son extrême efficacité, pour m'avoir toujours apporté rapidement des solutions aux problèmes administratifs, d'enseignement, de financement de missions, ou de vie du labo, ainsi que pour avoir relu mon manuscrit. Je le remercie aussi pour toujours contribuer à faire de l'équipe crypto ce qu'elle est, entre-autres un bon environnement pour faire sa thèse.

Je tiens à remercier Anne Canteaut et Damien Stehlé pour avoir accepté d'être mes rapporteurs. Ce sont deux chercheurs que j'admire pour leurs résultats, la qualité de leurs travaux, et leur engagement pédagogique. Je suis reconnaissant pour leurs nombreux commentaires qui m'ont permis d'améliorer ce manuscrit.

Je remercie mes coauteurs avec qui j'ai beaucoup appris à travers différentes collaborations : Claude Carlet, Romain Gay, Anthony Journault, Yann Rotella, François-Xavier Standaert, et Hoeteck Wee. Je remercie tout particulièrement François-Xavier et Claude. Je remercie François-Xavier pour m'avoir accueilli chaleureusement au sein de l'équipe à LLN, à de nombreuses reprises. Nos discutions de recherches, confrontant parfois des notions diamétralement opposées (du hardware au FHE), m'ont beaucoup apporté. Je lui suis reconnaissant pour avoir accepté d'être dans mon jury, et de faire un post-doc avec lui. Je remercie Claude pour les nombreux échanges que l'on a pu avoir au cours de nos collaborations. Ses connaissances en fonctions booléennes, (et l'élégance de certaines de ses démonstrations), ont largement influencé mon attrait récent pour ce domaine. Je lui suis reconnaissant pour avoir accepté d'être dans mon jury.

Je remercie aussi Sihem Mesnager et Renaud Sirdey pour avoir accepté de faire parti de mon jury. J'ai trouvé très intéressantes les différentes discussions que j'ai eu l'occasion d'avoir avec chacun de ces deux chercheurs. Et je les remercie pour l'intérêt qu'ils ont manifesté concernant mes travaux.

Au cours de cette thèse j'ai pu rencontrer d'autres chercheurs lors de conférences ou réunions diverses. Je les remercie pour des échanges intéressants sur la cryptographie ou plus généraux, ou pour m'avoir invité à présenter mes travaux. Je pense plus particulièrement à Lilya Budaghyan, Léo Ducas, Sébastien Duval, Jean-Pierre Fiori, Malika Izabachène, Virginie

Lallemand, Adeline Langlois, Fabrice Mouhartem, Pascal Pailler, María Naya-Plasencia, et Elizabeth Quaglia.

Ces trois ans de thèse ont plus souvent été marqués par beaucoup de temps à chercher et peu de temps à trouver. Je remercie toute l'équipe de l'ENS pour m'avoir permis de relativiser sur les hauts et les bas de la recherche, ou de sortir de la solitude le temps d'une (parfois longue) pause café. Je remercie Damien Vergnaud pour m'avoir fait confiance pour être son chargé de TD. J'ai beaucoup apprécié ces deux années de LFCC et plus généralement d'avoir pu m'impliquer dans l'enseignement à l'ENS. Je tiens aussi à remercier Michel Abdalla, pour ses conseils avisés et bienveillants sur la vie d'un jeune chercheur. Je lui suis aussi reconnaissant pour les responsabilités qu'il m'a confiées pour l'organisation d'Eurocrypt 2017, et je n'oublierai pas l'intense semaine que toute l'équipe a passée.

Je remercie le personnel administratif qui a fréquemment contribué au bon déroulement de ma thèse, en particulier Nathalie Gaudechoux, à qui j'ai toujours pu faire appel, indépendamment du jour et du problème. Je pense aussi à Jacques Beigbeder, Lise-Marie Bivard, Isabelle Delais, Joëlle Isnard, Sophie Jaudon, Valérie Mongiat, Ludovic Ricardou et Benoit Speriazi.

Les nombreux moments passés avec les doctorants, post-docs et jeunes chercheurs de l'équipe crypto ont élargi mon domaine de connaissance en cryptographie, et ont veillé à garder le cap durant cette thèse. Je remercie d'abord les plus anciens doctorants, qui ont partagé leur expérience : Tancrède Lepoint pour Ses conseils divers, Thomas Prest un presque mentor, Mario Cornejo por esos cafés antes de las diez, Alain Passelègue pour nos débats sans fin, Adrian Thillard pour son humour, et aussi Sonia Belaid, Fabrice Benhamouda, et Sylvain Ruhault. Je remercie ceux avec qui j'ai partagé pendant ces trois années la vie au labo, des conférences, et parfois même des vacances : Florian Bourse, Geoffroy Couteau, Rafaël Del Pino, Pierre-Alain Dupont, Thierry Mefenza et Anca Nitulescu. Je remercie particulièrement Romain Gay pour nos repas/soirées au labo, toutes nos discussions crypto et nos digressions. Je tiens aussi à remercier les plus "jeunes" du labo (et leur souhaite bon courage pour la future rédaction) : Balthazar Bauer, Jérémy Chotard, Dahmun Goudarzi (qui mérite bien son mug), Chloé Hébant, Louiza Khati, Michele Minelli, Michele Orrù, Razvan Rosie et Mélissa Rossi. Je remercie beaucoup Aurélien Dupin, sur qui l'on peut compter même pour coder des fonctions Booléennes, discuter de PRG et plus encore. Je pense aussi à ceux que j'ai côtoyés plus brièvement : Raphaël Bost, Céline Chevalier, Simon Cogliani, Angelo De Caro, Aisling Connolly, Itai Dinur, Edouard Dufour-Sans, Pooya Farshim, Houda Ferradi, Georg Fuchsbauer, Rémi Géraud, Giuseppe Guagliardo, Aurore Guillevic, Julia Hesse, Duong Hieu Phan, Thomas Peters, Antonia Schmidt-Lademann et Bogdan Ursu. J'ai une pensée pour l'équipe de LLN où j'ai été très bien accueilli à chaque fois.

Je suis reconnaissant envers mes amis, qui m'ont soutenu, ou tout simplement changé les idées. Je remercie les "cryptis" : Élise, Tom, Adrien, Zoé, Colin, et Anthony pour ces bons moments passés entre Paris, Limoges, les Vosges et Bruxelles. Je remercie tout particulièrement Anthony, qui en plus d'être un ami de longue date est un chercheur avec qui j'ai pris beaucoup de plaisir à travailler. Je remercie Christophe, qui bien qu'il ait préféré la physique, est resté un ami proche. Je remercie le groupe des "corréziens", pour nos week-end ou "jeudi corréziens" à Paris : Maxime (jamais très loin ces 16 dernières années), Jérémy, Baptiste, Aline, Louis, Mélou et Milou, et plus récemment Marlène et Clément. Agradezco a mi grupo del Erasmus : Paulina, Thomas, Magda P, Magda F, Fran y Claire por nuestros encuentros en varias ciudades esos tres años. Je remercie aussi ceux qui ont égayé les dimanches soirs au quizz du Bombardier, comme Ilaria et Camilla.

Cette thèse n'aurait pu être possible sans ma famille, qui sans forcement comprendre ce

qui pouvait autant me plaire dans les maths, m'ont soutenu. Je remercie particulièrement mes parents et ma soeur pour l'amour dont ils m'ont fait part.

Je finis ces remerciements par Éyandé, qui par sa joie de vivre, ses encouragements, son amour, et son imprévisibilité, m'a rendu heureux ces deux dernières années et demie.

Contents

Chapter 1

Introduction

In this chapter we introduce the topic of this thesis in the area of cryptology.

First we briefly motivate cryptology, then we specify the focal point of this thesis, and finally we present the organization of this manuscript.

Contents

Telling everything to everyone might not be the best way to build a human society. When hearing the same message, two people might react quite differently, based on their feelings, knowledge and aspirations; the fear of a bad reaction justifies not communicating everything publicly. When one wants to establish a communication with one person (or a group), rather than with everyone, the notion of secret emerges, as occulted knowledge. The development of this notion leads to cryptology, from the Greek *kruptos*, hidden, and *logos*, the study, the science, as science of the secret.

For a long time, cryptology mostly consisted in enabling private communications between lovers, ensuring secure transmission of military knowledge, or trying to learn these secrets when not authorized to do so. Since then, cryptology is grossly divided in two: first the construction of systems or protocols (cryptosystems) to securely communicate called cryptography, second the analysis of the security and the concrete attacks on these constructions as the cryptanalysis. Cryptology has been considered as an art, before being mentioned as a science and then more systematically defined and studied since the XX-th century. At the beginning, the most important functionalities supplied by cryptography, secure communication between two parties and authentication, were supposed to be guaranteed by ad-hoc constructions, and secret on the design of the cryptosystems. Later, the industrial revolution and the apparitions of machines changed both the possible functionalities and the theoretic treatment of cryptosystems. Indeed, the amount of computations that could make a particular machine, designed to create or break a cryptosystem, is way more important than the one of a human, as witnessed during World War II by the Enigma machine. Since then, cryptology lives in the intersection of mathematics and computer science, the security of the cryptosystems being related to mathematical problems and the limitations of the machines designed to break them.

As any science, cryptology evolves with the epochs, depending on the needs and the capabilities of the society. In our digital era, the number of ways to communicate explodes, together with the concerns about security and privacy, giving birth to plenty of new challenges in cryptology. In this thesis, we focus on one of these challenges, and on the solutions developed to address it.

1.1 Outsourcing Computation

The challenge we investigate in this thesis is outsourcing computation; or more precisely, how a person could allow another entity to perform some computations on his data without renouncing to his privacy relatively to this data. First we motivate this challenge, then we present a first theoretical answer and then a second more suited solution.

1.1.1 Asymmetric Situation

In today's world, computers and Internet are everywhere, and take an important part in the day-to-day life. For a lambda user of Internet, later called Alice, well-connected to the digital world, we highlight the asymmetry between the quantity of data she is the source of, and the devices she owns to process it.

Alice is very connected: she communicates via instant messaging, she is involved in various social networks, she often does on-line shopping, and she uses plenty of apps. Alice is always connected to Internet through various devices, as her smartphone, laptop, smartwatch, tablet computer, smart card, *etc.* The number of on-line activities she may be interested in grows

up whereas the devices she uses are more and more limited devices; devices with small storage capacity and low computational power. The current trend is to be more and more implicated in the digital world and with smaller and smaller connected devices. Alice stores and accesses more and more pictures, music, texts, videos, *etc.* whereas she uses more and more limited devices.

With this trend, Alice is no longer able to store all the data she owns. The limited devices in her possession are not powerful enough to handle the quickly increasing amount of data related to her digital life. As an example, it becomes impossible for Alice to simply access the pictures of her last year holidays on her phone. The storage incapacity is the tip of the iceberg of what Alice cannot do anymore relatively to her data. Beside storing, Alice can be interested in doing some processing on her data, as some research functions, on her emails or address book; as photographic modification, correcting the red-eye effect or more generally photoshopping. She can be interested in computing basic statistics over her data, as financial incomes and outcomes or more involved computations.

What Alice cannot do anymore on her data, other entities can. In the recent years the proliferation of small embedded devices with limited storage and computing facilities for users have been compensated by the apparition of cloud services with extensive storage and computing facilities. Therefore, to bypass her storage and computation boundaries, Alice can use a cloud, later called Claude, a society using multiple servers and providing huge storage and computational power. This context justifies the outsourcing of data and the delegation of data processing. With an agreement between Alice and Claude, all storage and computational functionalities desired by Alice can be handled.

Nevertheless, it raises new security and privacy concerns, Alice may not want the cloud to access all her data and learn her whole life. The data outsourced by Alice can be sensitive as health information or business indications, and she may not want the cloud to own and potentially use or distribute the outsourced data or any knowledge inferred from this data. Namely, users typically want to prevent the server from learning about their data and processing. Such privacy for outsourcing computation framework can be guaranteed using Fully Homomorphic Encryption (FHE).

1.1.2 Fully Homomorphic Encryption

Fully homomorphic encryption allows to encrypt data such that any operation on the data can be performed on the encrypted data without knowing the concrete values of this data. Gentry's breakthrough fully homomorphic encryption scheme [Gen09] brought a perfect conceptual answer for outsourcing computation.

In this context, it means that Alice can encrypt her data, and that Claude can perform computations on the encrypted data. Alice learns the result of the computation whereas Claude learns nothing on Alice's initial data or on the result of the computation. This use of fully homomorphic encryption can be described in terms of cloud-service application framework. First, Alice subscribes to Claude for outsourcing her computations and they exchange some information. Then, Alice sends homomorphically encrypted data to Claude, corresponding to a storage phase. When Alice wants to get the result of a computation on her data, she describes the computation to Claude, he evaluates it on the encrypted data and returns a homomorphic ciphertext to Alice. Finally, Alice decrypts and obtains the results of the computation she asked for. It is also possible for Alice to repetitively encrypt new data to Claude and ask for other computations.

Fully homomorphic encryption gives access to an outsourcing computation framework with privacy guarantees for Alice, as desired. However, today's knowledge on fully homomorphic encryption does not give a practical solution for this context. Indeed, current fully homomorphic encryption (abbreviated as FHE) schemes are not efficient enough to be performed on limited devices as the one used by Alice. Diverse barriers prevent this framework from being a practical solution, given the limited devices of Alice and the time of execution she can expect. The main limitation for the deployment of cloud services based on such FHE frameworks relates to its important overheads, that can be linked to two main concerns. First, the computational and memory costs, especially on Alice's side, are very important. Homomorphic encryption and decryption algorithms that Alice should execute are still very expensive in time, and the memory cost is mostly influenced by the homomorphic ciphertexts and public key sizes which are still prohibitive for limited devices. For use-cases where Alice needs to contract a cloud to perform the computations she wants, we cannot assume that she has powerful enough devices to run a typical FHE scheme. Second, fully homomorphic encryption allows to evaluate any function on the encrypted data but any processing is not performed with the same efficiency; this efficiency can be measured in terms of homomorphic capacity. The homomorphic capacity relates to the fact that FHE constructions are built on noise-based cryptography, where the unbounded amount of homomorphic operations is guaranteed by an expensive bootstrapping technique. The homomorphic capacity corresponds to the amount of operations doable before the noise grows too much, forcing the use of bootstrapping and preventing quick evaluation.

These two main limitations lead to not using fully homomorphic encryption schemes on their own for outsourcing computation, but preferably use FHE as a building block. Taken into account these strong limitations, a more suited framework arises, called hybrid homomorphic framework, and later referred to as efficient homomorphic framework.

1.1.3 Hybrid Fully Homomorphic Framework

One possible solution for outsourcing computation (efficiently) is to use a hybrid encryption scheme: preserving the homomorphic encryption scheme to keep the privacy of Alice's data, and using an efficient scheme to make the framework handleable in memory for Alice's limited devices and in time for real-life applications. Consequently, the efficient homomorphic framework contains two encryption schemes, a (fully) homomorphic one, and another used for its efficiency, that will be a symmetric encryption scheme. The whole framework, appearing first in [LNV11], is called hybrid fully homomorphic framework, or hybrid homomorphic framework, depending on whether we consider a homomorphic encryption scheme enabling to evaluate any function (case of fully), or a restricted class of functions.

The framework is called hybrid as it uses two types of encryption schemes, and it can be described in the following way. First, Alice contracts Claude for outsourcing her computations and they exchange some information about the two schemes. Then, Alice symmetrically encrypts her data and sends it to Claude, corresponding to the storage phase. Claude computes a homomorphically encrypted version of Alice's data from the information given by Alice at the contract signature and the symmetrically encrypted data he received. As in the previous framework, when Alice wants to get the result of a computation on her data, she describes the computation to Claude and he evaluates it on the homomorphically encrypted data. Then, Claude compresses the (homomorphically) encrypted result and sends it back to Alice, who decrypts it and gets the result.

Three differences between this hybrid framework and the previous one are the core points to get a framework efficient for outsourcing computation. These critical points are the symmetric encryption done by Alice, the transformation performed by Claude and the compression. To develop them we use the following denomination, the data of Alice is called plaintext and then ciphertext when it is encrypted, the words symmetric, homomorphic and others from the same families are used to differentiate what relates to the symmetric scheme or to the homomorphic one. The symmetric encryption done by Alice has to be easier to handle for the devices she owns than homomorphic encryption. This is generally the case as symmetric encryption is supposed to be fast and symmetric ciphertexts are supposed to be much smaller in term of data than homomorphic ciphertexts. The compression is a feature exhibited for some existing homomorphic scheme, it consists in transforming the homomorphic ciphertext in a ciphertext of reduced size such that decryption is still possible. In the known FHE constructions there is some redundancy in the ciphertexts, the associated structure allows the homomorphisms but this structure is not necessary to recover the plaintexts. As an example, in some schemes the matrix structure of the ciphertexts allows homomorphic operations whereas only one column of the ciphertext is sufficient to decrypt. The transformation performed by Claude is the trickiest part of the hybrid encryption and the central problem of (fully) hybrid homomorphic framework.

Claude has some information on the two schemes from his first exchange with Alice and he needs to convert ciphertexts for the symmetric scheme into ciphertexts for the homomorphic scheme. More details on the pieces of information given by Alice relatively to the two schemes are necessary to understand this transformation. For the symmetric encryption scheme, a piece of information called key is chosen and kept private by Alice. This key combined with a plaintext gives the corresponding symmetric ciphertext (encryption), and reciprocally this key combined with a ciphertext gives the corresponding plaintext (decryption). Therefore, Alice keeps private her key and sends only the specifications of the scheme to Claude. For the homomorphic encryption scheme, there are two important pieces of information, one allowing anyone to encrypt, which is sent to Claude, and one allowing to decrypt which is kept private by Alice. Alice communicates this information, the specifications of the scheme and a homomorphic encryption of the (symmetric) key, to Claude. The cloud cannot recover this key, as he can encrypt homomorphically but he cannot decrypt, however with all these informations he can do the transformation.

Claude cannot evaluate homomorphic operations on the symmetric ciphertexts, but he can do homomorphic encryption, hence he homomorphically encrypts the symmetric ciphertexts. Some caution is required here: these ciphertexts correspond to the homomorphic encryption of symmetrically encrypted plaintexts (two layers of encryption) and they are not homomorphic ciphertexts relatively to the considered scheme, so it is not sufficient for the conversion. Instead, if we consider as function the decryption of the symmetric scheme, Claude owns homomorphic encryptions of all its inputs: the homomorphic encryption of the key sent by Alice, and the homomorphic encryption of the symmetric ciphertext he computed. Using the fully homomorphic property, he can compute the homomorphic encryption of any function evaluated on homomorphically encrypted ciphertext. Then, Claude homomorphically evaluates the symmetric decryption function on these encrypted inputs, which gives a homomorphic encryption of the plaintext, and finalizes the transformation. A more efficient method is used in practice for this transformation, using a more intricate hybrid decryption function. It consists in tweaking the homomorphic evaluation of the symmetric decryption function to directly take as inputs the symmetric ciphertexts rather than homomorphic encryption

of these ciphertexts. It avoids an unnecessary homomorphic encryption, and it enables to design a decryption function more efficiently evaluable. This tweaked evaluation is even more adapted for some symmetric schemes, later leading to consider stream ciphers more particularly.

With this hybrid framework, all the algorithms performed by Alice are handleable by constrained devices, as Alice performs the algorithms of the symmetric scheme and a simplified decryption of the homomorphic scheme. On his side, Claude takes care of all the homomorphic operations, which is adapted to his huge memory and computational power. Even with the second scheme, he only gets encrypted versions of Alice's data, keeping the privacy concern of the whole framework. These two points are sufficient to design the hybrid homomorphic framework as a better candidate for outsourcing computation than fully homomorphic encryption framework. However, core questions on this hybrid framework still have to be correctly addressed to provide a concrete solution.

Mainly, the efficiency of the whole framework depends on the symmetric encryption scheme used, and its compatibility with homomorphic encryption. Its efficiency as encryption scheme used on limited devices is already assumed, but it does not guarantee that the corresponding decryption function can be quickly evaluated by Claude to produce homomorphically encrypted data. A first problem is therefore to determine which functions can be evaluated quickly enough by Claude in this context. Note that the step performed by Claude to obtain homomorphically encrypted data should not be noticed by Alice, as it does not depend on the computation she asks for. Then, it requires to find or build a symmetric encryption scheme that can be expressed in terms of these functions easy to homomorphically evaluate. The high compatibility between the two schemes makes a small overhead of Claude's conversion step. Finally, the security of this scheme is a main concern for the privacy requirements of the whole outsourcing computation framework.

1.2 Our Results

In this thesis we present our results relative to the hybrid homomorphic framework previously introduced. These results come from two articles and further personal results. The first article, "Towards Stream Ciphers for Efficient FHE with Low-Noise Ciphertexts" [MJSC16], published at EUROCRYPT 2016, is a joint work with Anthony Journault, François-Xavier Standaert and Claude Carlet, introducing a new kind of stream-ciphers designed for the context of outsourcing computation. This article contributes to a better study of the homomorphic error-growth of a family of fully homomorphic encryption schemes in theory and practice, it presents a new design of stream ciphers, the Filter Permutator, with associated security analysis and gives a candidate instantiation of hybrid homomorphic framework with record efficiency. The second article, "Boolean functions with restricted input and their robustness; application to the FLIP cipher" [CMR17], accepted in ToSC(Transactions on Symmetric Cryptology) 2018, is a joint work with Claude Carlet and Yann Rotella, investigating the cryptographic criteria when the input of a Boolean function is restricted. This article presents a theoretical study on the main cryptographic criteria on Boolean functions when they are evaluated only on a restricted set of inputs, it shows how this study can be used to investigate the security of some symmetric encryption schemes, and more precisely how it can contribute to the security analysis of Filter Permutator instantiations.

Therefore in this thesis we present a hybrid framework for outsourcing computation and

instantiations, and more specifically theoretical results developed to achieve it, on the error growth metric of a FHE and on Boolean functions.

1.2.1 Hybrid Homomorphic Framework and Error-Growth

The first component of the hybrid homomorphic framework we consider is a (fully) homomorphic encryption scheme. For the efficiency of the whole framework this scheme has to be compatible with the symmetric scheme and well understood. Therefore our first contribution concerns homomorphic encryption, for which we solve the efficiency bottlenecks by analyzing functions efficiently evaluable homomorphically.

We analyze the error-growth produced by homomorphic evaluation using a particular family of homomorphic encryption schemes. This detailed analysis enables to consider particular functions which involve a low error-growth and therefore to determine good building blocks for a scheme homomorphically evaluated quickly and with limited memory resources.

We also define more formally the hybrid homomorphic framework, showing how and in which cases its efficiency can be improved, depending on particular choices of the two component schemes.

1.2.2 New Stream Cipher Design: the Filter Permutator

The Filter Permutator is the stream-cipher construction we designed for homomorphic encryption framework. Its goal is to be instantiated as a symmetric encryption scheme with the most efficient homomorphic evaluation of its decryption relatively to a specified homomorphic encryption scheme.

We analyze the behavior of various symmetric schemes relatively to homomorphic evaluations, and then more particularly the error-growth associated to the Filter Permutator construction. In particular we investigate the efficiency of its instantiation: FLIP ciphers, in general and more concretely in the hybrid homomorphic framework context.

The security of the Filter Permutator and its instantiation as symmetric encryption scheme are examined. Various attacks are considered, some generic ones for stream-ciphers, others dedicated to the new design of the Filter Permutator, a class of attacks connected to guess-and-determine strategies and a new class related to a Hamming weight invariant appearing in the scheme.

1.2.3 New Criteria on Boolean Functions

Relatively to the specific functions involving a low error-growth and the security analysis of the Filter Permutator we investigate particular cryptographic criteria on Boolean functions. This study aims to determine which functions can provide security in the considered context and it contributes to the knowledge of Boolean functions used in cryptography and cryptanalysis.

The functions efficiently homomorphically evaluated, also called low-cost functions, are not currently considered Boolean functions. Consequently we examine their parameters relatively to the standard cryptographic criteria on Boolean function as algebraic immunity, nonlinearity and resiliency.

New cryptographic criteria on Boolean functions emerge from the security analysis of the Filter Permutator, referred as recurrent criteria and restricted input criteria. We study these adaptations, or generalizations, of standard cryptographic criteria on Boolean functions;

examining the behavior of commonly considered Boolean functions and investigating optimal constructions.

1.2.4 Organization

According to these contributions, the thesis is organized in the following way. After this introduction, Chapter 2 contains the preliminaries, introducing the notions used through the thesis. Then, Chapter 3 presents the results relatively to fully homomorphic encryption, in Chapter 4 we develop the construction of the Filter Permutator, its achievements and its security, and in Chapter 5 we investigate the related new cryptographic Boolean criteria. Finally, Chapter 6 concludes on the contributions presented in this thesis and gives more perspectives.

1.3 Other Contributions

Besides the material exposed in this thesis, we worked on other cryptographic topics, one of them leading to the article "Predicate Encryption for Multi-Dimensional Range Queries from Lattices" [GMW15], published at PKC 2015. In this joint work with Romain Gay and Hoeteck Wee we construct a predicate encryption scheme for two principal predicates, with security based on Learning With Errors.

Predicate encryption is a recent paradigm to treat search queries on encrypted data. A predicate encryption scheme is a kind of public key scheme where ciphertexts (encrypted data) are associated with a descriptive value x in addition to the plaintext (data), and secret keys are associated to a predicate f (*i.e.* a Boolean function) such that the decryption is possible only if $f(x) = 1$. In this article, we consider the predicate of multi-dimensional range queries: we are given a point $(z_1, \cdots, z_D) \in [T]^D$, where $[T]$ denotes the set of integers $\{1, \cdots, T\}$, and interval ranges $[x_1, y_1], \cdots, [x_D, y_D] \subseteq [T]$ and we want to know if $(x_1 \leq z_1 \leq y_1) \wedge \cdots \wedge (x_D \leq z_D \leq y_D)$. We also give an adapted construction for the predicate of multi-dimensional subset queries where the ranges of the precedent definition are replaced by subsets of $[T]$. On the predicate side the results are obtained by firstly building a predicate encryption scheme with the appropriate notions of security for a simple predicate of disjunctions of equalities. Then this predicate is generalized to the conjunction of disjunctions of equalities and other reductions between predicates enable to get results on the predicates multi-dimensional range queries and multi-dimensional subset queries.

On the lattice side, the security of the scheme is based on the standard Learning with error problem. Our result comes from an adaptation of lattice-based identity based encryption and techniques on lattice trapdoors. The encryption scheme follows the generic LWE-based anonymous identity based encryption obtained using the trapdoor sampling techniques. The correctness of the scheme is obtained by using properties on trapdoors on the extending right part of a matrix, as used in (hierarchical) identity based encryption. The security of the scheme, selectively secure and weakly attribute-hiding, is obtained using the leftover hash lemma and the technique of trapdoor extension on left side of the matrix. We refer to [GMW15] for the detailed constructions and proofs as it is not the theme of this thesis.

Chapter 2

Preliminaries

In this chapter, we introduce the notations used throughout this manuscript and give some definitions and notions.

We begin with standard notations of mathematics and computer science, then we define less general notions. More particularly we introduce the basic notions on cryptographic primitives, on Lattice-based cryptography and on Boolean functions later referred in this thesis. Finally we give additional preliminaries, completing the introduction part of the thesis.

Contents

2.1 Notations and Preliminaries

The notation and preliminaries introduced in this chapter are common in cryptography or are basic notions used later in the thesis, they enable us to define and give basic results on objects used or mentioned later. These preliminaries are adapted from several books, cited in the relevant parts, and inspired by various PhD thesis preliminaries; the ones by Fabrice Ben Hamouda–Guichoux [Ben16], Geoffroy Couteau, Virginie Lallemand [Lal16], Alain Passelègue [Pas16], and Thomas Prest [Pre15].

2.1.1 Mathematical Notations

2.1.1.1 Sets, Rings, Integers

We denote by $\mathbb{Z}$ the set of integers, by $\mathbb{N}$ the set of non-negative integers and by $\mathbb{R}$ the set of real numbers. For a and b two integers such that $a < b$, we denote by $\{a, \cdots, b\}$ the set of integers between a and b (a and b included); if $a = 1$ we denote this set $[b]$. $s \in \{0,1\}^n$ is a bitstring of length n and we denote $E_{n,k}$ the set of all bitstrings of length n with k 1's and $n-k$ 0's. For a finite set we denote $|S|$ its cardinality.

If n is a positive integer, $\mathbb{Z}_n$ denotes the ring of integers modulo n. $\mathbb{F}_2$ denotes the field of two elements $(\mathbb{Z}_2, +, \cdot)$, also referred as Boolean field; relatively to this field the operation $+$ is called addition or XOR, also denoted $\oplus$. If R is a ring, then $R[X]$ denotes the ring of polynomials with coefficients in R.

For $x \in \mathbb{Z}$, $x \mod q$ or $[x]_q$ denotes the remainder of the Euclidean division of x by q. For $y \in \mathbb{R}$, $|y|$ is its absolute value, $\log(y)$ denotes its logarithm in basis 2, $\lceil y \rceil$ is the smallest integer a such that $y \leq a$; $\lfloor y \rfloor$ is the biggest integer a such that $a \leq y$. The rounding $\lfloor [a]_q \rceil_2 \in \{0,1\}$ is a function in $a \in \mathbb{Z}_q$ giving 1 if $\lfloor \frac{q}{4} \rfloor \leq a \leq \lfloor \frac{3q}{4} \rfloor$ and 0 otherwise.

2.1.1.2 Matrices and Vectors

Vectors are usually bold (*e.g.* $\mathbf{u}$, $\mathbf{v}$) and by default column vectors, while matrices are usually denoted by capital bold letters (*e.g.* $\mathbf{A}$, $\mathbf{G}$).

For a matrix $\mathbf{M}$ we refer to the i-th row as $\mathbf{m}_i^\top$ and to the j-th column as $\mathbf{m}_j$ (as we do not use the operation transposition the sign $^\top$ enables to differentiate row vectors from column vectors).

If $\boldsymbol{u} \in R^n$ is a vector, its coordinates or entries are $u_1, \ldots, u_n$ (not in bold). We use $|x|$ and $||x||_2$ for the standard norms 1 and 2 on vectors $x \in \mathbb{R}^n$. For two vectors $\mathbf{u}$ and $\mathbf{v}$ of R^n, $< \mathbf{u}, \mathbf{v} >$ or $\mathbf{u} \cdot \mathbf{v}$ denotes their inner product $\sum_{i \in [n]} u_i v_i$.

Elements u, v of $\mathbb{F}_2^n$ are sometimes considered as bitstrings of $\{0,1\}^n$ with order $\preceq$ where $u \preceq v$ if and only if for all $i \in [n]$ $u_i \leq v_i$. For a bitstring $u \in \{0,1\}^*$ we denote $|u| \in \mathbb{N}$ its length. For an element u of $\mathbb{F}_2^n$, considered as a bitstring or a vector we define its Hamming weight (or weight), denoted $\mathsf{w_H}(v)$ as the number of its non-zero coefficients.

2.1.1.3 Distributions and Probabilities

If S is a set, $x \leftarrow_\$ S$ indicates that x is taken uniformly at random from the set S (independently of everything else). Similarly, if χ is a probability distribution, $x \leftarrow_\$ \chi$ indicates that x is drawn randomly according to χ. For a probability distribution we generally denote μ its mean and σ its parameter *i.e.* standard deviation.

We denote $Pr[X = x]$ the probability of a random variable X taking value x, and $Pr_{x\in\chi}[f(x) = y]$ to denote the probability that $f(x)$ is equal to a fixed value y when x is sampled from the distribution χ. We denote by U_n the uniform distribution over $\{0,1\}^n$. We often use the term "random" to mean "uniformly at random".

2.1.1.4 Miscellaneous

When f and g are two functions from $\mathbb{N}$ to the set of real numbers, we write $f = O(g)$ or $g = \Omega(f)$ to indicate that there exists a constant c and an integer $n_0 \in \mathbb{N}$ such that for any integer $n \geq n_0$, $|f(n)| \leq c \cdot |g(n)|$.

2.1.2 Algorithms, and Provable Security

We will use the notion of algorithm (at least implicitly) in most of this thesis and the notion of provable security when dealing with homomorphic encryption.

2.1.2.1 Algorithms, and Turing machines

Algorithms are programs for Turing machines. By default, algorithms are probabilistic, *i.e.* they can use an additional tape of the Turing machine containing random bits (random coins). For an algorithm A, we write $y \leftarrow A(x)$ or $y = A(x)$ to say that we execute the algorithm A on input x with fresh random coins and that we store the result in y. For a deterministic A, we also use the same notation.

By default, algorithms are not supposed to run in polynomial time; if we want them to be polynomial time, we say it explicitly, denoting by PPT algorithm a probabilistic polynomial time algorithm. Relatively to provable security, we qualify an algorithm as "efficient" if it is a PPT algorithm in its input size (in opposition to exponential time).

2.1.2.2 Security Parameters and Negligibility

Almost any cryptosystem could be broken with a powerful enough computer, then the goal of cryptographers is to construct fast enough cryptosystems which cannot be broken by reasonable computers. Estimating an upper bound on the amount of computations deployable to break a cryptosystem gives a security level, defining the limit between cryptosystems considered insecure or secure. In practice, we often consider today that if at least 2^{128} elementary operations are required to break a cryptosystem with high probability (greater than $1/2$), then this cryptosystem is considered secure, this cryptosystem is said to provide $\lambda = 128$ bits of security. 80 bits of security is also a very commonly used level of security, particularly for cryptosystems used in the context of low-cost devices.

For asymptotic constructions, it is relevant to formalize this idea of security as a security parameter $\lambda \in \mathbb{N}$. In this particular context, all algorithms of a cryptosystem take as input a unitary representation 1^λ and all these algorithms run in polynomial time in λ. Adequately all attacks against the cryptosystem are performed by algorithms running in time polynomial in λ.

For concrete attacks, when the parameter λ is fixed to a particular integer, we consider the cryptosystem broken if the amount of computation of an algorithm is smaller than 2^λ elementary operations and if its data complexity is smaller than 2^λ, as detailed later in this chapter.

The notion of negligibility is often used for security, to mathematically quantify the proportion of events we can neglect. We say that the quantity ε is negligible or $1-\varepsilon$ is overwhelming, if for any $k \in \mathbb{N}$, $\varepsilon = O(1/\lambda^k)$. When the parameter λ is fixed, we will consider as negligible a positive quantity ε such that $\varepsilon \leq 2^{-\lambda}$.

2.1.2.3 Adversaries, Experiments, Oracles, Success Probabilities and Advantage.

Adversaries are probabilistic algorithms or Turing machines, we consider that they implicitly take as input a unary representation of the security parameter. As inputs to adversaries are always polynomial in the security parameter, a polynomial-time adversary runs in time polynomial in the security parameter. We write $\mathcal{A}(x)$ to say that the adversary $\mathcal{A}$ is called with input x.

An experiment is a succession of executions of algorithms; security notions and security assumptions are often described as experiments where an adversary is called one or several times with various inputs. An experiment can also be seen as a game between an adversary $\mathcal{A}$ and an implicit challenger which gives its input to the adversary and allows some oracle calls. As usual, experiments are parameterized by the security parameter λ.

An adversary may also be given access to oracles, machines which running times are not taken into account in the running time of the adversary: a query to an oracle always only counts for one clock cycle. We write $\mathcal{A}^{\mathsf{O}}(x)$ to say that the adversary $\mathcal{A}$ is called with input x and has access to the oracle O.

The success probability of an adversary $\mathcal{A}$ in an experiment $\mathsf{Exp}^{\mathtt{exp}}$ is the probability that this adversary outputs 1 in this experiment:

$$\mathsf{Succ}^{\mathtt{exp}}(\mathcal{A}, \lambda) = Pr[\mathsf{Exp}^{\mathtt{exp}}(\mathcal{A}, \lambda) = 1].$$

Sometimes, it is more relevant to formulate a security notion in term of advantage. When the security notion or assumption consists in distinguishing two experiments $\mathsf{Exp}^{\mathtt{exp\text{-}0}}$ and $\mathsf{Exp}^{\mathtt{exp\text{-}1}}$, we define the advantage of an adversary $\mathcal{A}$ in distinguishing the experiments $\mathsf{Exp}^{\mathtt{exp}\text{-}b}$ (where $b \in \{0,1\}$) as:

$$\mathsf{Adv}^{\mathtt{exp}}(\mathcal{A}, \lambda) = \left| Pr[\mathsf{Exp}^{\mathtt{exp\text{-}1}}(\mathcal{A}, \lambda) = 1] - Pr[\mathsf{Exp}^{\mathtt{exp\text{-}0}}(\mathcal{A}, \lambda) = 1] \right|.$$

When the experiment $\mathsf{Exp}^{\mathtt{exp}}$ corresponds to a cryptographic assumption or to a security notion, we say that the assumption or the security notion statistically holds when this success probability is negligible (in λ) for any unbounded adversary $\mathcal{A}$. It computationally holds when this success probability is negligible (in λ) for any polynomial-time adversary $\mathcal{A}$.

2.2 Cryptographic Primitives

2.2.1 Generic Primitives

We introduce some basic cryptographic primitives as in [Gol01], referring to this book for a rigorous and developed treatment of these primitives and associated security notions.

First we present the concept of one-way function, a core notion in cryptography expressing the property of some functions to be easy to evaluate and difficult to invert.

Definition 2.2.1 (One-Way Function)**.** *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called One-Way if the two following conditions hold:*

- *Easy to compute: There exists a polynomial-time algorithm A such that $A(x) = f(x)$ for all x.*
- *Hard to invert: For every* PPT *algorithm B, every positive polynomial $p(.)$, and all sufficiently large n's:*

$$Pr[B(f(U_n), 1^n) \in f^{-1}(f(U_n))] < \frac{1}{p(n)}.$$

Note that the second item corresponds to a negligible probability. One-way functions play a central role in cryptography as, informally, they enable efficient computation for a user and inefficient computation (overwhelmingly) for an attacker. In provable security most of the functions considered one-way are functions which inversion rely on a mathematical problem assumed difficult (with all known resolution algorithm running in exponential time).

To introduce the other basic cryptographic primitives we need to precise the notions of indistinguishability and pseudorandomness. The first one expresses the difficulty of distinguishing two mathematical objects, distributions mostly. As it is difficult to tackle the randomness quality of a sequence of elements, the second notion permits to qualify what we can consider as random, or more precisely what we cannot reject as random samples. It leads to the three following definitions:

Definition 2.2.2 (Probability Ensemble)**.** *Let I be a countable index set. An ensemble indexed by I is a sequence of random variables indexed by I. Namely, any $X = \{X_i\}_{i \in I}$, where each X_i is a random variable is an ensemble indexed by I.*

Definition 2.2.3 (Polynomial-Time Indistinguishability)**.** *Let X and Y be two ensembles indexed by $\mathbb{N}$, they are indistinguishable in polynomial-time if for every* PPT *algorithm A, every polynomial $p(.)$ and all sufficiently large n's:*

$$|Pr[A(X_n, 1^n) = 1] - Pr[A(Y_n, 1^n) = 1]| < \frac{1}{p(n)}.$$

This definition can be extended to other ensembles.

Definition 2.2.4 (Pseudorandom Ensembles)**.** *The ensemble $X = \{X_n\}_{n \in \mathbb{N}}$ is called pseudorandom if there exists a uniform ensemble $U = \{U_{\ell(n)}\}_{n \in \mathbb{N}}$, where $\ell : \mathbb{N} \to \mathbb{N}$, such that U and X are indistinguishable in polynomial time.*

With these definitions we can formally define the important cryptographic primitive called Pseudorandom generator; later in this thesis we will denote PRG or PRNG for Pseudorandom (Number) Generator such constructions.

Definition 2.2.5 (Pseudorandom Generator (PRG))**.** *A pseudorandom generator is a deterministic polynomial-time algorithm G satisfying the two following conditions:*

- *Expansion: There exists a function $\ell : \mathbb{N} \to \mathbb{N}$ such that $\ell(n) > n$ for all $n \in \mathbb{N}$, and $|G(s)| = \ell(|s|)$, for all bitstring $s \in \{0,1\}^*$.*
- *Pseudorandomness: The set $\{G(U_n)\}_{n \in \mathbb{N}}$ is pseudorandom.*

We finish this introduction of basic cryptographic primitives by defining pseudorandom functions and pseudorandom permutations. Informally, these functions are such that it is impossible to efficiently distinguish if they are taken from a particular set or randomly taken from the set of all functions (of specified input and output size). Formally it leads to the following definitions:

Definition 2.2.6 (Function Ensembles)**.** *Let $\ell : \mathbb{N} \to \mathbb{N}$, a ℓ-bit function ensemble is a sequence $F = \{F_n\}_{n\in\mathbb{N}}$ of random variables such that the random variable F_n assumes values in the set of functions mapping $\ell(n)$-bit-long strings to $\ell(n)$-bit-long strings.*

The uniform ℓ-bit function ensemble, denoted $H = \{H_n\}_{n\in\mathbb{N}}$ has H_n uniformly distributed over the set of all functions mapping $\ell(n)$-bit-long strings to $\ell(n)$-bit-long strings.

This definition can be generalized to functions with different input and output sizes.

Definition 2.2.7 (Pseudorandom Function (PRF))**.** *Let $d, r : \mathbb{N} \to \mathbb{N}$. We say that:*

$$\{f_s : \{0,1\}^{d(|s|)} \to \{0,1\}^{r(|s|)}\}_{s\in\{0,1\}^*},$$

is a pseudorandom function ensemble if it satisfies the following two conditions:

- *Efficient Evaluation: There exists a* PPT *algorithm that on input s and $x \in \{0,1\}^{d(|s|)}$ returns $f_s(x)$*
- *Pseudorandomness: For every* PPT *adversary using oracles O_{F_n} sampling from F_n or O_{H_n} sampling from H_n, every polynomial $p(.)$, and all sufficiently large n's:*

$$|Pr[\mathcal{A}^{\mathsf{O}_{F_n}}(1^n) = 1] - Pr[\mathcal{A}^{\mathsf{O}_{H_n}}(1^n) = 1]| < \frac{1}{p(n)},$$

  where F_n is a random variable uniformly distributed over the multi-set $\{f_s\}_{s\in\{0,1\}^n}$, and H_n is uniformly distributed among all functions mapping $d(n)$-bit-long strings to $r(n)$-bit-long strings.

We call pseudorandom functions, denoted PRF, functions from this ensemble.

This definition corresponds more precisely to efficiently computable generalized pseudorandom function ensemble in [Gol01], we adapt it as we will not consider other kinds of PRF.

The more specific concept of pseudorandom random permutation can be of particular use. We similarly define it.

Definition 2.2.8 (Permutation Ensembles)**.** *A permutation ensemble is a sequence $P = \{P_n\}_{n\in\mathbb{N}}$ of random variables such that the random variable P_n assumes values in the set of permutations mapping n-bit-long strings to n-bit-long strings.*

The uniform permutation ensemble, denoted $K = \{K_n\}_{n\in\mathbb{N}}$ has K_n uniformly distributed over the set of all permutations mapping n-bit-long strings to n-bit-long strings.

Definition 2.2.9 (Pseudorandom Permutation (PRP))**.** *A permutation ensemble $P = \{P_n\}_{n\in\mathbb{N}}$ is called pseudorandom if it satisfies the three following conditions:*

- *Efficient Evaluation: There exists a* PPT *algorithm that on input $s, |s| = n$ and $x \in \{0,1\}^n$ returns $f_s(x)$*

- *Efficient Inversion: There exists a* PPT *algorithm that on input* $s, |s| = n$ *and* $x \in \{0,1\}^n$ *returns* $f_s^{-1}(x)$
- *Pseudorandomness: For every* PPT *adversary using oracle* O_{P_n} *sampling from* P_n *or* O_{K_n} *sampling from* K_n*, every polynomial* $p(.)$*, and all sufficiently large* n*'s:*

$$|Pr[\mathcal{A}^{\mathsf{O}_{P_n}}(1^n) = 1] - Pr[\mathcal{A}^{\mathsf{O}_{K_n}}(1^n) = 1]| < \frac{1}{p(n)},$$

where K_n *is uniformly distributed among all permutations of* n*-bit length strings.*

We call pseudorandom permutations, denoted PRP, functions from this ensemble.

This definition corresponds more precisely to efficiently computable and invertible pseudorandom function ensemble in [Gol01], we adapt it as we will not consider other kinds of PRP.

The four primitives presented above are theoretical objects permitting many applications and they are a (unreachable) goal to keep in mind when constructing cryptosystems, a direction to follow. The existence of one-way functions would imply the inequality of the computational classes P $\neq$ NP, this equality or inequality being (one of) the main open problems of computer science. The existence of these four primitives has been proven equivalent since the eighties, OWF and PRG [GL89], PRG and PRF [GGM84], PRF and PRP [LR88].

Consequently, we will mention these concepts for some cryptographic constructions we consider as potential instantiation of this theoretic primitives. In the following part we introduce current families of instantiations of some of these primitives for fixed security parameters.

2.2.2 Symmetric Encryption

We begin by giving a general definition of an encryption scheme, before specifying it to symmetric (or secret key) encryption.

2.2.2.1 Encryption Scheme

Encryption is one of the most fundamental primitives in cryptography, it permits to communicate between two users; an encrypted message under a first key is sent to the user knowing a second key (the same or a different one, making the difference between symmetric and asymmetric encryption) enabling to decrypt and recover this message. The two users communicate securely if anybody without knowledge of the second key intercepting the encrypted message learns nothing on the original message. The original message is called plaintext and the encrypted message is called ciphertext.

Definition 2.2.10 (Encryption Scheme)**.** *Let* $\mathcal{M}$ *be the plaintext space,* $\mathcal{C}$ *the ciphertext space and* λ *the security parameter. We define an encryption scheme as a tuple of three polynomial-time algorithms:*

- $\mathsf{KeyGen}(1^\lambda)$*. Generates a pair* $(\mathsf{sk}, \mathsf{pk})$*, where* pk *is an encryption key (public key) and* sk *is a decryption key (secret key); these two keys are supposed to implicitly contain the global parameters.*

- $\mathsf{Enc}(m, \mathsf{pk})$. *Generates $c \in \mathcal{C}$ called a ciphertext encrypting the plaintext (or message) $m \in \mathcal{M}$ under the public key pk.*
- $\mathsf{Dec}(c, \mathsf{sk})$. *Decrypts the ciphertext $c \in \mathcal{C}$ using the secret key sk, outputs $m' \in \mathcal{M}$.*

The encryption scheme should satisfy a notion of correctness (and of security which will be discussed later). For symmetric encryption scheme we will consider the property of perfect correctness.

Definition 2.2.11 (Perfect Correctness)**.** *An encryption scheme satisfies the Perfect Correctness property if for any key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, for any plaintext $m \in \mathcal{M}$, for any ciphertext $c \leftarrow \mathsf{Enc}(m, \mathsf{pk})$, we have $\mathsf{Dec}(c, \mathsf{sk}) = m$.*

A symmetric encryption scheme, or secret key encryption scheme, is an encryption scheme such that $\mathsf{pk} = \mathsf{sk}$. It means that there is only one key in the scheme, which is therefore secret, and two users can communicate securely via a symmetric encryption scheme if they use the same key. When various kinds of encryption schemes are used, we distinguish a symmetric encryption scheme using the capital letter S, at the beginning for an algorithm as $S.\mathsf{Enc}$ or as exponent as for sk^S.

Various notions of security can be defined on symmetric encryption schemes, as investigated in [BDJR97], mostly relying on the assumption of some constructions to be PRP or PRF. These notions go from the consideration of an adversary learning the secret key to its advantage to distinguish a ciphertext from a random bitstring (of specified length), through various models of attacks. As in this thesis we will focus on a concrete family of symmetric encryption schemes, we introduce in the following part concrete instantiations of symmetric encryption schemes and a concrete consideration of the security.

2.2.2.2 Instantiating Symmetric Encryption Schemes

Instantiating symmetric encryption, depending on the plaintext space, two classes of schemes appear; the block-ciphers close to the concept of PRP and the stream ciphers close to the concept of PRG. We introduce here some basic notions of these constructions as we will refer to it later in this thesis, more details on these classic instantiations can be found in books on applied cryptography *e.g.* [MvV97].

Block-ciphers are deterministic algorithms acting on a fixed-length bitstring, called a block and parametrized by a key. More formally we give the following definition of block-cipher:

Definition 2.2.12 (Block-cipher)**.** *A Block-cipher is a family of permutations of n bits parametrized by a key sk:*

$$E_{\mathsf{sk}} : \mathbb{F}_2^n \to \mathbb{F}_2^n,$$

where E_{sk} and E_{sk}^{-1} can be computed by an efficient deterministic algorithm.

They are the core piece of block-cipher encryption, a kind of symmetric encryption where the length of the key sk is generally the security parameter λ and for the key generation algorithm the key is randomly chosen in the set of all bitstrings of length λ. For encryption, the plaintext considered as a bitstring is divided in blocks of length n (which is often equal to $|\mathsf{sk}|$) with a particular padding for the last block if the plaintext length is not a multiple of n. A particular mode of operation is chosen to connect the different blocks encrypted with the block-cipher; as example the mode Cipher Block Chaining (CBC) consists in XORing the

first plaintext block to an Initialization Vector (IV) before applying the block-cipher, then each following plaintext block is XORed to the precedent ciphertext block before applying the block-cipher. Therefore it gives a ciphertext of the same length as the padded plaintext. The decryption algorithm is deduced from the encryption algorithm, as block-ciphers are families of (efficiently invertible) permutations.

Many Block-ciphers are built on the idea of iterated encryption; informally it consists in decomposing the encryption in various similar rounds in order to obtain a secure permutation that can be studied based on simpler building blocks. More formally:

Definition 2.2.13 (Iterated Block-cipher)**.** *We call Iterated Block-cipher a block-cipher E, with:*

- $r \in \mathbb{N}^*$ *rounds,*
- r *round keys* $k_1, \cdots, k_r$ *efficiently derivable from each* sk*,*
- *a round function* F_k *indexed by the round key* k*,*
- *For each key* sk *and each plaintext* m*:*

$$E_{\mathsf{sk}}(m) = F_{k_r} \circ F_{k_{r-1}} \circ \cdots \circ F_{k_1}(m).$$

The separation of an iterated block-cipher in rounds is useful for the implementation and analysis of the construction, as when the round functions are very similar, studying one round and the round keys derivation can be sufficient to derive results on the efficiency or security of the whole scheme. Various families of block-ciphers have been developed and investigated in many works, we presented here only the basic notions used in this thesis.

In the following we present the class of stream ciphers. Informally, they are deterministic algorithms generating a pseudorandom sequence from a secret key and an IV and combining it to the plaintext. We give a more formal definition of stream ciphers (more precisely of synchronous binary stream ciphers):

Definition 2.2.14 (Stream Cipher)**.** *A stream cipher is defined by the following four deterministic efficiently computable algorithms:*

- $\mathsf{Init}(IV, \mathsf{sk})$*. Generates an initial state* st_1*.*
- $\mathsf{Transition}(\mathsf{st}_i)$*. Outputs the next state* st_{i+1}*.*
- $\mathsf{Filter}(\mathsf{st}_i)$*. Generates a keystream bit* s_i*.*
- $\mathsf{Combine}(s_i, m_i)$*. Generates a ciphertext bit* c_i*.*

We decided to have a more "algorithmic" definition of stream-ciphers as we will focus on some of its components. Otherwise, it could be defined in term of a family of functions as for block-ciphers. Note that with this definition the keystream (*i.e.* the sequence $(s_j)_{j\in[r+1,r+m]}$) is independent of the plaintext (apart from its length); and we sometimes refer to it as key and IV dependent part of the cipher. We also refer to additive IV stream cipher when the $\mathsf{Combine}$ algorithm gives the result of the XOR of its two inputs.

Stream ciphers are the core piece of stream cipher encryption, a kind of symmetric encryption where the length of the key sk is generally the security parameter λ, and the IV

length is frequently set to the same quantity. For the key generation algorithm sk is randomly chosen in a set $\mathcal{K}$, which is often the set of all bitstrings of length λ. To encrypt an ℓ-bit plaintext m, an IV is chosen (randomly or not) and the stream cipher is applied in three steps. First, the Init algorithm is evaluated once. Then the Transition algorithm is evaluated r times to update the register, where the fixed integer value r corresponds to the number of transitions evaluated without keystream output, *i.e.* blank rounds. Finally, the last three algorithms are successively evaluated ℓ times, giving an ℓ-bit ciphertext. As the keystream is independent of the plaintext the decryption algorithm is deduced from the encryption, performing the same algorithms with the same inputs except Combine where the role of the plaintext and ciphertext bits are interchanged.

A primitive often used as a building block for stream ciphers is the Linear Feedback Shift Register (LFSR), used to efficiently generate long sequences. Informally, it is a construction with a register of L cells with a particular update process used to produce a sequence of bits. As example we describe it for the case of a binary LFSR in Fibonacci mode. First, the register is initialized with a L length bitstring, it corresponds to the initial state. Then this register is updated at each clock tick by applying a linear function on the current values of the L cells, outputting this result as s_t (the t-th bit of the sequence). The content of each of the $L-1$ last cells is overwritten by the current value of the precedent cell, and the contain of the first cell is set to s_t. Applying iteratively this update gives the sequence $(s_t)_{t\geq 1}$, called the LFSR sequence. More formally we give the following definition of (binary) LFSR:

Definition 2.2.15 (Linear Feedback Shift Register)**.** *A Linear Feedback Shift Register of length L is a finite state automaton which produces a semi-infinite binary sequence $(s_t)_{t\geq 1}$ satisfying a linear recurrence relation of degree L over $\mathbb{F}_2$:*

$$\forall t \in \mathbb{N}^*, \quad s_{t+L} = \sum_{i=1}^{L} c_i s_{t+L-i},$$

where the L binary coefficients c_i are the feedback coefficients of the LFSR.

Another commonly used primitive in the context of stream ciphers is the Nonlinear Feedback Shift Register (NFSR), where the feedback function is nonlinear. These two primitives are often used as building blocks for stream ciphers, where it is common to combine and/or filter various LFSR and NFSR.

We need to discuss the notion of efficiency in the context of symmetric instantiations. When we deal with theoretical (asymptotic or connected to an idealized model) constructions, an efficient algorithm corresponds to a PPT algorithm (independently of the constants). When we mention practical, concrete, or real-life constructions, as symmetric instantiations, an efficient algorithm should be understood as an algorithm with execution time bounded in seconds or minutes on a current desktop computer, and with data resources handleable by the same kind of computer. This concern is naturally extended to the security notion, as for deployed cryptosystems as symmetric encryption schemes, the feasibility of an attacker using todays computer to decrypt a message is a main concern, even if the construction is asymptotically secure. It motivates the next part, dealing with heuristic security.

2.2.2.3 Heuristic Security

For concrete considerations of the security we consider cryptanalyses performed by a real-life attacker on an instantiated scheme with a fixed security parameter. In this case the security is

not expressed in terms of experiment performed by PPT algorithms but in terms of estimations of the best time (or data) complexity over the known attacks in term of elementary machine operation (or storage units). It corresponds to a heuristic approach, as a non investigated attack could break the scheme in less than 2^λ operations, without compromising the security claims concerning the known attacks. Particular interests of this approach are to set concrete parameters for applications (indeed, this approach is necessary for any concrete instantiation), and to investigate the behavior of constructions not relying on well studied assumptions. We introduce here the basic notions of heuristic cryptanalysis used to investigate the security of a concrete instantiation of a stream cipher encryption scheme.

For a symmetric encryption scheme with proposed security level λ bits, the scheme is considered broken if a key recovery attack in less that 2^λ machine operations is exhibited. Different kinds of attacks are possible, depending on the capacities of the attacker and on her target; recovering the key or only distinguishing the encrypted message from a random message. More formally, we consider the following cryptanalysis models:

Definition 2.2.16 (Cryptanalysis Models)**.** *For a symmetric encryption scheme we define four kinds of cryptanalysis models, from the weaker to the stronger attacker:*

- *Ciphertexts Only. The attacker accesses only to the ciphertexts produced during the encryption.*
- *Known Plaintexts. The attacker accesses to pairs of corresponding plaintext-ciphertext.*
- *Chosen Plaintexts. The attacker chooses plaintexts to be encrypted (in an adaptive manner or not).*
- *Chosen Ciphertexts. The attacker chooses ciphertexts to be decrypted (in an adaptive manner or not).*

The cryptanalyses are considered with a fixed key not influenced by the attacker, and the efficiency of a cryptanalysis is measured in terms of time complexity or data complexity.

Definition 2.2.17 (Cryptanalysis Efficiency)**.** *When cryptanalysing an instantiated symmetric encryption scheme we define the following measures for the efficiency of the key recovery attack:*

- *Time Complexity. Denoted C_T, it corresponds to the number of operations necessary to recover the key sk used in the scheme.*
- *Data Complexity. Denoted C_D, it corresponds to the number of ciphertexts needed by the attacker to conduct the attack.*

Note that these definitions can be adapted to attacks aiming at distinguishing the ciphertexts from random bitstrings.

2.2.3 Fully Homomorphic Encryption

In this section we recall the usual definitions of (Fully) Homomorphic Encryption, a particular kind of encryption. It belongs to the class of public key encryption schemes, as introduced in Definition 2.2.10, a large class where the decryption key and the encryption key are different (in opposition to symmetric encryption), and encryption key is public. In order

to differentiate homomorphic encryption schemes from other encryption schemes, as when different encryption schemes are combined, we use the capital letter H, for algorithms or objects relative to the homomorphic encryption scheme.

Definition 2.2.18 (Homomorphic Encryption Scheme)**.** *Let $\mathcal{M}$ be the plaintext space, $\mathcal{C}$ the ciphertext space and λ the security parameter. A homomorphic encryption scheme consists of four* PPT *algorithms:*

- $H.\mathsf{KeyGen}(1^\lambda)$. *Generates a pair $(\mathsf{pk}^H, \mathsf{sk}^H)$ the public and secret keys of the scheme.*
- $H.\mathsf{Enc}(m, \mathsf{pk}^H)$. *From the plaintext $m \in \mathcal{M}$ and the public key, outputs a ciphertext $c \in \mathcal{C}$.*
- $H.\mathsf{Dec}(c, \mathsf{sk}^H)$. *From the ciphertext $c \in \mathcal{C}$ and the secret key, outputs $m' \in \mathcal{M}$.*
- $H.\mathsf{Eval}(f, c_1, \cdots, c_k, \mathsf{pk}^H)$. *With $c_i = H.\mathsf{Enc}(m_i, \mathsf{pk}^H)$ for $1 \leq i \leq k$, outputs a ciphertext $c_f \in \mathcal{C}$.*

Different notions of homomorphic encryption exist, it depends on the set where the function f can be taken, or in practice it depends on which operations are possible. For all these kinds of homomorphic encryptions we assume a compactness property, $|\mathcal{C}|$ is finite, and the size of a ciphertext does not depend on the number of homomorphic operations performed to obtain it. When only one kind of operation is permitted the scheme is simply homomorphic; it is called SomeWhat Homomorphic (SWHE) when more than one operation can be performed, at least partially. We refer to Leveled Homomorphic Encryption (LHE) scheme, when f is restricted to be any polynomial of bounded degree (defining the level) and bounded coefficients, and finally we refer to Fully Homomorphic Encryption (FHE) scheme when f can be any function defined over $\mathcal{M}$.

For these encryption schemes we consider particular correctness and security notions. For the correctness it is linked to the family of functions f and corresponds to overwhelming correctness rather than perfect correctness, as the probabilistic encryption algorithm used in these kind of schemes can lead to $H.\mathsf{Dec}(H.\mathsf{Enc}(m, \mathsf{pk}^H), \mathsf{sk}^H) \neq m$. Formally, it leads to the following definition:

Definition 2.2.19 (Correctness of Homomorphic Encryption Scheme)**.** *A Homomorphic Encryption Scheme achieves the Correctness property for the family of functions $\mathcal{F}$ if for all $f \in \mathcal{F}$ and for all $m_1, \cdots, m_k \in \mathcal{M}$, with $(\mathsf{pk}^H, \mathsf{sk}^H) \leftarrow H.\mathsf{KeyGen}(1^\lambda)$, such that $c_i \leftarrow H.\mathsf{Enc}(m_i, \mathsf{pk}^H)$ for $i \in [k]$ and $c_f \leftarrow H.\mathsf{Eval}(f, c_1, \cdots, c_k, \mathsf{pk}^H)$ are properly generated it holds:*

$$Pr[H.\mathsf{Dec}(c_f, \mathsf{sk}^H) \neq f(m_1, \cdots, m_k)] = negl(\lambda),$$

where the probability is taken over all the randomness in the experiment.

The security notion we consider for these encryption schemes is Indistinguishability under Chosen Plaintext Attack (IND-CPA), informally it relies to the impossibility for an adversary to know which plaintext is encrypted between two plaintexts of her choice. More formally:

Definition 2.2.20 (Indistinguishability under Chosen Plaintext Attack Security)**.** *Let $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme, the advantage of an adversary $\mathcal{A}$ against Indistinguishability under Chosen Plaintext Attack (IND-CPA) is defined by the following experiments (defined for $b = 0$ and $b = 1$):*

$\mathsf{Exp}^{ind\text{-}cpa\text{-}b}(\mathcal{A}, \lambda)$:
$(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$,
$(m_0, m_1) \leftarrow \mathcal{A}(pk)$,
$c^* \leftarrow \mathsf{Enc}(m_b)$,
$b' \leftarrow \mathcal{A}(c^*)$,
return b'.

The encryption scheme is IND-CPA if this advantage is negligible in λ, for any polynomial-time adversary $\mathcal{A}$.

2.3 Lattice-Based Cryptography

Lattice-based cryptography is a wide research topic, relatively to the presented material of this thesis we restrict our preliminaries to notions related to the Learning With Errors (LWE) problem, for a general introduction to Lattice-based cryptography we refer to [Pei15]. We first present basic definitions on lattices, then we introduce the Learning With Errors problem and finally some definitions relatively to Gaussian distributions.

2.3.1 Generalities on Lattices

We give a definition of lattice and of the successive minima of a lattice.

Definition 2.3.1 (Lattice)**.** *A lattice is a discrete additive subgroup of $\mathbb{R}^m$ denoted Λ.*

Let $\mathbf{b}_1, \cdots, \mathbf{b}_n$ be n linearly independent vectors of $\mathbb{R}^m$ and $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_n\}$, we call Lattice generated by $\mathbf{B}$:

$$\mathcal{L}(\mathbf{B}) = \mathsf{Span}_{\mathbb{Z}}(\mathbf{B}) = \mathbf{B} \cdot \mathbb{Z}^n = \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i \ ; \ x_i \in \mathbb{Z} \right\},$$

where:

- $\mathbf{B}$ *is a basis of the lattice,*
- m *is the dimension of the lattice,*
- n *is the rank of the lattice.*

When $m = n$ the lattice is called full rank.

Definition 2.3.2 (Successive Minima of a Lattice)**.** *Let $\Lambda \subseteq \mathbb{R}^m$ be a rank-n Lattice, for $i \in [n]$ we denote $\lambda_i(\Lambda)$ the i-th Successive Minimum of Λ the value:*

$$\lambda_i(\Lambda) = \min_{\in \mathbb{R}^+} \{ r \, ; \, \dim(\mathsf{Span}(\Lambda \cap \bar{B}(0, r))) \geq i \},$$

where $\bar{B}(0, r)$ is the closed ball of radius r centered at 0.

We recall classical problems defined over lattices, defined on full-rank lattices.

Definition 2.3.3 (Shortest Vector Problem (SVP))**.** *Given a basis of an n-dimensional lattice Λ, find a vector $\mathbf{v} \in \Lambda$ such that $||\mathbf{v}|| = \lambda_1(\Lambda)$.*

Definition 2.3.4 (Decisional Approximate Shortest Vector Problem (GapSVP_γ))**.** *Given a basis of an n-dimensional Lattice Λ, determine whether $\lambda_1(\Lambda) \leq 1$ or $\lambda_1(\Lambda) > \gamma(n)$.*

Definition 2.3.5 (Approximate Shortest Independent Vector Problem (SIVP_γ))**.** *Given a basis of an n-dimensional Lattice Λ, find a set $\mathbf{S} = \{\mathbf{s}_i\} \subset \Lambda$ of n linearly independent vectors such that $\forall i \in [n],\ \mathbf{s}_i \leq \gamma(n)\lambda_n(\Lambda)$.*

2.3.2 Learning With Errors

In this section, we recall useful notation and definitions needed about the decisional LWE problem and its ring variation. The decisional Learning With Error problem (dLWE) was introduced by Regev [Reg05].

Definition 2.3.6 (dLWE Assumption)**.** *For an integer $q = q(n) \geq 2$, an adversary $\mathcal{A}$ and an error distribution $\chi = \chi(n)$ over $\mathbb{Z}_q$, we define the following advantage function:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\chi}} := |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{z}_0) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{z}_1) = 1]|,$$

where

$$\mathbf{A} \leftarrow_\$ \mathbb{Z}_q^{n\times m}, \mathbf{s} \leftarrow_\$ \mathbb{Z}_q^n, \mathbf{e} \leftarrow_\$ \chi^m, \mathbf{z}_0 := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \quad \textit{and} \quad \mathbf{z}_1 \leftarrow_\$ \mathbb{Z}_q^m\ .$$

The $\mathsf{dLWE}_{n,m,q,\chi}$ assumption asserts that for all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\chi}}$ is a negligible function in n.

The ring variant was introduced in [SSTX09], and later referred as Ring LWE in [LPR10].

Definition 2.3.7 (dR-LWE Assumption)**.** *For a polynomial ring $R = \mathbb{Z}[X]/f(X)$ with f of degree n, an integer $q \geq 2$, an adversary $\mathcal{A}$ and an error distribution χ over $R_q^\vee$, where $R_q = R/qR$, $R^\vee$ being R dual fractional ideal, we define the following advantage function:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{d}R\mathsf{LWE}_{R,q,\chi,m}} := |\Pr[\mathcal{A}(\mathbf{a}, \mathbf{z}_0) = 1] - \Pr[\mathcal{A}(\mathbf{a}, \mathbf{z}_1) = 1]|,$$

where

$$\mathbf{a} \leftarrow_\$ R_q^m,\ s \leftarrow_\$ R_q^\vee,\ \mathbf{e} \leftarrow_\$ \chi^m,\ \mathbf{z}_0 := s \cdot \mathbf{a} + \mathbf{e} \quad \textit{and} \quad \mathbf{z}_1 \leftarrow_\$ R^m\ .$$

With $f(X)$ a cyclotomic polynomial, the $\mathsf{d}R\mathsf{LWE}_{R,q,\chi,m}$ assumption asserts that for all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{d}R\mathsf{LWE}_{R,q,\chi,m}}$ is a negligible function in n.

The problems associated to these assumptions have nice reductions to older Lattice problems; in [Reg05] Regev proved that an efficient solution to LWE implies a quantum algorithm for GapSVP and SIVP. The reduction have been improved since, and the result generalized for RLWE; making the LWE problem a fruitful assumption to build cryptographic primitives and relying on well-studied assumed-hard problems.

Variants of LWE are sometimes considered, as the Learning Parity With Noise (LPN) problem and the Learning with Rounding (LWR) problem. The first one is a learning problem, predecessor of LWE using only binary values; the second one is a more recent problem [BPR12], partially de-randomized, with reduction from the LWE problem. We present the search versions of these problems.

Definition 2.3.8 (Learning Parity With Noise Problem)**.** *Let* $\mathbf{s} \leftarrow_\$ \{0,1\}^n$*,* $\varepsilon \in (0, 1/2)$ *be a constant noise parameter and Bernoulli(*ε*) be a Bernoulli distribution of parameter* ε*, let* $A_{\mathbf{s},\varepsilon}$ *be the following distribution:*

$$\{\mathbf{a} \leftarrow_\$ \{0,1\}^n;\ \nu \leftarrow_\$ \textit{Bernoulli(}\varepsilon\textit{)};\ (\mathbf{a}, \langle \mathbf{a}, \mathbf{s}\rangle + \nu)\},$$

solving the Learning Parity With Noise Problem consists in finding the vector $\mathbf{s}$ *from independent samples taken from the distribution* $A_{\mathbf{s},\varepsilon}$*.*

Definition 2.3.9 (Learning With Rounding Problem)**.** *Let* $\mathbf{s} \leftarrow_\$ \mathbb{Z}_q^n$*, and let* $\lfloor \cdot \rfloor_p$ *be a rounding defined as:*

$$\lfloor \cdot \rfloor_p : \mathbb{Z}_q \to \mathbb{Z}_p : x \mapsto \lfloor (p/q) \cdot x \rfloor,$$

let $A_{\mathbf{s},q,p}$ *be the following distribution:*

$$\{\mathbf{a} \leftarrow_\$ \mathbb{Z}_q^n;\ (\mathbf{a}, \lfloor \langle \mathbf{a}, \mathbf{s}\rangle \rfloor_p)\},$$

solving the Learning With Rounding Problem consists in finding the vector $\mathbf{s}$ *from independent samples taken from the distribution* $A_{\mathbf{s},\varepsilon}$*.*

2.3.3 Gaussians

For our constructions based on the LWE problem, we will take the distribution χ as a subgaussian random variable which we define hereafter. More details about subgaussian distributions and the lemmata's proof can be found in [AP14].

Definition 2.3.10 (Subgaussian Random Variables)**.** *Let* X *be a random variable. We say* X *is subgaussian with parameter* σ *if there exists* σ *such that:*

$$\forall t \in \mathbb{R}, \mathbb{E}[e^{tX}] \leq e^{\sigma^2 t^2/2},$$

where $\mathbb{E}[e^{tX}]$ *is the moment generating function of* X*.*

Lemma 2.3.11 (Subgaussian Random Variables properties)**.** *Let* X*,* X' *be independent subgaussian random variables of parameter* σ *and* σ' *respectively. Assuming* $\mathbb{E}(X) = \mathbb{E}(X') = 0$ *we have the following properties:*

- *Tails:* $\forall t \geq 0$ *we have* $Pr[|X| \geq t] \leq 2e^{-\pi t^2/\sigma^2}$*.*
- *Homogeneity:* $\forall c \in \mathbb{R}$*,* cX *is subgaussian with parameter* $|c|\sigma$*.*
- *Pythagorean additivity:* $X + X'$ *is subgaussian with parameter* $\sqrt{\sigma^2 + \sigma'^2}$*.*

We extend the notion of subgaussianity to vectors and polynomials. Since the coefficients of a polynomial are seen as a vector, we call subgaussian vector of parameter σ a vector where each coefficient follows an independent subgaussian distribution with parameter σ.

Lemma 2.3.12 (Subgaussian Vector Norm, adapted from [AP14], Lemma 2.1)**.** *Let* $\mathbf{x} \in \mathbb{R}^n$ *be a random vector where each coordinates follows an independent subgaussian distribution of parameter* σ*. Then for some universal constant* $C > 0$ *we have* $Pr\left[||\mathbf{x}||_2 > C\sigma\sqrt{n}\right] \leq 2^{-\Omega(n)}$ *and therefore* $||\mathbf{x}||_2 = \mathcal{O}(\sigma\sqrt{n})$*.*

2.4 Boolean Functions

Boolean functions are a wide research area, we introduce here some core notions of Boolean functions in cryptography. We use the following definition of Boolean function, more restrictive than vectorial Boolean function, as in this thesis we will consider only this type of functions.

Definition 2.4.1 (Boolean Function)**.** *A Boolean function f with n variables is a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$. The set of all Boolean functions in n variables is denoted by $\mathcal{B}_n$.*

We call pseudo-Boolean function a function with input space $\mathbb{F}_2^n$ but output space different to $\mathbb{F}_2$.

As example, considering a pseudo-Boolean functions with output space $\{-1, 1\}$ will be more convenient to use in some proofs.

The following representation is commonly used, and its basic properties also.

Definition 2.4.2 (Algebraic Normal Form (ANF))**.** *We call Algebraic Normal Form of a Boolean function f its n-variable polynomial representation over $\mathbb{F}_2$ (i.e. belonging to $\mathbb{F}_2[x_1, \cdots, x_n]/(x_1^2 + x_1, \cdots, x_n^2 + x_n)$):*

$$f(x) = \sum_{I \subseteq [n]} a_I \left(\prod_{i \in I} x_i \right) = \sum_{I \subseteq [n]} a_I x^I,$$

where $a_I \in \mathbb{F}_2$.

Corollary 2.4.3 (Algebraic Normal Form Properties)**.** *The following properties follow the ANF definition:*

- *The algebraic degree of f equals the global degree $\max_{I:a_I=1} |I|$ of its ANF.*
- *Any term $\prod_{i \in I} x_i$ in such an ANF is called a monomial and its degree equals $|I|$. A function with only one non zero coefficient a_I is called a monomial function.*
- *The function f is affine if and only if its algebraic degree is at most 1; the function is linear if in addition $a_\emptyset = 0$.*

2.4.1 Boolean criteria

In this part, we recall the cryptographic properties of Boolean functions, mostly taken from [Car10], as we refer to for more details on Boolean functions. We define here cryptographic criteria on Boolean functions, whose connection to cryptography will be explained later in this thesis. Through this thesis we refer to "cryptographic criteria on Boolean functions" as "Boolean criteria", or "criteria".

2.4.1.1 Balancedness and Resiliency

Definition 2.4.4 (Balancedness)**.** *A Boolean function $f \in \mathcal{B}_n$ is said to be balanced if its outputs are uniformly distributed over $\{0, 1\}$.*

Definition 2.4.5 (Resiliency)**.** *A Boolean function $f \in \mathcal{B}_n$ is said to be m-resilient if any of its restrictions obtained by fixing at most m of its coordinates is balanced. We denote by $\mathsf{res}(f)$ the maximum resiliency (also called resiliency order) m of f and set $\mathsf{res}(f) = -1$ if f is unbalanced.*

Note that the resiliency is the extended notion of the balancedness, a balanced function is a k-resilient function with $k \geq 0$.

The Fourier transform is an important tool to study the resiliency of a Boolean functions, we define here the discrete Fourier transform (or Hadamard) transform:

Definition 2.4.6 (Discrete Fourier Transform)**.** *The discrete Fourier transform is the linear mapping which maps any pseudo-Boolean function f on $\mathbb{F}_2^n$ (with output space included in $\mathbb{Z}$) to the function $\hat{f}$ defined on $\mathbb{F}_2^n$ as:*

$$\hat{f}(a) = \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{a \cdot x},$$

where $a \cdot x$ denotes the inner product in $\mathbb{F}_2^n$, and the sum is performed in $\mathbb{Z}$.

The discrete Fourier transform can be applied to a Boolean function f itself but also to the sign function $f_\chi(x) = (-1)^{f(x)}$, giving the Walsh transform:

Definition 2.4.7 (Walsh Transform)**.** *Let $f \in \mathcal{B}_n$ a Boolean function, its Walsh transform $\hat{f}_\chi$ at $a \in \mathbb{F}_2^n$ is defined as:*

$$\hat{f}_\chi(a) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+a \cdot x}.$$

Note that the Walsh transform is strongly connected to the balancedness and the resiliency, as shown by the following theorem from [XM88]:

Theorem 2.4.8 ([XM88])**.** *Let $f \in \mathcal{B}_n$, f is m-resilient if and only if $\hat{f}_\chi(a) = 0$ for all $a \in \mathbb{F}_2^n$ such that $\mathsf{wH}(a) \leq m$.*

2.4.1.2 Nonlinearity

Definition 2.4.9 (Nonlinearity)**.** *The nonlinearity NL of a Boolean function $f \in \mathcal{B}_n$, where n is a positive integer, is the minimum Hamming distance between f and all the affine functions in $\mathcal{B}_n$:*

$$\mathsf{NL}(f) = \min_{g, \deg(g) \leq 1} \{d_H(f, g)\},$$

with $d_H(f, g) = \#\{x \in \mathbb{F}_2^n \mid f(x) \neq g(x)\}$ the Hamming distance between f and g; and $g(x) = a \cdot x + \varepsilon$, $a \in \mathbb{F}_2^n, \varepsilon \in \mathbb{F}_2$ (where $\cdot$ is some inner product in $\mathbb{F}_2^n$; any choice of an inner product will give the same definition).

The nonlinearity of a Boolean function can also be defined by its Walsh transform:

$$\mathsf{NL}(f) = 2^{n-1} - \frac{1}{2} \max_{a \in \mathbb{F}_2^n} |\hat{f}_\chi(a)|.$$

The functions maximizing the nonlinearity are called bent functions, they exist for even n only, many families of bent functions are known [McF73], reaching the upper bound $\mathsf{NL}(f) \leq 2^{n-1} - 2^{n/2-1}$. Note that the nonlinearity measures the distance to affine functions, it can be generalized to the notion of higher order nonlinearity of denoted $\mathsf{NL_d}$ where the distance is taken over all functions of degree less than or equal to d.

2.4.1.3 Algebraic Immunity and Fast Algebraic Immunity

Definition 2.4.10 (Algebraic Immunity)**.** *The algebraic immunity of a Boolean function $f \in \mathcal{B}_n$, denoted as $\mathsf{AI}(f)$, is defined as:*

$$\mathsf{AI}(f) = \min_{g \neq 0}\{\mathsf{deg}(g) \mid fg = 0 \text{ or } (f \oplus 1)g = 0\},$$

where $\mathsf{deg}(g)$ is the degree of g. The function g is called an annihilator of f (or $(f \oplus 1)$).

Note that this definition directly leads to the following properties for simple functions:

Corollary 2.4.11 (Algebraic Immunity Properties)**.** *Let f be a Boolean function:*

- *The null function and the all-one function are the only functions such that $\mathsf{AI}(f) = 0$.*
- *All monomial non constant functions f are such that $\mathsf{AI}(f) = 1$.*
- *For all f it holds: $\mathsf{AI}(f) \leq \deg(f)$.*

Definition 2.4.12 (Fast Algebraic Immunity)**.** *The fast algebraic immunity of a Boolean function $f \in \mathcal{B}_n$, denoted as $\mathsf{FAI}(f)$, is defined as:*

$$\mathsf{FAI}(f) = \min\{2\mathsf{AI}(f), \min_{1 \leq \deg(g) < \mathsf{AI}(f)}(\max[\deg(g) + \mathsf{deg}(fg), 3\deg(g)])\}.$$

This definition finished our preliminary presentation of Boolean cryptographic criteria. The high nonlinearity and resiliency properties have to be ensured to thwart correlation attacks [Sie85] and fast correlation attacks [MS88]. The high algebraic immunity and fast algebraic immunity have to be ensured to thwart algebraic attacks [Cou03a].

2.4.2 Constructions of Boolean Functions

In this part we highlight some constructions of Boolean functions, widely used in this thesis. We begin by the secondary construction commonly called direct sum construction.

Definition 2.4.13 (Direct Sum)**.** *Let f be a Boolean function of n variables and g a Boolean function of m variables, f and g depending on distinct variables, the direct sum h of f and g is defined by:*

$$h(x, y) = f(x) + g(y), \qquad \text{where } x \in \mathbb{F}_2^n \text{ and } y \in \mathbb{F}_2^m.$$

A family of functions obtained by direct sums can be of particular interest when looking for functions simple to evaluate; functions obtained by direct sums of monomials. Informally it consists of functions where each variable appears at most once in the ANF, and we focus on the ones where each variable appears once and only once.

Definition 2.4.14 (Direct Sum of Monomials)**.** *Let f be a Boolean function of n variables, we call f a Direct Sum of Monomials a function obtained by direct sum of monomials if the following holds for its ANF:*

$$\forall (I, J) \text{ such that } a_I = a_J = 1,\ I \cap J \in \{\emptyset, I \cup J\}.$$

Definition 2.4.15 (Direct Sum Vector)**.** *Let f be a direct sum of monomials, we define its Direct Sum Vector:*

$$\mathbf{m}_f = [m_1, m_2, \cdots, m_k],$$

of length $k = \deg(f)$, where each m_i is the number of monomials of degree i of f:

$$m_i = |\{a_I = 1 \,|\, |I| = i\}|.$$

When we consider a function F associated to the direct sum vector $\mathbf{m}_F = [m_1, m_2, \cdots, m_k]$ it corresponds to the function with:

- *$M = \sum_{i=1}^{k} m_i$ monomials,*
- *$N = \sum_{i=1}^{k} i m_i$ variables.*

Note that it corresponds to a function without variables not used, each variable appears once and only once in the ANF.

A sub-family of direct sum of monomials is the family of triangular functions.

Definition 2.4.16 (Triangular Functions)**.** *Let k be a strictly positive integer. The k-th Triangular Function T_k is a direct sum of monomials of $k(k+1)/2$ variables:*

$$T_k(x_1, \cdots, x_{k(k+1)/2}) = \sum_{i=1}^{k} \prod_{j=1}^{i} x_{j+i(i-1)/2}.$$

It can also be defined from its direct sum vector which is the all-1 vector of length k: $\mathbf{m}_{T_k} = [1, 1, \cdots, 1]$.

Another sub-family of direct sum of monomials is of particular interest in this thesis, called the family of FLIP functions.

Definition 2.4.17 (FLIP Functions)**.** *Let n_1, n_2, nb and h be positive integers, we call FLIP function the direct sum of monomials defined as:*

$$F(x) = L_{n_1}(x) + Q_{n_2/2}(x) + {}^{nb}\Delta^h,$$

where:

- *F is an $(n_1 + n_2 + n_3)$-variable Boolean function, with $n_3 = nb(h(h+1)/2)$,*
- *L_{n_1} is called the linear part and is defined as:*

$$L_{n_1}(x) = x_1 + x_2 + \cdots + x_{n_1},$$

- *$Q_{n_2/2}$ is called the quadratic part and defined as:*

$$Q_{n_2/2}(x) = x_{n_1+1}x_{n_1+2} + x_{n_1+3}x_{n_1+4} + \cdots + x_{n_1+n_2-1}x_{n_1+n_2},$$

- ${}^{nb}\Delta^h$ *is called the triangular part and defined as:*

$$ {}^{nb}\Delta^h(x) = \sum_{1}^{nb} T_h(x) = \sum_{i=1}^{nb} \left(\sum_{j=1}^{h} \prod_{k=1}^{j} x_{n_1+n_2+\ell} \right), $$

where $\ell = (i-1)(h(h+1)/2) + k + j(j-1)/2$.

It can also be defined from its direct sum vector of length h*:*

$$ \mathbf{m}_F = [n_1 + nb, n_2/2 + nb, nb, \cdots, nb]. $$

We finish these preliminaries on Boolean functions by defining the majority functions, sometimes considered in the area of Boolean functions used in cryptography.

Definition 2.4.18 (Majority Function)**.** *For any positive odd integer* N *we define the Boolean function* Maj_N *as:*

$$ \forall x = (x_1, \cdots, x_N) \in \mathbb{F}_2^N, \quad Maj_N(x) = \begin{cases} 0 & \text{if } \mathsf{w_H}(x) \leq \lfloor \frac{N}{2} \rfloor, \\ 1 & \text{otherwise.} \end{cases} $$

2.5 Additional Preliminaries

2.5.1 Circuits

We introduce here the basic vocabulary relative to circuits.

Definition 2.5.1 (Boolean Circuit)**.** *We define a Boolean Circuit with* n *input bits as a finite directed acyclic graph in which every node (usually called gates in this context) is either an input node of in-degree* 0 *labeled by one of the* n *input bits, an* AND *gate, a* XOR *gate, or a* NOT *gate. One node is designated as the output node.*

We call:

- *size the number of Boolean gates (AND, XOR, NOT) of the circuit,*
- *depth of the circuit the maximal length from an input node to the output node,*
- *multiplicative depth of the circuit the maximal number of* AND *gates from an input node to the output node.*

More generally we will refer to circuits for finite directed acyclic graph with nodes being gates. These gates are the usual Boolean functions defined as $AND(x_1, x_2) = x_1x_2$, $XOR(x_1, x_2) = x_1 + x_2$, $NOT(x_1) = 1 + x_1$. We additionally describe the so-called MUX gate (named after multiplexer).

Definition 2.5.2 (MUX Gate)**.** *We define the Boolean function used in Boolean circuit* MUX *gate as:*

$$ MUX(x_1, x_2, x_3) = x_1x_2 + x_1x_3 + x_2, $$

where x_1 *is called the control bit as if it equals* 0 *the gate's output is the value of* x_2*, otherwise it equals* 1 *and the gate's output is the value of* x_3*.*

We also give the basic vocabulary related to branching programs.

Definition 2.5.3 (Branching Program)**.** *We define a Branching Program on the variable set $\{x_1, \cdots, x_n\}$ as a finite directed acyclic connected graph with ℓ layers. The first layer consists of one source node and the last layer consists of sink nodes partitioned in two sets (corresponding to the output value 0 or 1). Each non-sink node is labeled by a variable, common to its layer, and has two outgoing edges labeled 0 and 1 respectively. The output of the Branching program on an input is the value of the sink node where ends the path obtained by taking the edges corresponding to the input value of the variable indexing each layer.*

We call:

- *length of the program the maximum length of any path from the source node to a sink node,*
- *width of the program the maximum number of nodes in a layer.*

2.5.2 Binomial Coefficients

We recall here some basic relations on binomial coefficients used several times in this thesis.

Definition 2.5.4 (Binomial Coefficient)**.** *Let n be a positive integer and k an integer such that $0 \leq k \leq n$, we call binomial coefficient n choose k denoted $\binom{n}{k}$ the integer defined as:*

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}, \qquad \text{where } n! = \prod_{i=1}^{n} i.$$

It corresponds to the number of ways to choose a subset of k elements, disregarding their order, from a set of n elements.

Proposition 2.5.5 (Basic Properties on Binomial Coefficients)**.** *Let n be a positive integer and k an integer such that $0 \leq k \leq n$, the following holds:*

- *Pascal Identity.*
$$\binom{n}{k} + \binom{n}{k+1} = \binom{n+1}{k+1}.$$
- *Symmetry.*
$$\binom{n}{k} = \binom{n}{n-k}.$$
- *Sum.*
$$\sum_{i=0}^{n} \binom{n}{k} = 2^n.$$
- *Vandermonde Convolution. Let m be a positive integer:*
$$\sum_{i=0}^{n} \binom{n}{i}\binom{m}{k-i} = \binom{n+m}{k}.$$
- *Parity, (sub-case of Lucas's Theorem). Considering a bitstring representation of n and k ($\mathbf{n}$ and $\mathbf{k}$) of length r where $r = \lceil \log(2) \rceil$:*
$$\binom{n}{k} \equiv 1 \mod 2 \Leftrightarrow \exists i \in [r+1] \mid k_i > n_i.$$

It finishes our preliminaries, letting place to the core chapters of this thesis.

Chapter 3

Fully Homomorphic Encryption

In this part we explain fully homomorphic encryption principles and how this theoretical construction can be converted into a practical primitive for real-life applications. Despite a theoretic feasibility, the road from FHE to efficient computation delegation is quite long. In this chapter we describe the important steps and how to overcome the bottlenecks towards efficiency.

We firstly recall some historic of FHE and principal techniques, then we focus on particular encryption schemes; their mechanism, security, and potential efficiency. After identifying the main bottleneck for the real-world applications, we amplify our study on the error-growth for the considered schemes. Finally, we describe hybrid frameworks, and new protocols that lead to practical constructions for outsourcing computation.

Contents

The first consideration we have in mind when we are looking for practical homomorphic encryption frameworks is the practicability of fully homomorphic encryption by itself. Studying the requirements of this primitive shows various barriers against efficiency. However they can be circumvented or reduced for the applications we consider such as outsourcing computations. Some of the bottlenecks of FHE may never be overcome, as the important size of the ciphertexts or the time complexity of some algorithms. Nevertheless, a better understanding of this kind of encryption and its use jointly with other primitives, can lead to efficient outsourcing computation frameworks. Therefore, when we refer to practical homomorphic encryption frameworks, we consider frameworks using homomorphic encryption and other primitives, particularly hybrid frameworks using an FHE scheme and a symmetric encryption scheme (more details are given in the last part of this chapter). The goal of the current section is to handle the homomorphic part of such frameworks; the principal steps towards efficiency depend on two main points. The first one is the understanding of what can be efficiently done by a homomorphic encryption scheme. The second one is how compatible a homomorphic encryption scheme and the symmetric encryption scheme can be in a hybrid framework. Answering these two main questions enables us to design efficient frameworks, assuming that the symmetric encryption scheme is efficient (executable quickly, and using limited memory resources).

In order to handle the homomorphic part of practical homomorphic encryption frameworks, we proceed in successive steps. We begin by presenting the first constructions of fully homomorphic encryption. The objective of this part is to explain how fully homomorphic encryption is possible, and to present the main components of FHE. These descriptions illustrate the limitations towards efficiency, the minimal complexity involved in homomorphic encryption. It shows that some operations are more costly (homomorphic multiplication) and some algorithms are not practical enough (bootstrapping). Then, based on these considerations, we further investigate what can be efficient enough in the homomorphic schemes to be used in a more general framework. It consists in examining more specifically one kind of FHE schemes and its adequateness with some functions. To do so, we focus on one family of homomorphic schemes, and examine its efficiency using the notion of error-growth. We perform a detailed study of the error-growth involved in the homomorphic operations, and we investigate families of functions related to very low error-growth. The objective of this part is to identify the efficient homomorphic operations and how to design functions or circuits that allow homomorphic evaluation with a reasonable execution time. Finally, these efficient homomorphic blocks are considered in a hybrid framework designed for outsourcing computation. We describe this environment and explain how handling the homomorphic part enables to reach a framework where the latency for Alice depends on the amount, and on the complexity of the computations she asks rather than on the usual FHE overheads.

Most of this chapter is adapted from the article [MJSC16]. Other origins are specified by the relevant citations, except Section 3.3.3.2 (and Section 3.3.2.3) which develops additional personal results (not published yet).

3.1 First and Second Generations

In this section, we present the first constructions of fully homomorphic encryption and we introduce the main techniques of FHE. These schemes illustrate the common difficulties of homomorphic encryption and highlight the bottlenecks we want to overcome in efficient

frameworks. More particularly, it shows that homomorphic operations have a minimal unavoidable cost, some algorithms are intrinsically difficult to evaluate efficiently, and it leads to consider solutions emerging from a better management of the homomorphic operations.

3.1.1 First FHE: Gentry's Thesis

First suggested in 1978 by Rivest, Adleman and Dertouzos [RAD78], fully homomorphic encryption was presumed out of reach until Gentry's breakthrough in 2009. This kind of encryption, enabling unlimited computation on encrypted data, was considered as a "never-to-be-found Holy Grail of cryptography" [Mic10]. It seemed impossible to build ciphertexts with enough structure to allow computation over the ciphertexts but without too much structure to ensure security. Gentry used some key ideas to circumvent this barrier, we succinctly explain these key ideas in the following.

First, let us informally describe his encryption scheme published at STOC 2009 [Gen09]; a formal description and more details can be found in his article or in [GH10]. Here we do not present a technical description of the scheme, as we want to give insights on the core properties enabling FHE. Therefore we invite the reader to forget about the technicality of (fractional) ideal lattices, and to focus on the general picture. To do so, we describe Gentry's scheme using the vocabulary of lattice-based constructions, but we neglect the mathematical objects to focus on the underlying ideas. This scheme is a public-key encryption scheme that can be seen as a GGH-type scheme [GGH97] over ideal lattices in a polynomial ring R. In the ring R two ideals I and J are chosen, the public key is a bad basis of the ideal lattice J whereas the secret key is a good basis for this ideal. To encrypt a 1-bit plaintext, the plaintext is embedded in R/I and added to an error term sampled from the ideal I, the sum is then reduced using the public basis of J; giving the ciphertext. If the error is low enough, the secret basis enables to identify the fractional part of the ciphertext, and then the error and the plaintext can be recovered. If the error is too big, the fractional part cannot be identified, then the plaintext cannot be recovered. The homomorphic operations are straightforward: the homomorphic addition is obtained by adding two ciphertexts, the homomorphic multiplication is obtained by multiplying two ciphertexts. The simplicity of these homomorphic operations comes from the expression of a ciphertext as a plaintext part and an error part; then, for the homomorphic addition the two parts can be treated separately. For the homomorphic product, the mixed terms between plaintext and error belong to the ideal I and they are therefore part of the error, still giving a decomposition as plaintext part and error part. This decomposition is therefore an important underlying idea; which first enables to preserve a structure across the homomorphic operations. Second, beyond the fractional ideal structure, the magnitude of the error part determines the feasibility of decryption.

This overview shows an encryption scheme which allows to perform some operations on the ciphertexts. When published, this scheme was not standard; the security being based on problems on ideal lattices and polynomial rings rather than well established problem of number theoretic algebra. One of the key ideas for fully homomorphic encryption is to use noise-based cryptography, or equivalently to use the notion of error as in this scheme. Note that this idea was already present when the firsts public-key encryptions schemes appeared, as in [BMT78]. With this kind of cryptography, the mathematical structure used in the scheme is borrowed by some noise, by some error term, guarantying security. This provides some plasticity usable for homomorphic encryption; manipulating noisy ciphertexts has an impact

on the error, and when the error-growth is compatible with the desired computation over the encrypted data, it enables somewhat homomorphic encryption (as defined in Section 2.2.3).

In Gentry's scheme, the noise doubles for an addition and it is squared for a product. A ciphertext of this scheme can be decrypted as long as the error magnitude is lower than a certain threshold after which we cannot identify with certainty the decomposition enabling to recover the plaintext. This boundary limits this scheme to evaluate a fixed amount of computation, as it is the case for all somewhat homomorphic encryption scheme. The revolutionary idea of Gentry was to extend the homomorphic property from a somewhat homomorphic encryption scheme to a fully homomorphic encryption scheme using the so-called *bootstrapping* algorithm.

After some operations, a ciphertext has a noise close to the decryption threshold, if this ciphertext can be publicly converted in a ciphertext of the same plaintext with a smaller noise, then the encryption scheme becomes fully homomorphic. The method consists in homomorphically evaluating the decryption circuit of the encryption scheme, which gives a low-noise ciphertext if the decryption circuit is simple enough. Given a bootstrapping key, *i.e.* an encryption of the secret key of the scheme, a ciphertext can be publicly manipulated to obtain a ciphertext, referred as refreshed ciphertext, of the same plaintext, with smaller noise. This step obliges to assume the hypothesis of circular security, as a secret key encrypted by its public key is given as a public parameter. Therefore, bootstrapping was the missing stone to theoretically reach fully homomorphic encryption, but the road is still long to reach fully homomorphic encryption usable in practice.

In this scheme, the decryption function is not simple enough to provide ciphertexts with small noise. Then another technique, called squashing, is used: the decryption circuit is replaced by a simpler one (implying for the security to additionally assume the difficulty of the sparse subset sum problem, a special case of the well-known knapsack problem). After this breakthrough the next steps for fully homomorphic encryption are to build somewhat homomorphic encryption schemes with good properties of the decryption circuit, to use more convenient assumptions as learning with errors and to optimize the schemes to approach a practical use.

3.1.2 Second Generation

Lots of fully homomorphic encryption schemes following Gentry's idea appeared in the next years as [SV10; BGV12; BV11; GH11; Bra12], these works improve the security or the simplicity of the construction, leading to the new notion of leveled homomorphic encryption (see Section 2.2.3 for the various notions of homomorphic encryption). For a leveled homomorphic encryption scheme there is a parameter L in input of the *Setup* algorithm, it is the maximal depth of a circuit (the depth is the maximal length of a path from an input gate to the output gate) that we want to homomorphically evaluate, and the scheme enables to evaluate all circuits of depth inferior or equal to L. This notion enables to compare the various schemes, its efficiency relatively to a particular depth, and to improve the practicability of the constructions. The previously listed schemes are generally considered as the *second generation* as the error-growth mostly depends on the multiplicative depth of the evaluated circuit and not the multiplicative size as for Gentry's scheme. This list contains schemes which security is related to the LWE problem, another large family is the one of fully encryption schemes on the integers as [DGHV09; CNT12; CCK+13; CLT14; CS15]. For these schemes which security is linked to the approximate greatest common divisor problem,

the same concept of second or third generation can be used.

We present here some techniques developed in [BV11], as presented in [BGV12], they are examples of the improvements necessary on the path to practical homomorphic encryption. Some of these techniques are relevant for more recent homomorphic schemes and they witness some difficulties to improve the practicability of homomorphic encryption. This scheme is based on LWE, an easier assumption to use than the one in Gentry's scheme. Here we focus on important techniques for bootstrapping; modulus switching, and key switching.

First, we informally describe a version of Regev's public key cryptosystem:

- secret key: $\mathbf{s} = (1, \mathbf{t}) \in \mathbb{Z}_q^{n+1}$,
- public key : $\mathbf{A} = [\mathbf{b} \| -\mathbf{B}] \in \mathbb{Z}_q^{m\times(n+1)}$ with $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{m\times n}$, $\mathbf{b} = \mathbf{B}\mathbf{t} + 2\mathbf{e}$, and $\mathbf{e}$ from an error distribution over $\mathbb{Z}_q^n$,
- Enc: $\mu \in \{0,1\}; \mathbf{c} = (\mu, \mathbf{0}) + \mathbf{A}^\top \mathbf{r}$ with $\mathbf{r} \leftarrow_{\$} \{0,1\}^m$,
- Dec: $\mu = [[\langle \mathbf{c}, \mathbf{s}\rangle]_q]_2$, where $\langle\ ,\ \rangle$ denotes a scalar product and $[\ ]_k$ a modular reduction.

Note that $\mathbf{A}\mathbf{s} = 2\mathbf{e}$ by construction, then decryption is meaningful: the first coordinate of $\mathbf{s}$ being 1, the inner product is the sum of μ with an inner product between $\mathbf{r}^\top\mathbf{A}$ and $\mathbf{s}$. The resulting term being small due to the choices of $\mathbf{r}$ and $\mathbf{e}$, the modular reductions give μ. This construction based on LWE, or equivalently on RLWE (defined in Section 2.3.2), is more plastic: it is possible to encode the plaintext bit in the lower bit or in the upper bit. Various moduli and the inner product enable to use different techniques to bound the error-growth and manage bootstrapping.

In this scheme, two algorithms are used for the key switching technique: $\mathsf{BitDecomp}$ and $\mathsf{PowerOf2}$, using the binary expression. Let $\mathbf{c} \in \mathbb{Z}_q^n$, we define $\mathsf{BitDecomp}(\mathbf{c})$ and $\mathsf{PowerOf2}(\mathbf{c})$ as vectors of length $n\lceil \log q \rceil$ with coefficients in $\mathbb{Z}_q$. $\mathsf{BitDecomp}(\mathbf{c})$ is the concatenation of the binary decomposition of all coefficients of $\mathbf{c}$. Instead, $\mathsf{PowerOf2}(\mathbf{c})$ is the concatenation of the coefficients of $\mathbf{c}$ multiplied by the successive powers of 2 up to $\lceil \log q \rceil$. Combining these algorithms gives the equality:

$$\langle \mathbf{c}, \mathbf{s}\rangle = \langle \mathsf{BitDecomp}(\mathbf{c}), \mathsf{PowerOf2}(\mathbf{s})\rangle \mod q.$$

The key switching technique from $\mathbf{s}_1$ to $\mathbf{s}_2$ consists then in generating a public key $\mathbf{A}$ for the key $\mathbf{s}_2$ and in adding at the first column $\mathsf{PowerOf2}(\mathbf{s}_1)$; this matrix $\mathbf{B}$ is the one of the key switching. We get the cipher $\mathbf{c}_2$, relatively to the key $\mathbf{s}_2$, of the same plaintext as $\mathbf{c}_1$ relatively to the key $\mathbf{s}_1$, by computing $\mathbf{c}_2 = (\mathsf{BitDecomp}(\mathbf{c}_1))^\top * \mathbf{B}$:

$$\begin{aligned}\langle \mathbf{c}_2, \mathbf{s}_2\rangle &= (\mathsf{BitDecomp}(\mathbf{c}_1))^\top * \mathbf{B} * \mathbf{s}_2 \\ &= (\mathsf{BitDecomp}(\mathbf{c}_1))^\top * \mathbf{A} * \mathbf{s}_2 + \langle \mathsf{BitDecomp}(\mathbf{c}_1), \mathsf{PowerOf2}(\mathbf{s}_1)\rangle \\ &= 2\langle \mathsf{BitDecomp}(\mathbf{c}_1), \mathbf{e}_2\rangle + \langle \mathbf{c}_1, \mathbf{s}_1\rangle.\end{aligned}$$

The left term depends on a sub-sum of $\mathbf{e}_2$ coordinates which is small, then we get:

$$m = [[\langle \mathbf{c}_2, \mathbf{s}_2\rangle]_q]_2 = [[\langle \mathbf{c}_1, \mathbf{s}_1\rangle]_q]_2.$$

The modulus switching technique relies on an algorithm $\mathsf{Scale}(\mathbf{c}, q, p, r)$ which outputs the vector $\mathbf{c}'$ with coefficients in $\mathbb{Z}_p$ closest to $\frac{p}{q}\mathbf{c}$ such that $\mathbf{c} \equiv \mathbf{c}' \mod r$. For all $\mathbf{s}$ fulfilling

a norm restriction on $\|\langle \mathbf{c}, \mathbf{s} \rangle\|_q$ (see [BGV12] for the exact analysis), if $p \equiv q \mod r$ and $\mathbf{c}' = \mathsf{Scale}(\mathbf{c}, q, p, r)$ then

$$m = [[\langle \mathbf{c}, \mathbf{s} \rangle]_q]_r = [[\langle \mathbf{c}', \mathbf{s} \rangle]_p]_r.$$

These two techniques enable to obtain more efficient FHE schemes; the modulus switching enables to reduce the error magnitude (which is a major concern in SWHE schemes) even if the ratio error/modulus is not improved a lot. An appropriate reduction scale enables to keep an error under a particular bound whereas usual multiplication would blow it up. The key switching enables to treat in an easier way the homomorphic product; under the same key the product of two ciphertexts can be seen as a ciphertext under the tensor product $\mathbf{s} \otimes \mathbf{s}$, instead here we get a new cipher under a key of smaller size, reusable for the scheme. The use of binary decomposition has become a usual technique for fully homomorphic encryption; practical for noise management, as presented later in this chapter.

Another technique presented in this article is the so-called *batching*, first appeared in [SV11], which consists in encrypting various bits in the same ciphertext, using the version of the scheme based on RLWE with a modulus p such that $x^d + 1$ totally splits, enabling to apply the Chinese remainders theorem. The batching is one of the techniques giving the possibility of encrypting elements of $\mathbb{Z}_p$ or other rings rather than encrypting only one bit by ciphertext. These techniques are very important for the efficiency of Homomorphic Encryption (HE) schemes because homomorphic properties lead to use schemes with larger key and ciphertext sizes compared to classic public key schemes and these techniques amortize this cost. Lots of batching techniques have been developed since the second generation, in order to optimize the delays of homomorphic evaluation or improve the quantity of data treated in the same time.

All the techniques developed for the second generation led to a variety of schemes relying on classic assumptions, with smaller key and ciphertext sizes, with a better management of the error. Even after a lot of improvements, the bootstrapping technique still involves a high time and data cost. Therefore, practicability of homomorphic encryption relies on a well managed error-growth to limit the number of bootstrappings or to bootstrap in a particular way. For the second generation, it gives a quite simple metric to determine how efficiently a computation can be homomorphically evaluated; it mostly depends on the multiplicative depth of the computed circuit. Indeed, these schemes contain homomorphic addition and homomorphic product, the second involving much more error than the first one leading to an error of the computed ciphertext proportional to the multiplicative depth of the circuit.

3.2 Third Generation

In 2013, Gentry, Sahai and Waters [GSW13] introduced a homomorphic encryption scheme based on LWE using a new technique stemming from the so-called *approximate eigenvector problem*. This new technique leads to a new family of FHE, called 3rd-generation FHE, consisting in HE schemes such that the homomorphic product, main issue for the previous schemes, is easier to treat. Let consider an informal public key cryptosystem based on LWE; m is a plaintext in $\mathbb{Z}_q$, the approximate eigenvector technique consists in building a ciphertext $\mathbf{C}$ such that the secret key $\mathbf{s}$ is almost an eigenvector of $\mathbf{C}$ with associated eigenvalue m:

$$\mathbf{C} * \mathbf{s} = \mathbf{e} + m * \mathbf{s} \mod q.$$

We focus on the impact of this construction on the homomorphic product:

$$\begin{aligned}\mathbf{C}_1 * \mathbf{C}_2 * \mathbf{s} &= \mathbf{C}_1(\mathbf{e}_2 + m_2\mathbf{s}) \\ &= \mathbf{C}_1 * \mathbf{e}_2 + m_2\mathbf{C}_1 * \mathbf{s} \\ &= \mathbf{C}_1 * \mathbf{e}_2 + m_2 m_1 \mathbf{s} + m_2\mathbf{e}_1 \\ &= m_2 m_1 \mathbf{s} + \mathbf{e}'\end{aligned}$$

The resulting error is interesting over various aspects. First, this error is asymmetric: the error term changes depending on the order of multiplication of the ciphertexts. This is primordial when we compute a product between ciphertexts with errors of different magnitudes. Then, the two parts of the error have different consequences on the efficiency of the scheme. The term $m_2\mathbf{e}_1$ depends on the plaintext space: if m_2 is a bit then this term is canceled or only equal to the error of the first ciphertext. Nevertheless if the plaintext space is bigger, as example $\mathbb{Z}_p$, then this error term can be multiplied by p, giving a worst case being exponential in the plaintext space size. The other term is $\mathbf{C}_1 * \mathbf{e}_2$, if $\mathbf{C}_1$ were a random matrix it would give an important error but a formal scheme guarantees a small norm by using a function on $\mathbf{C}_1$ to conserve the multiplicative property, and by reducing the norm of $\mathbf{C}_1 * \mathbf{e}_2$. The bit decomposition techniques or relatives enable to refresh $\mathbf{C}_1$ and then to obtain a quasi-additive error for the homomorphic product.

Due to the particular error-growth of the third generation schemes and their novelty we decide to mostly focus on these constructions in the rest of this thesis, rather than all kinds of fully homomorphic encryption schemes. First, their novelty and the better understanding of the underlying mathematical property (approximate eigenvector), make us hope that the time and space complexity of the schemes can be improved. Then, we study theoretic tools to reach practicability, identifying the bottlenecks in the process from FHE to outsourcing computations, therefore we develop solutions that can be adapted to any kind of FHE we could use. The focus on the third generation enables to concretely study the problems arising when looking for real-life instantiations, finding the solutions and extend them to all noise-based HE constructions. More particularly, the understanding of the error-growth, the metric of a HE scheme is predominant. Controlling the error-growth is the cornerstone to design efficient primitives, fast and using limited memory, to make possible the functionalities desired in each client-server frameworks. Then, our tools are generic and can be adapted to future improvements on homomorphic operation timing, space reduction, new batching, better bootstrapping, *etc.*

Hereafter, we present two schemes belonging to this generation, the first one with security based on LWE (assuming circular security), and the second one based on RLWE. As we modified these schemes from the original works introducing them in order to fit better in our context, we give the correctness and security proofs of our adapted schemes.

3.2.1 Batched GSW

This scheme is a batched version of GSW presented in [HAO15], enabling to pack independently r plaintexts in one ciphertext. We described this particular version because it enables to treat different plaintexts in parallel, which can be useful for many cases of outsourcing computations, as applying the same function on parallel sets of data. This version is more efficient in data than encrypting r bits as r standard GSW ciphertexts, and therefore decreases the expansion factor of this homomorphic encryption scheme. From the security parameter

λ and the considered applications, we can derive the parameters n, q, r, χ of the scheme described below.

- $H.\mathsf{KeyGen}(n, q, r, \chi)$. From the following inputs: the lattice dimension n, the modulus q, the number of bits by ciphertext r and the error distribution χ:
  - Set $\ell = \lceil \log q \rceil$, $m = \mathcal{O}(n\ell)$, $N = (r+n)\ell$, $\mathcal{M} = \{0,1\}^r$ and $\mathcal{C} = \mathbb{Z}_q^{(r+n)\times N}$.
  - Pick $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{n\times m}$, $\mathbf{S}' \leftarrow_{\$} \chi^{r\times n}$ and $\mathbf{E} \leftarrow_{\$} \chi^{r\times m}$.
  - Set $\mathbf{S} = [\mathbf{I}| - \mathbf{S}'] \in \mathbb{Z}_q^{r\times(r+n)}$ and $\mathbf{B} = \left(\begin{array}{c} \mathbf{S}'\mathbf{A} + \mathbf{E} \\ \hline \mathbf{A} \end{array}\right) \in \mathbb{Z}_q^{(r+n)\times m}$.
  - For all $\mathbf{m} \in \{0,1\}^r$:
    - Pick $\mathbf{R_m} \leftarrow_{\$} \{0,1\}^{m\times N}$.
    - Set $\mathbf{P_m} = \left(\mathbf{BR_m} + \left(\begin{array}{c} m_1 \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_r \cdot \mathbf{s}_r^\top \\ \hline \mathbf{0} \end{array}\right) \mathbf{G} \right) \in \mathbb{Z}_q^{(r+n)\times N}$.

    with $\mathbf{s}_i^\top$ the i-th row of $\mathbf{S}$ and $\mathbf{G} = (2^0, \cdots, 2^{\ell-1})^\top \otimes \mathbf{I} \in \mathbb{Z}_q^{(r+n)\times N}$.
  - Output $\mathsf{pk}^H := (\{\mathbf{P_m}\}_{m\in\{0,1\}^r}, \mathbf{B})$ and $\mathsf{sk}^H := \mathbf{S}$.

- $H.\mathsf{Enc}(\mathsf{pk}^H, \mathbf{m})$. Using as inputs $\mathsf{pk}^H = (\{\mathbf{P_m}\}, \mathbf{B})$, and $\mathbf{m} \in \{0,1\}^r$:
  - Pick $\mathbf{R} \leftarrow_{\$} \{0,1\}^{m\times N}$, and output $\mathbf{C} = [\mathbf{BR} + \mathbf{P_m}]_q \in \mathbb{Z}_q^{(r+n)\times N}$.

- $H.\mathsf{Dec}(\mathbf{C}, \mathsf{sk}^H)$. Using as inputs the secret key sk^H, and a ciphertext $\mathbf{C}$,
  - For all $i \in [r]$: $m'_i = \lfloor [\langle \mathbf{s}_i^\top, \mathbf{c}_{i\ell} \rangle]_q \rceil_2$ where $\mathbf{c}_{il}$ is the column of index $i\ell$ of $\mathbf{C}$ and the rounding $\lfloor [a]_q \rceil_2$, is a function from $a \in \mathbb{Z}_q$ to $\{0,1\}$ giving 1 if $\lfloor \frac{q}{4} \rfloor \leq a \leq \lfloor \frac{3q}{4} \rfloor$ and 0 otherwise.
  - Output $(m'_1, \cdots, m'_r) \in \{0,1\}^r$.

  Note that $\mathbf{SC} = \mathbf{SBR} + \mathbf{SP_m} = \mathbf{ER} + \mathbf{ER_m} + \left(\begin{array}{c} m_1 \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_r \cdot \mathbf{s}_r^\top \end{array}\right) \mathbf{G} = \mathbf{E}' + \left(\begin{array}{c} m_1 \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_r \cdot \mathbf{s}_r^\top \end{array}\right) \mathbf{G}$.

- The $H.\mathsf{Eval}$ algorithm finally consists in iterating, following a circuit f, the homomorphic operations $H.\mathsf{Add}$ and $H.\mathsf{Mul}$:
  - $H.\mathsf{Add}(\mathbf{C}_1, \mathbf{C}_2) : \mathbf{C}_+ = \mathbf{C}_1 + \mathbf{C}_2$.
  - $H.\mathsf{Mul}(\mathbf{C}_1, \mathbf{C}_2) : \mathbf{C}_\times = \mathbf{C}_1 \times \mathbf{G}^{-1}\mathbf{C}_2$ with $\mathbf{G}^{-1}$ a function such that $\forall \mathbf{C} \in \mathbb{Z}_q^{(r+n)\times N}, \mathbf{G}\mathbf{G}^{-1}(\mathbf{C}) = \mathbf{C}$ and the values of $\mathbf{G}^{-1}(\mathbf{C})$ follow a subgaussian distribution with parameter $\mathcal{O}(1)$ ($\mathbf{G}^{-1}$ uses a bit decomposition technique, see [MP12] for the existence and correctness proof of $\mathbf{G}^{-1}$).

Remark 3.2.1. *For practical use, we only need to store $r+1$ matrices $\mathbf{P}_m$, namely the $r+1$ ones with $\mathbf{m}$ of Hamming weight equal to 0 or 1 are sufficient to generate correct encryption of all $\mathbf{m} \in \{0,1\}^r$ with at most r additions of the corresponding $\mathbf{P}_m$ matrices.*

Lemma 3.2.2 (Correctness of Batch GSW scheme.)**.** *For every* $(\mathsf{pk}^H, \mathsf{sk}^H) \leftarrow H.\mathsf{KeyGen}(n, q, r, \chi)$, $\mathbf{m} \in \{0,1\}^r$, *and* $\mathbf{C} \leftarrow H.\mathsf{Enc}(\mathsf{pk}^H, \mathbf{m})$, *(respectively* $\mathbf{C}_f \leftarrow H.\mathsf{Eval}(f, \mathbf{C}_1, \cdots, \mathbf{C}_k, \mathsf{pk}^H)$*) such that for all* $i \in [r], |\mathbf{s}_i^\top \mathbf{c}_{i\ell} - m_i 2^{\ell-1} \mod q| < 2^{\ell-2}$ *(where* $x \mod q \in [-q/2+1, q/2]$*) we have* $\mathbf{m} = H.\mathsf{Dec}(\mathbf{C}, \mathsf{sk}^H)$ *(respectively* $f(\mathbf{m}_1, \cdots, \mathbf{m}_k) \mod 2 = H.\mathsf{Dec}(\mathbf{C}_f, \mathsf{sk}^H)$*).*

Proof. With the expression of $\mathbf{SC}$ for all $i \in [r]$; $\mathbf{s}_i^\top \mathbf{c}_{i\ell}$ can be written as:

$$(\mathbf{SC})_{i,i\ell} = e'_{i,i\ell} + 2^{\ell-1} m_i \mod q.$$

Then, if $|e'_{i,i\ell} \mod q| < 2^{\ell-2}$, the rounding in the decryption algorithm outputs m_i. If the inequality is correct for all $i \in [r]$, then $\mathbf{m} = H.\mathsf{Dec}(\mathbf{C}, \mathsf{sk}^H)$. □

Note that a sufficient condition for correctness is to ensure $|e'_{i,j}| < 2^{\ell-2}$ for all pairs (i, j) in $[r] \times [N]$.

Lemma 3.2.3 (Security of Batch GSW scheme)**.** *Let* $\mathbf{B}, \mathbf{R}_m, \mathbf{R}$ *be generated in* $H.\mathsf{KeyGen}$ *and* $H.\mathsf{Enc}$*. Then, the joint distribution* $(\mathbf{B}, \mathbf{BR}_m, \mathbf{BR})$ *is computationally indistinguishable from uniform over* $\mathbb{Z}_q^{(n+r)\times m} \times \mathbb{Z}_q^{(n+r)\times N} \times \mathbb{Z}_q^{(n+r)\times N}$.

Proof. $\mathbf{B}$ is indistinguishable from uniform over $\mathbf{Z}_q^{(n+r)\times m}$ using the $\mathsf{dLWE}_{q,n,m,\chi}$ assumption r times (more specifically, using the assumption where the secret is taken accordingly to an error distribution rather than randomly, considering an appropriate choice of the error distribution, this assumption reduces to the standard one). Then, we can apply the leftover hash lemma on $(\mathbf{B}, \mathbf{BR}_m)$ and $(\mathbf{B}, \mathbf{BR})$ which concludes the proof.

It is enough if the matrices $\mathbf{P}_m$ are not given to the adversary. Otherwise using also the circular security assumption on $\mathbf{P}_m$ as in [HAO15], the joint distribution $(\mathbf{B}, \mathbf{BR}_m, P_m, \mathbf{BR})$ is computationally indistinguishable from uniform over $\mathbb{Z}_q^{(n+r)\times m} \times \mathbb{Z}_q^{(n+r)\times N} \times \mathbb{Z}_q^{(n+r)\times N} \times \mathbb{Z}_q^{(n+r)\times N}$. □

Remark 3.2.4. *Note that for the decryption algorithm, only* r *columns of the ciphertext are used, in the hybrid framework context it enables Claude to compress the ciphertext in* $\mathbb{Z}_q^{(r+n)\times N}$ *to one in* $Z_q^{(r+n)\times r}$*, that Alice can still decrypt.*

3.2.2 Ring GSW

This scheme is a ring version of GSW, introduced in the appendix of [GSW13], and implemented in [KGV14], transposing the *approximate eigenvector problem* into the ring setting. From λ the security parameter and the considered applications, we can derive the parameters n, q and $\mathcal{M}$ of the scheme described below.

- $H.\mathsf{KeyGen}(n, q, \chi, \mathcal{M})$. On inputs the lattice dimension n, which is set to a power of 2, the modulus q, the error distribution χ and the plaintext space $\mathcal{M}$:
  - Set $R = \mathbb{Z}[X]/(X^n + 1)$, $R_q = R/qR$, $\ell = \lceil \log q \rceil$, $N = 2\ell$ and $\mathcal{C} = R_q^{2\times N}$.
  - Set $R_{0,1} = \{P \in R_q, p_i \in \{0,1\}, 0 \leq i < n\}$.
  - Pick $a \leftarrow_\$ R_q$, $s' \leftarrow_\$ \chi$ and $e \leftarrow_\$ \chi$.

  - Set $\mathbf{s} = [1| - s']^\top \in R_q^{1\times 2}$ and $\mathbf{b} = \begin{pmatrix} s'a + e \\ a \end{pmatrix} \in R_q^{2\times 1}$.
  - Output $\mathsf{pk}^H := \mathbf{b}$ and $\mathsf{sk}^H := \mathbf{s}$.

- $H.\mathsf{Enc}(\mathsf{pk}^H, m)$. On input pk^H, and $m \in \mathcal{M}$:
  - Pick $\mathbf{E} \leftarrow_\$ \chi^{2\times N}$.
  - Pick $\mathbf{r} \leftarrow_\$ R_{0,1}^N$, and output $\mathbf{C} = [\mathbf{b}\mathbf{r}^\top + m\mathbf{G} + \mathbf{E}]_q \in R_q^{2\times N}$, where $\mathbf{G} = (2^0, \cdots, 2^{\ell-1})^\top \otimes \mathbf{I} \in R_q^{2\times N}$.

- $H.\mathsf{Dec}(\mathbf{C}, \mathsf{sk}^H)$. On input the secret key sk^H, and a ciphertext $\mathbf{C}$:
  - Compute $m' = \lfloor [< \mathbf{s}, \mathbf{c}_l >]_q \rceil_2$.
  - Output $m' \in R_q$.

- The $H.\mathsf{Eval}$ algorithm finally consists in iterating $H.\mathsf{Add}$ and $H.\mathsf{Mul}$:
  - $H.\mathsf{Add}(\mathbf{C}_1, \mathbf{C}_2) : \mathbf{C}_+ = \mathbf{C}_1 + \mathbf{C}_2$.
  - $H.\mathsf{Mul}(\mathbf{C}_1, \mathbf{C}_2) : \mathbf{C}_\times = \mathbf{C}_1 \times \mathbf{G}^{-1}\mathbf{C}_2$.

Remark 3.2.5. *The plaintext space $\mathcal{M}$ has a major influence on the considered application in terms of quantity of information contained in a single ciphertext and error-growth. For our application we choose $\mathcal{M}$ as the set of polynomials with all coefficients of degree greater than 0 being zero, and the constant coefficient being bounded. This choice is very restrictive, later we will explain that the error-growth can be exponential in the plaintext norm, justifying this restriction. The use of polynomial is still interesting in this case as FFT can be applied to speed up the homomorphic operations.*

Lemma 3.2.6 (Correctness of Ring-GSW scheme)**.** *For every $(\mathsf{pk}^H, \mathsf{sk}^H) \leftarrow H.\mathsf{KeyGen}(n, q, \chi, \mathcal{M})$, $m \in \mathcal{M}$ and $\mathbf{C} \leftarrow H.\mathsf{Enc}(\mathsf{pk}^H, m)$ (respectively $\mathbf{C}_f \leftarrow H.\mathsf{Eval}(f, \mathbf{C}_1, \cdots, \mathbf{C}_k, \mathsf{pk}^H)$) such that all the coefficients of $|\mathbf{s}^\top \mathbf{c}_\ell - m2^{\ell-1} \mod q|$ are inferiors to $2^{\ell-2}$ then we have $m = H.\mathsf{Dec}(\mathbf{C}, \mathsf{sk}^H)$ (respectively $f(m_1, \cdots, m_k) \mod 2 = H.\mathsf{Dec}(\mathbf{C}_f, \mathsf{sk}^H)$).*

The proof follows directly from the one of Lemma 3.2.2 for each coefficient of the polynomial.

Lemma 3.2.7 (Security of Ring-GSW scheme)**.** *Let $\mathsf{pk}^H \leftarrow H.\mathsf{KeyGen}$ and $\mathbf{C} \leftarrow H.\mathsf{Enc}(\mathsf{pk}^H, m)$. Then the joint distribution $(\mathbf{b}, \mathbf{C})$ is computationally indistinguishable from uniform over $R_q^{2\times 1} \times R_q^{2\times N}$.*

Proof. $\mathbf{b}$ is computationally indistinguishable from uniform over R_q^2 using the $\mathsf{dRLWE}_{R,q,\chi}$ assumption (more specifically it relies on the assumption where s' is taken from χ rather than randomly from the dual fractional ideal $R_q^\vee$ but this assumption reduces to the $\mathsf{dRLWE}_{R,q,\chi}$ assumption). Applying the $\mathsf{dRLWE}_{R,q,\chi}$ assumption with a secret from $R_{0,1}$ on $\mathbf{b}$, $\mathbf{r}$, and $\mathbf{E}$, $(\mathbf{b}, \mathbf{C})$ is indistinguishable from uniform over $R_q^{2\times 1} \times R_q^{2\times N}$, it concludes the proof. □

Remark 3.2.8. *Note that for the decryption algorithm, only one columns of the ciphertext is used, in the hybrid framework context it enables Claude to compress the ciphertext in $R_q^{2\times N}$ to one in $R_q^{2\times 1}$, that Alice can still decrypt.*

3.3 Error-growth

We develop in the following the error-growth involved in the homomorphic operations of the two previous schemes we described; studying this evolution enables us to find the circuits or functions more adapted to homomorphic evaluation and therefore to find the best way to obtain practical FHE. Regardless of the functionality desired to be performed homomorphically, a huge number of circuits and also protocols can be used. Choosing the adapted designs relatively to the homomorphic scheme and the functionality results in choosing in the range from feasibility to practicability. By carefully examining the error metric of a HE scheme, we determine how we should modify a circuit to make it efficient for the same result of computation. The goal of this section is to study the error-growth of basic homomorphic operations in order to identify functions, or circuits, which are very compatible with homomorphic evaluation. This compatibility means that, when homomorphically evaluated with the described schemes, these functions or circuits involve low noise and therefore low time and data cost. Consequently, these are the appropriate basic blocks to use in a hybrid homomorphic framework to make it efficient.

3.3.1 Classical Operations

We first need to evaluate the error-growth of the basic homomorphic operations, the addition and the multiplication of ciphertexts. We use the analysis of [AP14] based on subgaussian distributions to study the error-growth in these homomorphic operations. From a coefficient or a vector following a subgaussian distribution of parameter σ, we can bound its norm with overwhelming probability. Then we can study the evolution of this parameter while performing the homomorphic operations. Hence we can bound the final error to ensure correctness.

For simplicity, we use two notations arising in the error-growth depending on the arithmetic of the underlying ring of the two schemes, γ the expansion factor (see [BGV12]) and $Norm(m_j)$ such that:

- Batched GSW: $\gamma = 1$ and $Norm(m_j) = |m_j|$ (arithmetic in $\mathbb{Z}$) .
- Ring GSW: $\gamma = n$ and $Norm(m_j) = ||m_j||_2$ (arithmetic in R).

3.3.1.1 Error-growth of $H.\mathsf{Add}$

Lemma 3.3.1 ($H.\mathsf{Add}$ error-growth)**.** *Suppose $\mathbf{C}_i$ for $1 \leq i \leq k$ are ciphertexts of a GSW based Homomorphic Encryption scheme with error components $\mathbf{e}_i$ of coefficients following a subgaussian distribution of parameter σ_i. Let $\mathbf{C}_f = H.\mathsf{Add}(\mathbf{C}_i$, for $1 \leq i \leq k)$, then $\mathbf{e}_f$ the related error follows a subgaussian distribution with parameter σ' such that:*

$$\sigma' = \sqrt{\sum_{i=1}^{k} \sigma_i^2} \quad or \quad \sigma' = \sigma\sqrt{k} \;\; if \;\; \sigma_i = \sigma, \forall i \in [k].$$

Proof. We first prove the lemma in the batched GSW setting following the analysis in [AP14] for the sum of two ciphertexts. Considering the addition of two ciphertexts we can write:

$$\mathbf{SC}_+ = \mathbf{SC}_1 + \mathbf{SC}_2 = \mathbf{E}'_1 + \begin{pmatrix} m_{1,1} \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_{1,r} \cdot \mathbf{s}_r^\top \end{pmatrix} \mathbf{G} + \mathbf{E}'_2 + \begin{pmatrix} m_{2,1} \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_{2,r} \cdot \mathbf{s}_r^\top \end{pmatrix} \mathbf{G}.$$

The error of $\mathbf{C}_+$ is therefore $\mathbf{E}_+ = \mathbf{E}'_1 + \mathbf{E}'_2 \in \mathbb{Z}_q^{r \times N}$; each row $\mathbf{e}_{+,j}^\top$ for $1 \leq j \leq r$ is the sum of $\mathbf{e}_{1,j}^\top$ and $\mathbf{e}_{2,j}^\top$. Then for $1 \leq j \leq r$ the N coefficients of $\mathbf{e}_{1,j}^\top$ (respectively $\mathbf{e}_{2,j}^\top$) follow a subgaussian distribution of parameter σ_1 (respectively σ_2) and by Pythagorean additivity each coefficient of $\mathbf{e}_{+,j}^\top$ has subgaussian parameter $\sigma_+ = \sqrt{\sigma_1^2 + \sigma_2^2}$.

Then we prove the analogous property in the ring setting. To add two ciphertexts we consider:

$$\mathbf{s}^\top \mathbf{C}_+ = \mathbf{s}^\top \mathbf{C}_1 + \mathbf{s}^\top \mathbf{C}_2 = \mathbf{e}_1'^\top + m_1 \mathbf{s}^\top \mathbf{G} + \mathbf{e}_2'^\top + m_2 \mathbf{s}^\top \mathbf{G}.$$

The error of $\mathbf{C}_+$ is therefore $\mathbf{e}_+^\top = \mathbf{e}_1'^\top + \mathbf{e}_2'^\top \in R^N$ where each one of the N coefficient is the sum of polynomials where each component follows a subgaussian distribution of parameter respectively σ_1 or σ_2. By Pythagorean additivity on subgaussian parameters, each component of the polynomials of the vector $\mathbf{e}_+^\top$ has therefore subgaussian parameter $\sigma_+ = \sqrt{\sigma_1^2 + \sigma_2^2}$.

Finally, in both cases the subgaussian parameter for the addition of k ciphertexts is simply obtained by applying successively the formula of the addition of two ciphertexts. The case $\sigma' = \sigma\sqrt{k}$ is a sub-case when all ciphertexts error distributions have identical parameter σ. □

3.3.1.2 Error-growth of $H.\mathsf{Mul}$

Lemma 3.3.2 ($H.\mathsf{Mul}$ error-growth)**.** *Suppose $\mathbf{C}_i$ for $1 \leq i \leq k$ are ciphertexts of a GSW based Homomorphic Encryption scheme with error components $\mathbf{e}_i$, of coefficients following a subgaussian distribution of parameter σ_i, corresponding to the plaintexts m_i. $\mathbf{C}_f$ is the result of a multiplicative homomorphic chain such that:*

$$\mathbf{C}_f = H.\mathsf{Mul}(\mathbf{C}_1, H.\mathsf{Mul}(\mathbf{C}_2, H.\mathsf{Mul}(\cdots, H.\mathsf{Mul}(\mathbf{C}_k, \mathbf{G})))),$$

and $\mathbf{e}_f$ is the corresponding error with subgaussian parameter σ' such that:

$$\sigma' = \mathcal{O}\left(\sqrt{N\gamma}\sqrt{\sigma_1^2 + \sum_{i=2}^{k}\left(\sigma_i \Pi_{j=1}^{i-1} Norm(m_j)\right)^2}\right).$$

Proof. We first prove the statement on batched-GSW following the same direction as [AP14] (considering only the sub-case of diagonal matrices as plaintexts). Let consider the noise in a

product of two ciphertexts $\mathbf{SC}_\times$. We have the following relations:

$$\begin{aligned}\mathbf{SC}_\times = \mathbf{SC}_1\mathbf{G}^{-1}(\mathbf{C}_2) &= \left(\mathbf{E}'_1 + \left(\begin{array}{c} m_{1,1}\cdot\mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_{1,r}\cdot\mathbf{s}_r^\top \end{array}\right)\mathbf{G}\right)\mathbf{G}^{-1}(\mathbf{C}_2)\\ &= \mathbf{E}'_1\mathbf{G}^{-1}(\mathbf{C}_2) + \left(\begin{array}{c} m_{1,1}\cdot\mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_{1,r}\cdot\mathbf{s}_r^\top \end{array}\right)\mathbf{C}_2 = \mathbf{E}'_1\mathbf{G}^{-1}(\mathbf{C}_2) + \begin{pmatrix} m_{1,1} & 0 & \dots & 0\\ 0 & m_{1,2} & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & m_{1,n}\end{pmatrix}\mathbf{SC}_2\\ &= \mathbf{E}'_1\mathbf{G}^{-1}(\mathbf{C}_2) + \begin{pmatrix} m_{1,1} & 0 & \dots & 0\\ 0 & m_{1,2} & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & m_{1,n}\end{pmatrix}\mathbf{E}'_2 + \left(\begin{array}{c} m_{1,1}m_{2,1}\cdot\mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_{1,r}m_{2,r}\cdot\mathbf{s}_r^\top \end{array}\right)\mathbf{G}.\end{aligned}$$

The error of $\mathbf{C}_\times$ is therefore $\mathbf{E}_\times = \mathbf{E}'_1\mathbf{G}^{-1}(\mathbf{C}_2) + \begin{pmatrix} m_{1,1} & 0 & \dots & 0\\ 0 & m_{1,2} & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & m_{1,n}\end{pmatrix}\mathbf{E}'_2$.

As $\mathbf{G}^{-1}(\mathbf{C}_2)$ is an $N\times N$ matrix sampled from independent subgaussian distribution with parameter $\sigma_{\mathbf{G}^{-1}} = 1$, we can consider independently the r rows of $\mathbf{E}_\times$, leading to:

$$\mathbf{e}_{\times,j}^\top = \mathbf{e}_{1,j}^\top\mathbf{G}^{-1}(\mathbf{C}_2) + m_{1,j}\mathbf{e}_{2,j}^\top.$$

The components of $m_{1,j}\mathbf{e}_{2,j}^\top$ follow a subgaussian distribution of parameter $|m_{1,j}|\sigma_2$ by homogeneity, with $|m_{1,j}|\in\mathbb{N}$ the absolute norm of $m_{1,j}$. By Pythagorean additivity and then Lemma 2.3.12, the components of $\mathbf{e}_{1,j}^\top\mathbf{G}^{-1}(\mathbf{C}_2)$ follow a subgaussian distribution of parameter:

$$\sqrt{\sum_{\ell=1}^{N}(\sigma_{\mathbf{G}^{-1}}\mathbf{e}_{1,j,\ell}^\top)^2} = \sigma_{\mathbf{G}^{-1}}||\mathbf{e}_{1,j}^\top||_2 = \mathcal{O}(\sigma_1\sqrt{N}).$$

The components of $\mathbf{e}_{\times,j}^\top$ therefore follow independent subgaussian distributions of parameter:

$$\sigma' = \mathcal{O}\left(\sqrt{(\sigma_1\sqrt{N})^2 + (|m_{1,j}|\sigma_2)^2}\right).$$

Applying this formula recursively for a multiplicative chain we obtain for σ':

$$\mathcal{O}\left(\sqrt{(\sigma_1\sqrt{N})^2 + (|m_{1,j}|\sigma_2\sqrt{N})^2 + \dots + ((\Pi_{i=1}^{k-1}|m_{i,j}|)\sigma_k\sqrt{N})^2 + ((\Pi_{i=1}^{k}|m_{i,j}|)\sigma_{\mathbf{G}})^2}\right).$$

As $\mathbf{G}$ is a noiseless encryption of $\mathbf{I}$, $\sigma_{\mathbf{G}} = 0$ and we can conclude:

$$\sigma' = \mathcal{O}\left(\sqrt{N}\sqrt{\sigma_1^2 + \sum_{i=2}^{k}(\sigma_i\Pi_{j=1}^{i-1}|m_j|)^2}\right).$$

Then we prove the property in the ring setting.

We first recall that R is a cyclotomic polynomial of degree being a power of two. Hence we have the following relations on a product of $a, b \in R_q$:

$$ab = \sum_{i=0}^{n-1} \left(\sum_{j=0}^{i} a_j b_{i-j} X^i \right) + \sum_{i=0}^{n-2} \left(\sum_{j=i+1}^{n-1} a_j b_{n+i-j} X^{n+i} \right) \mod (X^n + 1).$$

Using the reduction modulus $X^n + 1$ we get:

$$ab = \sum_{i=0}^{n-1} \left(\sum_{j=0}^{i} a_j b_{i-j} X^i \right) - \sum_{i=0}^{n-2} \left(\sum_{j=i+1}^{n-1} a_j b_{n+i-j} X^i \right),$$

and for each coefficient:

$$(ab)_i = \left(\sum_{j=0}^{i} a_j b_{i-j} \right) - \left(\sum_{j=i+1}^{n-1} a_j b_{n+i-j} \right),$$

where each coefficient of a and b appears only once in the sum. This expression on the coefficients then enables us to obtain the subgaussian parameter of a product of subgaussian polynomials.

With $a, b \in R_q$, each coefficient of b following independent subgaussian distributions of parameter σ_b, by Pythagorean additivity we obtain the subgaussian parameter σ_{ab} of the coefficients of ab:

$$\sigma_{ab} = \sqrt{\sum_{j=0}^{n-1} (a_j \sigma_b)^2} = \sigma_b ||a||_2. \tag{3.1}$$

Hereafter, as in the batched setting, we obtain the subgaussian parameter in the simpler case of a product of two ciphertexts. Multiplying two ciphertexts $\mathbf{C}_1$ and $\mathbf{C}_2$ we can consider the resultant error vector:

$$\mathbf{e}_\times^\top = \mathbf{e}_1^\top \mathbf{G}^{-1}(\mathbf{C}_2) + m_1 \mathbf{e}_2^\top \in R_q^N.$$

For the second part, we consider all polynomials of $\mathbf{e}_2$ with coefficient from subgaussian independent distributions of parameter σ_2. Then, the subgaussian parameter obtained is $\sigma_2 ||m_1||_2$ using Equation (3.1).

For the first component $\mathbf{e}_1^\top \mathbf{G}^{-1}(\mathbf{C}_2)$, each polynomial of the result is the sum of N independent products from an element of $\mathbf{e}_1^\top$ and a polynomial with coefficients following subgaussian distributions of parameter 1 (by construction of $\mathbf{G}^{-1}$). The subgaussian parameter of the first part is $\sqrt{\sum_{i=1}^{N} (||\mathbf{e}_{1i}^\top||_2)^2}$ by Equation (3.1).

Finally, the subgaussian parameter $\sigma_\times$ of the coefficients of the polynomials of $\mathbf{e}_\times$ is:

$$\sigma_\times = \mathcal{O}\left(\sqrt{(\sigma_1 \sqrt{nN})^2 + (\sigma_2 ||m_1||_2)^2} \right).$$

using Lemma 2.3.12 on the $\mathbf{e}_{1i}$ and Pythagorean additivity.

We apply this formula recursively for a multiplicative chain ending by $\mathbf{G}$. The resulting subgaussian parameter is σ':

$$\mathcal{O}\left(\sqrt{\sigma_1^2 nN + ||m_{1,j}||_2^2\sigma_2^2 nN + \cdots + \left(\left(\prod_{i=1}^{k-1}||m_{i,j}||_2\right)\sigma_k\sqrt{nN}\right)^2 + \left(\left(\prod_{i=1}^{k}||m_{i,j}||_2\right)\sigma_{\mathbf{G}}\right)^2}\right).$$

As $\mathbf{G}$ is a noiseless encryption of $\mathbf{I}$, $\sigma_{\mathbf{G}} = 0$ and we can conclude:

$$\sigma' = \mathcal{O}\left(\sqrt{nN}\sqrt{\sigma_1^2 + \sum_{i=2}^{k}(\sigma_i \Pi_{j=1}^{i-1}||m_j||_2)^2}\right).$$

□

3.3.2 Optimized Operations

We describe here some optimizations when performing homomorphic evaluation, these optimizations improve in practice the efficiency on a fixed circuit. They are the first step to design functions or circuits which involves small error-growth with the third generation FHE.

3.3.2.1 Error-growth in $H.\mathsf{Comb}$

For the sake of clarity, we formalize hereafter the corresponding *comb homomorphic product* $H.\mathsf{Comb}$ and the quantity σ_{comb} which stands for the subgaussian parameter. We study the error-growth of $H.\mathsf{Comb}$ as we will use it as a tool for the error-growth analysis of direct sums of monomials (and FLIP ciphers as explained in Chapter 4).

Definition 3.3.3 (homomorphic comb $H.\mathsf{Comb}$)**.** *Let $\mathbf{C}_1, \cdots, \mathbf{C}_k$ be k ciphertexts of a GSW based Homomorphic Encryption scheme with error coefficients from independent distributions with same subgaussian parameter σ. We define $H.\mathsf{Comb}(y, \sigma, c, k) = H.\mathsf{Mul}(\mathbf{C}_1, \cdots, \mathbf{C}_k, \mathbf{G})$ where:*

- *$y = \sqrt{N\gamma}$ is a constant depending on the ring,*
- *$c = \max_{1\leq i\leq k}(Norm(m_i))$ is a constant which depends on the plaintexts,*

and $\mathbf{C}_{comb} = H.\mathsf{Comb}(y, \sigma, c, k)$ has error components following a subgaussian distribution of parameter $\mathcal{O}(\sigma_{comb})$.

Lemma 3.3.4 (σ_{comb} quantity)**.** *Let $\mathbf{C}_1, \cdots, \mathbf{C}_k$ be k ciphertexts of a GSW based Homomorphic Encryption scheme with same error parameter σ and $\mathbf{C}_{comb} = H.\mathsf{Comb}(y, \sigma, c, k)$. Then we have:*

$$\sigma_{comb}(y, \sigma, c, k) = y\sigma c_k, \qquad \text{where } c_k = \sqrt{\sum_{i=0}^{k-1} c^{2i}}.$$

Proof. Thanks to Lemma 3.3.2 we obtain:

$\sigma_{comb} = \sqrt{N\gamma}\sqrt{\sigma^2 + \sum_{i=2}^{k}(\sigma\Pi_{j=1}^{i-1}Norm(m_j))^2}$,
$\sigma_{comb} = y\sqrt{\sigma^2 + \sum_{i=2}^{k}(\sigma c^{i-1})^2}$,
$\sigma_{comb} = y\sigma\sqrt{\sum_{i=1}^{k}(c^{i-1})^2}$,
$\sigma_{comb} = y\sigma c_k$.

□

The compatibility of this comb structure with the asymmetric multiplicative error-growth property of GSW enables us to easily quantify the error in some constructions, with a better accuracy than computing the multiplicative depth. In order to minimize the quantity σ_{comb}, we can choose the plaintext space such that $c = 1$ for freshly generated ciphertexts. The resulting $\sigma_{comb}(y, \sigma, 1, k)$ quantity is therefore $y\sigma\sqrt{k}$, growing less than linearly in the number of ciphertexts. Fixing the constant c to be 1 is usual with FHE. As we mostly consider Boolean circuits, it is usual to use plaintexts in $\{-1, 0, 1\}$ to encrypt bits, leading to $c = 1$ and therefore $c_k = \sqrt{k}$.

3.3.2.2 Sign optimization

The particular error-growth in GSW Homomorphic Encryption enables to use more optimizations to reduce the error norm and perform more operations without increasing the parameter sizes. The error-growth in $H.\mathsf{Comb}$ depends on the quantity c_k derived from bounds on norms of the plaintexts, this quantity can be reduced using negative numbers. Note that even if the decryption is performed modulo 2, the plaintext and the errors evolve in the ring of the scheme. A typical example is in the LWE-based scheme to use $m \in \{-1, 0, 1\}$ rather than $\{0, 1\}$; the c_k quantity is the same and in average the sums in $\mathbb{Z}$ are smaller. Then the norm $|\sum m_i|$ is smaller which is important when multiplying. Conserving this norm as low as possible gives better bounds and c_k coefficients, leading to smaller noise when performing distinct levels of operations. An equivalent way of minimizing the error-growth is to still use $\mathcal{M} = \{0, 1\}$ but redefining the homomorphic addition as:

$$H.\mathsf{Add}(\mathbf{C}_1, \mathbf{C}_2) = \mathbf{C}_1 \pm \mathbf{C}_2,$$

where for each $H.\mathsf{Add}$ evaluation the operation $\pm$ is randomly (and independently) chosen between the $+$ operation and the $-$ operation with probability $1/2$. The chosen operation is applied coordinate-wise on the whole matrices. This homomorphic addition is still correct because:

$$\mathbf{S}(-\mathbf{C}_2) = -\mathbf{E}'_2 - \begin{pmatrix} m_{2,1} \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline m_{2,r} \cdot \mathbf{s}_r^\top \end{pmatrix} \mathbf{G} = \mathbf{E}''_2 + \begin{pmatrix} -m_{2,1} \cdot \mathbf{s}_1^\top \\ \hline \vdots \\ \hline -m_{2,r} \cdot \mathbf{s}_r^\top \end{pmatrix},$$

where the coefficients in $\mathbf{E}''_2$ rows follow a distribution of the same subgaussian parameter as the one in $\mathbf{E}'_2$ by homogeneity and $-m = m \mod 2$.

3.3.2.3 Error-growth in MUX gates

MUX gates are well adapted to third generation error-growth, as observed in the context of branching programs [BV14], or in the context of deterministic automata [CGGI16]. The asymmetric error-growth of the third generation enables to produce low-noise ciphertexts when a MUX gate or a combination of MUX gates are evaluated, as the final error depends only on the errors from the ciphertext of the control bit and only one of the two other errors.

Lemma 3.3.5 (Homomorphic MUX error-growth)**.** *Let $\mathbf{C}_a, \mathbf{C}_b, \mathbf{C}_d$ be 3 ciphertexts of a GSW based Homomorphic Encryption scheme with error parameter $\sigma_a, \sigma_b, \sigma_d$, and $\sigma_{max} =$*

$\max\{\sigma_a, \sigma_b\}$, *and* $y = \sqrt{N\gamma}$ *the constant from the ring. Defining the MUX ciphertext as* $\mathbf{C}_{MUX} = H.\mathsf{Add}(H.\mathsf{Mul}(\mathbf{C}_d, \mathbf{C}_a - \mathbf{C}_b), \mathbf{C}_b)$*, we have:*

$$\sigma_{MUX} = \mathcal{O}\left(\sqrt{y^2\sigma_d^2 + \sigma_{max}^2}\right).$$

Proof.

$$\begin{aligned}
\mathbf{C}_{MUX} &= \mathbf{C}_d\mathbf{G}^{-1}(\mathbf{C}_a - \mathbf{C}_b) + \mathbf{C}_b,\\
\mathbf{s}^\top\mathbf{C}_{MUX} &= \mathbf{s}^\top\mathbf{C}_d\mathbf{G}^{-1}(\mathbf{C}_a - \mathbf{C}_b) + \mathbf{s}^\top\mathbf{C}_b,\\
&= \mathbf{e}_d^\top\mathbf{G}^{-1}(\mathbf{C}_a - \mathbf{C}_b) + m_d\mathbf{s}^\top(\mathbf{C}_a - \mathbf{C}_b) + \mathbf{e}_b^\top + m_b\mathbf{s}^\top\mathbf{G},\\
&= \mathbf{e}_d^\top\mathbf{G}^{-1}(\mathbf{C}_a - \mathbf{C}_b) + m_d(\mathbf{e}_a^\top + \mathbf{e}_b^\top) + m_d(m_a - m_b)\mathbf{s}^\top\mathbf{G} + \mathbf{e}_b^\top + m_b\mathbf{s}^\top\mathbf{G}.
\end{aligned}$$

We obtain two cases depending on the value of d:

- If $m_d = 0$ then:

  $$\mathbf{s}^\top\mathbf{C}_{MUX} = \mathbf{e}_d^\top\mathbf{G}^{-1}(\mathbf{C}_a - \mathbf{C}_b) + \mathbf{e}_b^\top + m_b\mathbf{s}^\top\mathbf{G}.$$

  In term of errors, the first part has an error parameter of $y\sigma_d$ from Lemma 3.3.2, which gives a total error of $\sigma_{MUX} = \mathcal{O}\left(\sqrt{y^2\sigma_d^2 + \sigma_b^2}\right)$.

- If $m_d = 1$ then:

  $$\mathbf{s}^\top\mathbf{C}_{MUX} = \mathbf{e}_d^\top\mathbf{G}^{-1}(\mathbf{C}_a - \mathbf{C}_b) + \mathbf{e}_a^\top + m_a\mathbf{s}^\top\mathbf{G}.$$

  With the same reasoning we get a total error of $\sigma_{MUX} = \mathcal{O}\left(\sqrt{y^2\sigma_d^2 + \sigma_a^2}\right)$.

As d is a control bit, only these two cases can happen, giving the final formula. Note that we used the notation of ring-GSW but the result is exactly the same for batched-GSW, as we are only using lemmata 3.3.1, and 3.3.2.

□

Note that the formula in this lemma gives a final error parameter which depends on the error parameter of the 3 ciphertexts (due to the max), and not only 2, as it relates to an upper bound to tackle worst case scenarii rather than the exact value which depends on two of the 3 ciphertexts only.

3.3.3 Particular Functions

In this part we focus on particular functions whose evaluation generates a small error-growth. It is the cornerstone to find designs which strongly improve the efficiency of homomorphic encryption. In other words, finding the building blocks as compatible as possible with the error-growth metric enables to construct practical homomorphic protocols for which the time and space costs only depend on the functionalities required by the client.

3.3.3.1 Evaluating Direct Sums of Monomials

Direct sum of monomials are quite simple functions with many advantages (which are described in Chapter 5) and they are easy to evaluate homomorphically. All monomial functions can be evaluated in parallel, and all the sums are almost free, therefore the error-growth of this kind of function for a second generation FHE scheme is proportional to the logarithm of the degree of the highest monomial.

Thereafter we compute the error-growth for the third generation, we begin with a sub-case of direct sum of monomials: the sum of a linear part, a quadratic part and a sum of triangular functions as it is of particular interest later in this thesis (corresponding to FLIP functions from Definition 2.4.17). Then we expand this example to the general case of direct sums of monomials using the direct sum vector representation.

Lemma 3.3.6. *Let F be a FLIP function, i.e. a direct sum of monomials in N variables built as a sum of a linear function of n_1 variables, a quadratic function of n_2 degree-two monomials and a sum of triangular functions totalizing n_3 variables.*

Assume that $\mathbf{C}_i$ for $0 \leq i \leq N-1$ are ciphertexts of a GSW-like HE scheme with the same subgaussian parameter σ and $c = 1$. We define $\mathbf{C}_F = H.\mathsf{Eval}(F, \mathbf{C}_i)$ the output of the homomorphic evaluation of the ciphertexts $\mathbf{C}_i$'s along the circuit F. Then the error parameter σ' is:

$$\sigma' = \mathcal{O}\left(\sigma\sqrt{n_1 + y^2(n_2+n_3)}\right) \approx \mathcal{O}\left(\sigma y\sqrt{N}\right).$$

Proof. We first evaluate the noise brought by F for each of its components L_{n_1}, Q_{n_2}, ${}^{nb}\Delta^k$, defining the respective ciphertexts $\mathbf{C}_{L_{n_1}}, \mathbf{C}_{Q_{n_2}}, \mathbf{C}_{T_k}$(the last one standing for one triangle only) and the subgaussian parameter of the respective error distributions (of the components of the error vectors) $\sigma_{L_{n_1}}, \sigma_{Q_{n_2}}, \sigma_{T_k}$:

- L_{n_1}: $\mathbf{C}_{L_{n_1}} = H.\mathsf{Eval}(L_{n_1}, \mathbf{C}_0, \cdots, \mathbf{C}_{n_1-1}) = H.\mathsf{Add}(\mathbf{C}_0, \cdots, \mathbf{C}_{n_1-1})$ then $\sigma_{L_{n_1}} = \sigma\sqrt{n_1}$.

- Q_{n_2}: $\mathbf{C}_{Q_{n_2}} = H.\mathsf{Add}(H.\mathsf{Mul}(\mathbf{C}_{n_1+2j}, \mathbf{C}_{n_1+2j+1}, \mathbf{G}))$ for $0 \leq j < n_2$.
  $H.\mathsf{Mul}(\mathbf{C}_{n_1+2j}, \mathbf{C}_{n_1+2j+1}, \mathbf{G}) = H.\mathsf{Comb}(y, \sigma, 1, 2)$ has subgaussian parameter
  $\mathcal{O}(\sigma_{comb}(y, \sigma, 1, 2)) = \mathcal{O}(y\sigma\sqrt{2})$ for $0 \leq j < n_2$.
  Then $\sigma_{Q_{n_2}} = \mathcal{O}\left(y\sigma\sqrt{2}\sqrt{\frac{n_2}{2}}\right) = \mathcal{O}(y\sigma\sqrt{n_2})$.

- T_k: $\mathbf{C}_{T_k} = H.\mathsf{Add}(H.\mathsf{Mul}(\mathbf{C}_{n_1+n_2+j+(i-1)(i-2)/2}; 1 \leq j \leq i); 1 \leq i \leq k)$.
  $\mathbf{C}_{T_k} = H.\mathsf{Add}(H.\mathsf{Comb}(y, \sigma, 1, i), 1 \leq i \leq k)$.
  then $\sigma_{T_k} = \mathcal{O}\left(\sqrt{\sum_{i=1}^{k}(y\sigma\sqrt{i})^2}\right) = \mathcal{O}\left(y\sigma\sqrt{\frac{k(k+1)}{2}}\right)$.
  As ${}^{nb}\Delta^k$ is obtained by adding nb independent triangles, we get:
  $\mathbf{C}_{{}^{nb}\Delta^k} = H.\mathsf{Add}(\mathbf{C}_{T_k,i}, 1 \leq i \leq nb)$,
  and $\sigma_{{}^{nb}\Delta^k} = \mathcal{O}\left(y\sigma\sqrt{nb}\sqrt{\frac{k(k+1)}{2}}\right) = \mathcal{O}(y\sigma\sqrt{n_3})$.

By Pythagorean additivity the subgaussian parameter of $\mathbf{C}_F$ is finally:

$$\sigma' = \mathcal{O}\left(\sqrt{(\sigma\sqrt{n_1})^2 + (y\sigma\sqrt{n_2})^2 + (y\sigma\sqrt{n_3})^2}\right) = \mathcal{O}\left(\sigma\sqrt{n_1 + y^2(n_2+n_3)}\right).$$

□

Similarly we extend this result to all direct sums of monomials, using the direct sum vector definition (Definition 2.4.15).

Lemma 3.3.7. *Let F be the direct sum of monomials in N variables with associated direct sum vector $[m_1, m_2, \cdots, m_d]$.*

Assume that $\mathbf{C}_i$ for $0 \leq i \leq N-1$ are ciphertexts of a GSW-like HE scheme with same subgaussian parameter σ and $c = 1$. We define $\mathbf{C}_F = H.\mathsf{Eval}(F, \mathbf{C}_i)$ the output of the homomorphic evaluation of the ciphertexts $\mathbf{C}_i$'s along the circuit F. Then the error parameter σ' is:

$$\sigma' = \mathcal{O}\left(\sigma\sqrt{m_1 + y^2\left(\sum_{2=1}^{d} i \cdot m_i\right)}\right) \approx \mathcal{O}\left(\sigma y\sqrt{N}\right).$$

Proof. We first evaluate the error given by the monomials of a given degree i, that we denote σ_i. σ_1 corresponding to the error of the linear part, as in Lemma 3.3.6, we obtain $\sigma_1 = \sigma\sqrt{m_1}$. Then the monomials of degree i with $2 \leq i \leq d$ are evaluated as product of $i+1$ ciphertexts, giving:

$$H.\mathsf{Mul}(\mathbf{C}_j, \cdots, \mathbf{C}_{j+i-1}, \mathbf{G}) = H.\mathsf{Comb}(y, \sigma, 1, i),$$

with an error with subgaussian parameter $\mathcal{O}(y\sigma\sqrt{i})$. Adding all the ciphertexts related to the same degree we get:

$$\sigma_i = \mathcal{O}(y\sigma\sqrt{i}\sqrt{m_i}) = \mathcal{O}(y\sigma\sqrt{i \cdot m_i}).$$

Adding all these ciphertexts gives $\mathbf{C}_F$ with error parameter:

$$\mathcal{O}\left(\sigma\sqrt{m_1 + y^2(\sum_{i=2}^{d} i \cdot m_i)}\right).$$

By definition of the direct sum vector: $\sum_{i=1}^{d} i \cdot m_i \leq N$ (the equality corresponding to the case where all variables appear in the ANF of F), giving the final result:

$$\sigma' \approx \mathcal{O}\left(\sigma y\sqrt{N}\right).$$

□

3.3.3.2 Evaluating Majority Functions

The asymmetric error-growth of the third generation has been proven highly compatible with the evaluation of branching programs as in [BV14], and particularly with MUX gates (see Definition 2.5.2) as witnesses the work [CGGI16] where circuits with MUX gates and deterministic finite automata are quickly evaluated. In this part we show how the majority

function can be homomorphically evaluated without generating an important error-growth. The algebraic normal form (Definition 2.4.2) of the majority function in N variables contains at least N choose $\lceil N/2 \rceil$ monomials, *i.e.* an exponential number of monomials. This number of monomials rules out the evaluation using homomorphic combs, nevertheless using a branching program approach this function can be computed with a small number of gates, and therefore it can be a function of potential use in homomorphic frameworks. Majority functions are considered in the area of Boolean functions used in cryptography as they are optimal relatively to the algebraic immunity criterion (see Definition 2.4.10). As a majority function could then be used as building block of the symmetric encryption scheme used in the hybrid framework, it justifies to study the error-growth involved when such kind of function is homomorphically evaluated. A filtering function using a majority function could then been considered as an alternative of the construction studied in Section 4.3.2, which is based on a direct sum of monomials.

First, let us consider a branching program for the function majority on $2n+1$ bits, described in Figure 3.1. Barrington's theorem proves the existence of a width 5, polynomial length branching program for majority; here we focus on a circuit whose homomorphic evaluation with a GSW-like FHE produces a small error-growth. Therefore we consider a branching program of $2n+2$ layers, where each transition from layer i to $i+1$ is indexed by the variable x_i, each dashed vertical arrow corresponds to the value 0 of the variable and each diagonal arrow corresponds to the value 1. The final result is 0 if the path ends in the left half of the last layer, 1 otherwise. The idea behind this circuit is to force a path to finish in the right half when at least $n+1$ variables are equal to 1.

Proposition 3.3.8 (Branching program for majority (informal))**.** *Let $\mathcal{B}_{2n+1}$ be the branching program of $2n+2$ layers and $\frac{(2n+2)(2n+3)}{2}$ nodes described in Figure 3.1, then $\mathcal{B}_{2n+1}$ computes the majority function in $2n+1$ bits.*

Proof. First, the circuit is well defined, for all nodes not in the terminal layer there is a transition for the two potential values of the variable associated to the layer. Then, each variable is used once and only once: the variable i is used between layers i and $i+1$. Each transition is either leading to the same position in the following layer, or leading to the successive (right) position in the following layer. For all layers the transition keeps the position if and only if the variable is equal to 0. As the first layer begins at the leftmost position, by induction, the end node of the path (in the final layer) is in the right half if and only if at least $n+1$ variables took the value 1. Finally the result of $\mathcal{B}_{2n+1}$ on the entry $(x_1, \cdots, x_{2n+1})$ is 1 if and only if $\mathsf{w_H}(x_1, \cdots, x_{2n+1}) \geq n+1$ which is the definition of Maj_{2n+1}. □

Evaluating this branching program in clear, the final node reached gives the result: 0 if we reached the left part, 1 otherwise. Considering homomorphic evaluation we want to get a unique ciphertext, encrypting this binary result. We modify this branching program to get a circuit with gates AND, MUX, XOR easy to homomorphically evaluate, and with a unique final node. We use a standard technique using the truth table of a function, the truth table associate each $2n+1$-bit possible input to the value of F evaluated in this input. The value of $F(x)$ can be computed by testing the equality of x to an input of the truth table, the result of the test being multiplied by $F(x)$, applying it to all possible inputs of the truth table (and summing the result) gives a correct computation of $F(x)$. Applying this strategy only on the inputs such that F gives 1, and summing, still gives a correct computation. For

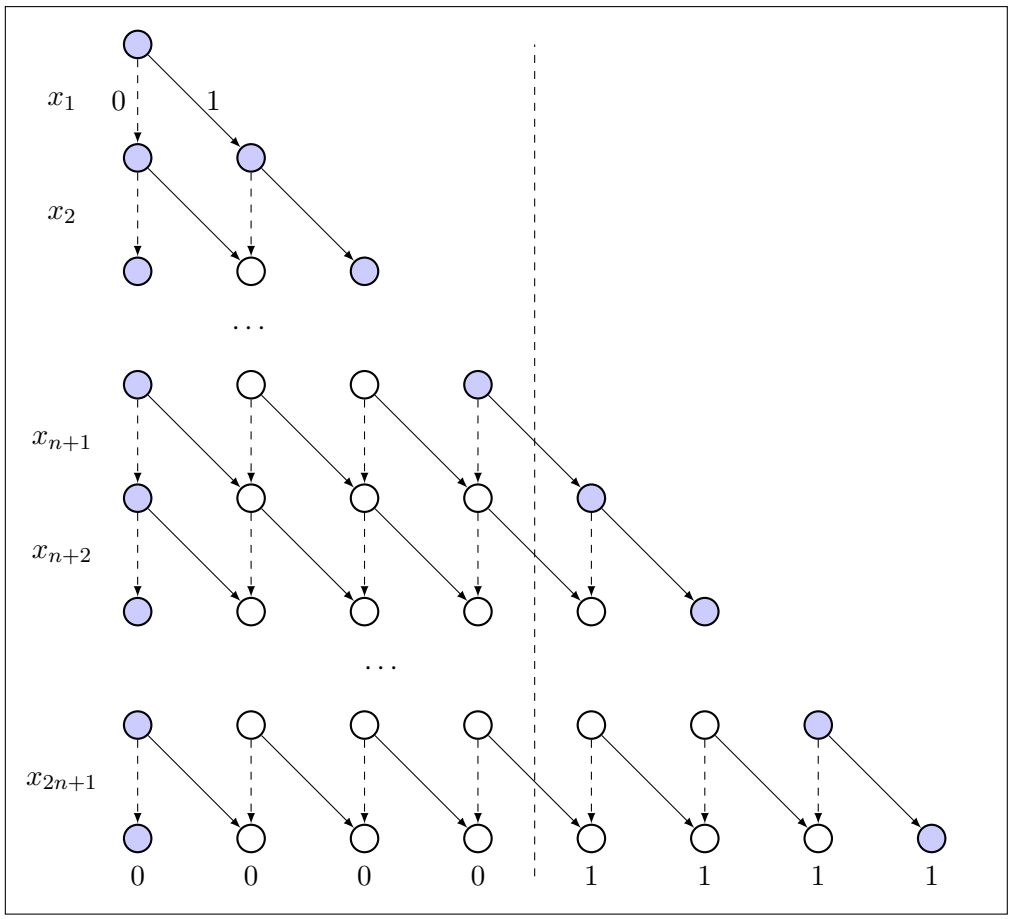

Figure 3.1: Branching program for majority.

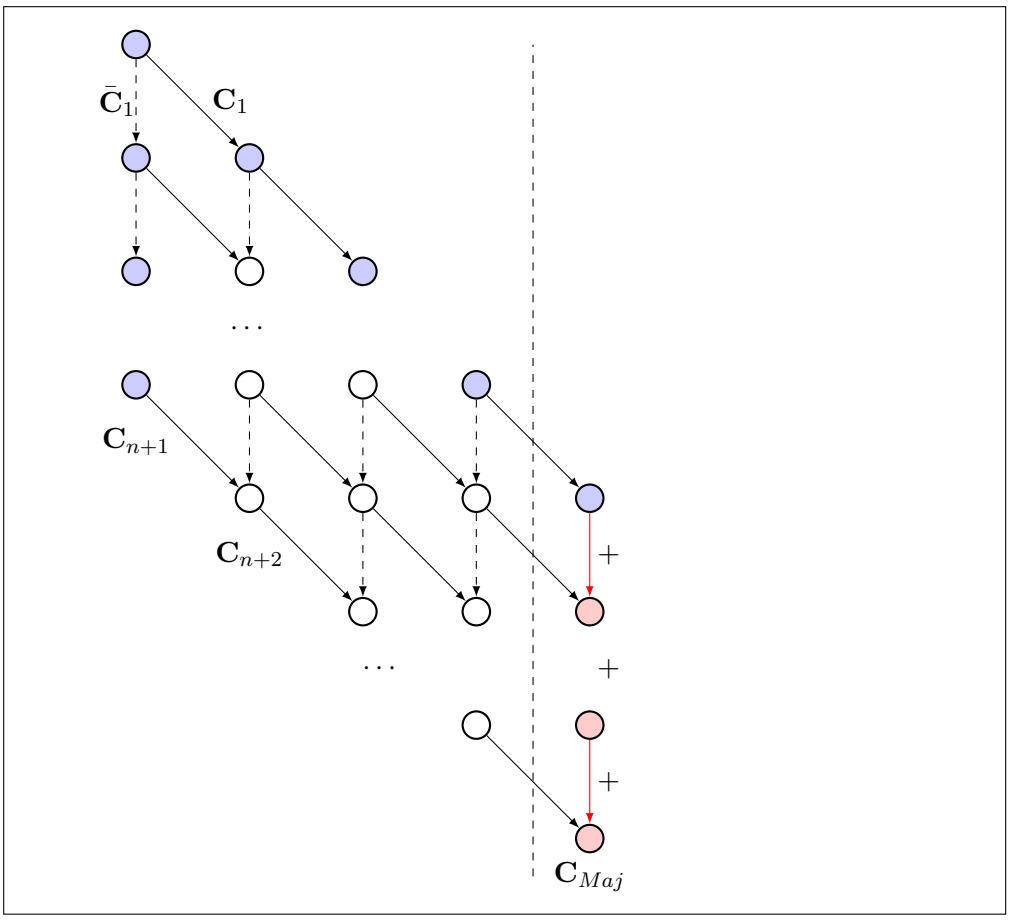

Figure 3.2: Homomorphic circuit for majority.

the branching program this approach consists in considering only the paths leading to one: the result can be obtained by summing the results of all the paths leading to 1. As only one path corresponds to the $2n+1$ bits entry really evaluated (the unique equality test giving 1), if the entry corresponds to one of these paths then the result is 1 otherwise it is 0. Therefore, homomorphically at each layer a transition indexed by x_i is replaced by a homomorphic product by $\mathbf{C}_i$ for the value 1 and by $\bar{\mathbf{C}}_i = \mathbf{G} - \mathbf{C}_i$ for the value 0.

The branching program is modified in the following way: all the transitions leading to final value 0 are cut, therefore all the bottom left part is deleted, and all the nodes for which all future transitions lead to 1 are merged: all the paths of the bottom right part are compressed, using additions. It gives the circuit to homomorphically evaluate, given in Figure 3.2. To deal with additions we modify the representation, the nodes in red are not computed as a MUX, but as a sum. The first summand is the product of the left parent node by the ciphertext of this layer, the second summand is the right parent node. Black arrows, dashed or not, symbolize the product by the ciphertext of the corresponding layer (its complementary for a dashed arrow), and a red arrow symbolizes the addition. The color of the nodes symbolizes how they are computed, a white node corresponds to a MUX, a red one to an addition, and a blue one to an AND or nothing.

From this circuit we can compute the standard deviation of the ciphertext obtained by

evaluating the majority function Maj_{2n+1}.

Lemma 3.3.9. *Let $\mathbf{C}_i$ for $0 \leq i \leq 2n+1$ be ciphertexts of a GSW-like FHE scheme with same subgaussian parameter σ and $c = 1$. We define $\mathbf{C}_{Maj}$ the output of the homomorphic evaluation of the ciphertexts $\mathbf{C}_i$'s along the circuit of Figure 3.2. Then the error parameter σ' associated to $\mathbf{C}_{Maj}$ is:*

$$\sigma' = \mathcal{O}\left(y\sigma n\sqrt{2n+1}\right).$$

Proof. We decompose the circuit in three parts to do the proof; the part of blue nodes, white nodes and red nodes on Figure 3.2. The blue nodes correspond to ciphertexts obtained by a chain of multiplications of freshly encrypted ciphertexts *i.e.* homomorphic comb, the white nodes are the output of a MUX gate and the red nodes are the output of an addition.

We first prove that the ciphertext of a blue or white node of the layer i has an error parameter of $\mathcal{O}(y\sqrt{i})$. Let us focus on the blue nodes; the ciphertext obtained at the layer i is the product of i freshly encrypted ciphertexts with error parameter σ, obtained from the ciphertexts $\bar{\mathbf{C}}_j$ (with $1 \leq j \leq i$) for the left part of the parallelogram or from the ciphertexts $\mathbf{C}_j$ (with $1 \leq j \leq i$) for the right part of the parallelogram. From Lemma 3.3.4 the associated error-growth is therefore:

$$H.\mathsf{Comb}(y, \sigma, 1, i) = y\sigma\sqrt{i}.$$

Then we prove by induction that the ciphertext of a blue or white node of the layer k has the following error parameter:

$$\mathcal{O}(y\sigma\sqrt{k}), \text{ for layer } 1 \leq k \leq 2n.$$

- $k = 1$, this layer has only two nodes, both blue, corresponding to the ciphertexts $\bar{\mathbf{C}}_1$ and $\mathbf{C}_1$. As $\bar{\mathbf{C}}_1 = \mathbf{G} - \mathbf{C}_1$, the associated error parameter is the same as the one of $\mathbf{C}_1$: $\sigma = \mathcal{O}(y\sigma\sqrt{1})$ validating the initialization (Note that the constant y appears naturally if we begin the initialization at step $k = 2$).

- $k \to k+1$, the layer $k+1$ has at least one white node. If the ciphertext corresponds to a blue node, the associate error has parameter $\mathcal{O}(y\sqrt{k+1})$ as previously proved. Otherwise, a ciphertext corresponding to a white node is obtained by a MUX gate with input two ciphertexts from the precedent layer and the control bit encrypted by $\mathbf{C}_i$. Then using Lemma 3.3.5 together with the induction hypothesis, the associated error parameter is:
$$\mathcal{O}(\sqrt{y^2\sigma^2 + \max\{y^2\sigma^2 k, y^2\sigma^2 k\}}) = \mathcal{O}(y\sigma\sqrt{k+1}),$$
proving the induction.

Note that this property also applies for the layer 0, but as for the layer 1 it serves more for notation (and for understanding the principle of the circuit) than for the final result.

The remaining part of the proof concerns the red nodes, which are the one adding two inputs; one of them (left) being the product of a cipher from the precedent level by $\mathbf{C}_i$, and the other (right) being a ciphertext of the precedent level. Note that the two summands may have not independent error terms, as they have been computed from the same ciphertexts (or minus the matrix $\mathbf{G}$), and that contrarily to the MUX gates, additions does not ensure

error independence. Then we prove by induction that the ciphertext corresponding to the red node of the layer k has the following error parameter:

$$\mathcal{O}(y\sigma \sum_{i=n+1}^{k} \sqrt{i}), \text{ for layer } n+2 \leq k \leq 2n+1.$$

- $k = n+2$, the parent nodes are a blue and a white node of the layer $n+1$, so corresponding to ciphertexts with error parameter $\mathcal{O}(y\sigma\sqrt{n+1})$. One of the two ciphertexts (left, white) is multiplied by $\mathbf{C}_{n+2}$, giving an error parameter of $\mathcal{O}(y\sigma\sqrt{n+2})$. The error parameter (recall that it is a standard deviation) of the sum is upper bounded by the sum of the two error parameters as the distributions can be correlated, giving an error parameter of:

  $$\mathcal{O}(y\sigma(\sqrt{n+1} + \sqrt{n+2})),$$

  validating the initialization.

- $k \to k+1$, the parent nodes are a white node and the red node of the layer k. The ciphertext corresponding to the white node of layer k is multiplied by $\mathbf{C}_{k+1}$, giving an error parameter of:

  $$\mathcal{O}(y\sigma\sqrt{k+1}).$$

  By the induction hypothesis, the other ciphertext has associated error parameter of:

  $$\mathcal{O}(y\sigma \sum_{i=n+1}^{k} \sqrt{i}).$$

  As the error of these two ciphertexts may be correlated, performing the sum we obtain an error parameter of:

  $$\mathcal{O}(y\sigma \sum_{i=n+1}^{k+1} \sqrt{i}),$$

  proving the induction.

The red node of the layer $2n+1$ corresponding to $\mathbf{C}_{Maj}$, we get that the final error is:

$$\mathcal{O}(y\sigma \sum_{i=n+1}^{2n+1} \sqrt{i}),$$

and as

$$\sum_{i=n+1}^{2n+1} \sqrt{i} \leq y\sigma n\sqrt{2n+1},$$

we obtain the final result of the lemma:

$$\sigma' = \mathcal{O}(y\sigma n\sqrt{2n+1})$$

□

This result shows that the majority function can be used in a homomorphic framework using a GSW-like FHE as the homomorphic error-growth involved is quite small (for a function in N variables it is proportional to $N^{1.5}$). Using a randomization technique it can be even more reduced (proportional to $N^{0.5}$). To do so, we avoid the use of additions which obliges to consider the sum of errors and we use a circuit of bigger size but using only AND and MUX gates.

The principle is to duplicate the circuit of Figure 3.1 without the part leading to 0 and to add the copy at the end of the first circuit in reverse order. This construction enables to get only one node in the final layer, and it guarantees that every path to 1 from the top circuit is by construction completed by the symmetric path (horizontal symmetry) to the final node on the bottom part. No path leading to 0 gets to the bottom circuit, and every path to 1 in the first circuit is completed by a unique path to 1 in the bottom circuit due to the symmetry. It enables to use the same technique based on the truth table of a function as for the first circuit for the homomorphic evaluation. The main difference is that the bottom circuit obliges to have new encryptions of the $2n+1$ plaintexts x_i, with independent errors. The new circuit is presented in Figure 3.3.

Note that this construction preserves the result of the homomorphic evaluation. As $x_i^2 = x_i$ in $\mathbb{F}_2$, re-use variables do not change the result. Then the final value is a sum of all potential entries of the majority function giving 1 and all corresponding paths are counted exactly once by construction (the symmetry). The final ciphertext is an encryption of the majority over the $2n+1$ encrypted bits. From this circuit of length $4n+3$ and size $3(n+1)^2$ with MUX and AND gates we can compute the standard deviation of the ciphertext obtained by evaluating the majority function Maj_{2n+1}.

Lemma 3.3.10. *Let $\mathbf{C}_i$ for $0 \leq i \leq 2n+1$ be ciphertexts of a GSW-like FHE scheme with same subgaussian parameter σ and $c = 1$. Let $\mathbf{C}'_i$ for $0 \leq i \leq 2n+1$ be ciphertexts of the same plaintext as $\mathbf{C}_i$ (but independent distribution) of a GSW-like FHE scheme with same subgaussian parameter σ and $c = 1$. We define $\mathbf{C}_{Maj}$ the output of the homomorphic evaluation of the ciphertexts $\mathbf{C}_i$ and $\mathbf{C}'_i$ along the circuit of Figure 3.3. Then the error parameter σ' associated to $\mathbf{C}_{Maj}$ is:*

$$\sigma' = \mathcal{O}\left(y\sigma\sqrt{4n+3}\right).$$

Proof. The proof is similar to the one of Lemma 3.3.9; the circuit containing only blue and white nodes, all ciphertexts of the layer $i > 0$ is computed by a product of a ciphertext from the precedent layer with a freshly encrypted ciphertext or by a MUX gate with input two ciphertexts from the precedent layer. Then the proof by induction giving that the error parameter associated to a ciphertext of the layer i is $\mathcal{O}(y\sigma\sqrt{i})$ can here be extended to all layers, from $k = 1$ to $k = 4n+3$. Finally $\mathbf{C}_{Maj}$ being the ciphertext of the layer $4n+3$, it enables to conclude:

$$\sigma' = \mathcal{O}\left(y\sigma\sqrt{4n+3}\right).$$

□

Remark 3.3.11. *The branching program approach enables to homomorphically compute the majority function with a small error-growth whereas the ANF approach leads to a different conclusion. The bigger length and size of the second circuit enables to have an error-growth similar to the one of direct sum of monomials nevertheless it requires to perform more*

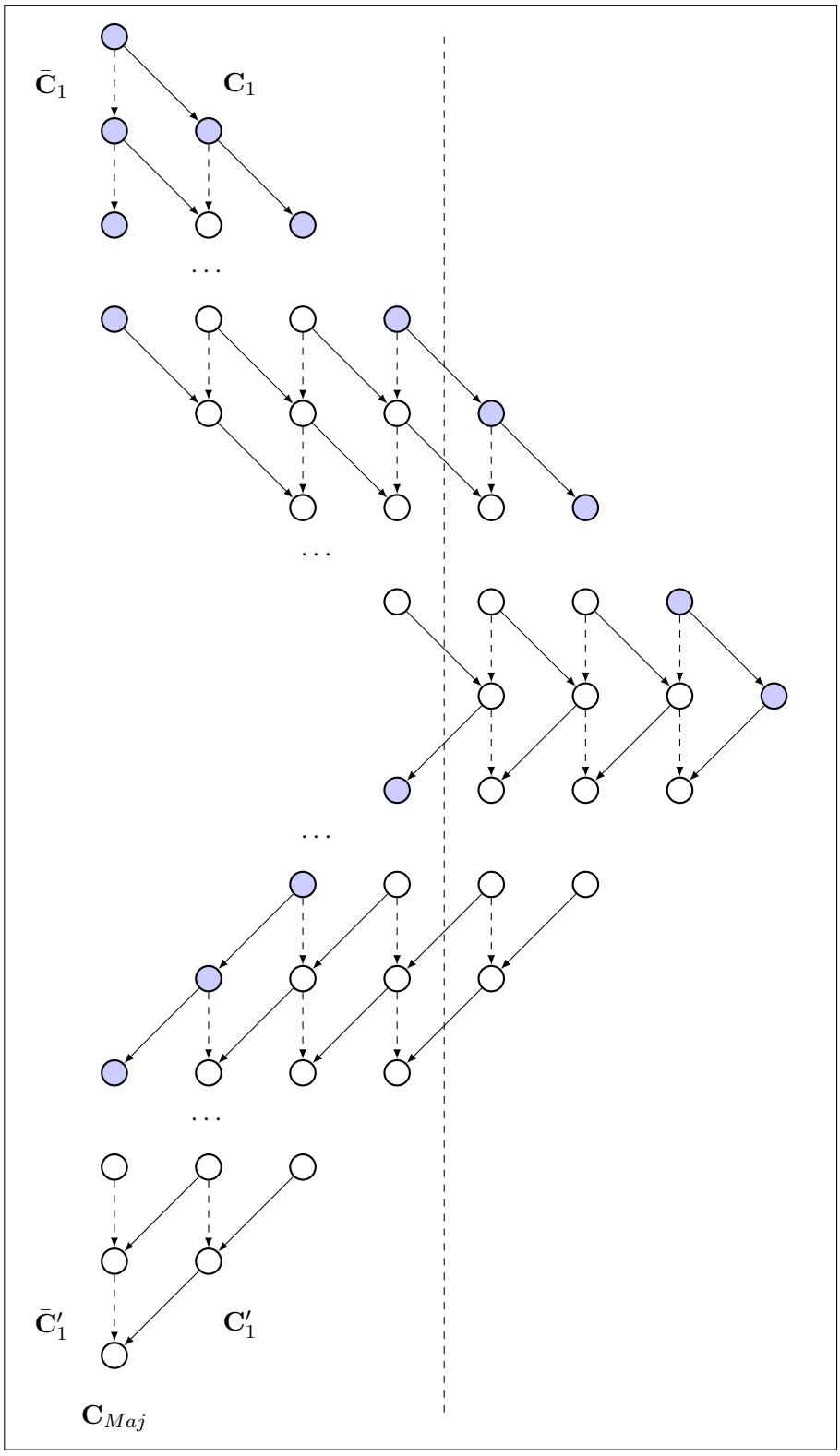

Figure 3.3: Homomorphic circuit for majority, using randomization.

homomorphic gates and to randomize ciphertexts, which has an additional cost in time and in data.

3.4 Hybrid Frameworks

In the previous part of this chapter, we saw that fully homomorphic encryption is feasible, all functions can be homomorphically evaluated, but we can debate on the efficiency aspect for the application we focus on: outsourcing computations. The error-growth study shows that the error in a ciphertext greatly depends on the function we homomorphically evaluate and how this function is expressed. The handling of the noise is the principal challenge for efficient FHE framework, it implies the usual bottlenecks of FHE that we briefly recall here. Afterward we describe a hybrid framework, as introduced in Chapter 1 here developed with the adequate notation, designed to circumvent the FHE bottlenecks to reach efficient framework for outsourcing computations.

From the presentation of the main FHE generations, three main concerns against efficient fully homomorphic frameworks emerge: the size of the ciphertexts, the complexity of the bootstrapping, and the choice of when to bootstrap. The first main problem is the size of the ciphertexts; the error used cannot be too small otherwise standard polynomial attacks on lattices as LLL or BKZ can break the security. However the error cannot be too big, otherwise only few operations are possible before bootstrapping. To handle this error the modulus is increased, the ciphertexts used for FHE applications are then way bigger than other constructions relying on the same assumptions. Therefore this problem is connected to the works on schemes based on better assumptions enabling smaller ciphertexts [BV11; BLP+13], and the works on cryptanalysis on lattices (*e.g.* [LLL82; SE94; CN11; APS15]) to find the parameters enabling compromise between security and ciphertext size. The concern of ciphertext size in homomorphic frameworks is referred as the *expansion factor* issue, which corresponds to the quantity of data needed to encrypt one bit of plaintext. For a security parameter $\lambda \approx 100$, the dimension of the lattices considered is at least around 2^9, or 2^{10}, the modulus is polynomial in this quantity, making the expansion factor of the order of one megabit at least. This factor can be reduced using *batching* techniques, encrypting more data in a single ciphertext, which has been studied by a large line of works [GHS12; HAO15].

Then, the second core problem relates to the complexity of bootstrapping. This algorithm consists in homomorphically evaluating a decryption circuit of a public key encryption scheme, therefore its complexity cannot be too low in practice. Concerning the data complexity, the bootstrapping key corresponds to an encryption of the secret key by this scheme, therefore both the expansion factor and the secret key size have an influence on the final size of the bootstrapping key. Concerning the time complexity, a lot of works have been made to make more efficient the bootstrapping technique [AP13; OPS15]. Other works examined how to reduce the number of homomorphic operations, or to adapt the decryption circuit to the error-growth metric of the underlying scheme [AP14; DM15; CGGI16].

Finally, the third main problem consists in determining the number and places of bootstrappings during a particular computation. As the execution of the bootstrapping algorithm is the process with the highest cost in an FHE framework, minimizing its number of executions enables consequent improvements in time. We can consider two extrema; on one side considering no bootstrapping, and on the other side considering a bootstrapping after each circuit gate computation. Between these two extrema, some work focused on determining the

minimal number of bootstrapping to evaluate a circuit [LP13; BLMZ16]. Another relatively close line of work consists in finding where to add bootstrapping gates in a circuit to optimize the time of the homomorphic evaluation of the whole process [PV16].

For outsourcing computation, an improvement of the previously cited bottlenecks will improve the efficiency but some issues can already be circumvented using a hybrid framework. We precise this framework overviewed in Chapter 1, using in the following the necessary concepts developed in Chapter 2 and in this chapter. The point of this hybrid framework is to use the advantage of classical symmetric encryption to minimize the costs for Alice and to make the time cost for the outsourced computation depending on the ordered computation rather than homomorphic encryption issues. It enables to transfer the factor expansion issue from Alice to Claude, and to allow Alice to only use algorithms with low cost in time and data for her part of the protocol. Reducing the size of the ciphertexts used by Alice via a hybrid encryption scheme has been first addressed by [LNV11]. More technically, the outsourcing computation can be integrated in a quite general cloud service application framework, that can be seen as a combination of 5 steps, combining a symmetric encryption scheme and an asymmetric homomorphic encryption scheme, as summarized in Figure 3.4 and described next:

1. *Initialization.* Accordingly to a security parameter λ, Alice runs the key generation algorithms $H.\mathsf{KeyGen}$ and $S.\mathsf{KeyGen}$ of the two schemes. She sends her homomorphic public key pk^H and the homomorphic ciphertext of her symmetric key $\mathbf{C}^H(\mathsf{sk}^S)$.

2. *Storage.* Alice encrypts her data m_i with the symmetric encryption scheme, and sends $\mathbf{C}^S(m_i)$ to Claude.

3. *Evaluation.* Claude homomorphically evaluates, with the $H.\mathsf{Eval}$ algorithm, the decryption $\mathbf{C}^H(m_i)$ of the symmetric scheme on Alice's data $\mathbf{C}^S(m_i)$. The $H.\mathsf{Eval}$ algorithm being defined only over homomorphic ciphertexts, he homomorphically encrypts $\mathbf{C}^S(m_i)$ (two layers of encryption) as $\mathbf{C}^H(\mathbf{C}^S(m_i)) = H.\mathsf{Enc}(\mathbf{C}^S(m_i), \mathsf{pk}^H)$, and then homomorphically evaluate the symmetric decryption:

$$\mathbf{C}^H(m_i) = H.\mathsf{Eval}(S.\mathsf{Dec}, \mathbf{C}^H(\mathbf{C}^S(m_i)), \mathbf{C}^H(\mathsf{sk}^S), \mathsf{pk}^H).$$

   If the homomorphic evaluation of the symmetric decryption algorithm is tweaked to apply directly on homomorphic and symmetric ciphertexts Claude applies it on inputs $\mathbf{C}^S(m_i)$, $\mathbf{C}^H(\mathsf{sk}^S)$, and pk^H and outputs $\mathbf{C}^H(m_i)$.

4. *Computation.* Claude homomorphically executes the treatment f on Alice's encrypted data.

5. *Result.* Claude sends a compressed encrypted result of the data treatment $\mathbf{c}^H(f(m_i))$, obtained with the $H.\mathsf{Comp}$ algorithm, and Alice decrypts it.

Some restrictions of this generic framework can lead to more efficient instantiations. Note that if we assume the existence of a trusted third party active only during the initialization step, Alice can avoid Step 1, which requires a significant computational and memory storage effort. Note also that Step 3 can be way easier to perform for some kind of symmetric schemes, those for which the homomorphic evaluation of their decryption can be tweaked to handle both symmetric and homomorphic ciphertexts. It can be done by treating separately the

symmetric encryption of the plaintext, and the homomorphic encryption of the symmetric key, during most of the evaluation process. It can also be done considering hybrid ciphertexts, containing a homomorphic part, and a part in clear. The separation technique generally applies to stream ciphers (with keystream denoted $s_{(j)}$), in this context the decryption algorithm combines homomorphic ciphertexts derived only from the encrypted symmetric key and the symmetric encryption of the plaintext. In [CCF+16], the authors refer to additive stream ciphers, they obtain a better evaluation separating it in two phases. In their context step 3 is decomposed in an off-line phase and an on-line phase. During the off-line phase Claude can homomorphically produce the whole part which is only key and IV dependent, as it only requires an IV and $\mathbf{C}^H(\mathsf{sk}_i^S)$, both sent to Claude at Step 1. During the on-line phase, Claude performs the homomorphic evaluation related to the plaintext dependent part, which can be way more efficient than the off-line phase. Note that this two-phase decomposition also applies to stream ciphers with a non-additive combination, and improve the efficiency of the whole transformation since the off-line phase is precomputed.

Then, the authors of [CCF+16] remark that for this phase performing hybrid clear and encrypted data calculations is possible for some ciphertexts, in order to avoid to homomorphically encrypt all $\mathbf{C}^S(m_j)$. Beyond this remark, for all schemes using a keystream combined with the plaintext, Claude can homomorphically produce the keystream (homomorphically encrypted) from $\mathbf{C}^H(\mathsf{sk}_i^S)$. Then the on-line phase can be trivial, he can produce hybrid ciphertexts of two parts: the first one depending only in $\mathbf{C}^S(m_j)$ and the second one depending only on $\mathbf{C}^H(s_j)$. As $\mathbf{C}^S(m_j)$ is obtained combining these two parts only, the hybrid ciphertext contains all the information needed by Alice to decrypt, and they are compatible with the homomorphic operations. These operations can be naturally extended to operations on a homomorphic part and a part in clear, allowing further computations (following some hybrid arithmetic). One advantage of these hybrid ciphertexts is to avoid to homomorphically encrypt all symmetric ciphertexts received from Alice. This data being already encrypted there is no need to encrypt it again for security reason. Therefore, the hybrid ciphertexts can be used during all the evaluation step, and even during the computation step, avoiding all re-encryptions. Even for other kinds of symmetric schemes, the inconvenience of this re-encryption can be reduced if 0-noised ciphertexts (independent of the key) are public. In this case, the symmetric ciphertexts can be replaced by the 0-noise ciphertext corresponding to their bit value, and therefore the re-encryption is not adding more noise. For the homomorphic schemes we presented, 0-noise homomorphic ciphertext corresponds for a binary symmetric ciphertext to replace 0 by the null matrix and 1 by $\mathbf{G}$.

Note also that this framework is very versatile: it can be adapted to particular client-server application and modified for efficiency of the underlying schemes. The presented framework uses a public key FHE, where Alice gives pk^H to Claude, enabling Claude to randomize ciphertext or to encrypt plaintexts that only Alice will decrypt. The homomorphic scheme can be reduced to secret-key FHE if Alice does not send pk^H. Provided 0-noise (independent of the key) ciphertexts Claude is still able to compute $\mathbf{C}^H(m_i)$ and to perform some homomorphic computations. A real use of the public key property of the FHE of this framework is to consider another client of the server, Bob, for which Alice makes the outsourced computation. In this scenario Bob is handling the homomorphic key generation, Alice uses his public key to encrypt $\mathbf{C}^H(\mathsf{sk}^S)$, and then the server is performing operations on Alice data that only Bob will be able to decrypt. Finally, a good point for the versatility of this framework is the possibility of computing in parallel, and therefore to be combined with the batch settings of homomorphic encryption schemes.

	Alice		**Claude**
1: Initialization	$(\mathsf{sk}^H, \mathsf{pk}^H) \leftarrow H.\mathsf{KeyGen}(\lambda)$ $\mathsf{sk}^S \leftarrow S.\mathsf{KeyGen}(\lambda)$ $\mathbf{C}^H(\mathsf{sk}^S) = H.\mathsf{Enc}(\mathsf{sk}^S, \mathsf{pk}^H)$	$\xrightarrow{\mathbf{C}^H(\mathsf{sk}^S),\, \mathsf{pk}^H}$	$\mathbf{C}^H(\mathsf{sk}^S), \mathsf{pk}^H$
2: Storage	$\mathbf{C}^S(m_i) = S.\mathsf{Enc}(m_i, \mathsf{sk}^S)$	$\xrightarrow{\mathbf{C}^S(m_i)}$	$\mathbf{C}^S(m_i)$
3: Evaluation			$\mathbf{C}^H(m_i)$ = $H.\mathsf{Eval}(S.\mathsf{Dec}, \mathbf{C}^H(\mathbf{C}^S(m_i)), \mathbf{C}^H(\mathsf{sk}^S), \mathsf{pk}^H)$
4: Computation	f	$\xrightarrow{f}$	$\mathbf{C}^H(f(m_i)) = H.\mathsf{Eval}(f, \mathbf{C}^H(m_i, \mathsf{pk}^H)$
5: Result	$\mathbf{c}^H(f(m_i))$ $f(m_i) = H.\mathsf{Dec}(\mathbf{c}^H(f(m_i)), \mathsf{sk}^H)$	$\xleftarrow{\mathbf{c}^H(f(m_i))}$	$\mathbf{c}^H(f(m_i)) = H.\mathsf{Comp}(\mathbf{C}^H(f(m_i)))$

Figure 3.4: Homomorphic Encryption - Symmetric Encryption framework. H and S respectively refer to homomorphic and symmetric encryption schemes, for algorithms (*e.g.* $H.\mathsf{KeyGen}$) or scheme components (*e.g.* sk^S).

Regarding security, this framework uses hybrid encryption and is related to the generic KEM-DEM construction, nevertheless the notions of security of this framework have not been particularly investigated. As the public key scheme used here is homomorphic, we cannot rely on its IND-CCA security and benefit from security guarantees of usual KEM-DEM [AGKS05]. For the applications we described, it seems reasonable to consider IND-CPA security for the homomorphic scheme, and a similar notion of indistinguishability for the symmetric scheme as explained in [BDJR97] which determines the security notion of the whole framework.

In this hybrid framework, the encryption of a scheme under another encryption scheme is sometimes called transciphering. Other transcipherings are studied in the FHE area, as between schemes of second and third generation [CGGI17], as each FHE generation has its advantages in term of error-growth, plaintext space, and ciphertext sizes. Here we focus on the main problem to make efficient the transciphering from a symmetric scheme to a fully homomorphic scheme. It consists in making the homomorphic error-growth of the symmetric decryption algorithm as low as possible, and the evaluation as fast as possible compared to the cost of the computation ordered by Alice, it will be the scope of the following chapter.

Chapter 4

Filter Permutator

In this part we introduce and study the Filter Permutator, a new kind of stream cipher designed for homomorphic frameworks. We in-light the core idea making this construction ideal to be homomorphically evaluated in the context of outsourcing computations.

First we examine the homomorphic behavior of some standard symmetric constructions, then we deduce from these behaviors a first attempt to design a scheme more adapted to homomorphic frameworks. Thereafter we present the Filter Permutator design, how the construction is optimally designed relatively to the homomorphic error-growth. We study the homomorphic results of the Filter Permutator and finally we analyze and measure its security as a symmetric encryption scheme.

Contents

The goal of this chapter is to study the Filter Permutator, a new symmetric primitive designed for efficient FHE frameworks. In the previous chapter, we tackled the homomorphic concern of the hybrid framework, here we handle the second half. In order to examine this construction, we first present other works on symmetric encryption schemes considered for this hybrid framework. They illustrate the difficulty of efficiently evaluating homomorphically a symmetric decryption algorithm. Then, some conclusions on these constructions enable to build a scheme with high potential in term of error-growth, that we consider as a first attempt. More lessons from this attempt lead to the Filter Permutator design, a new primitive we explain and instantiate.

In this chapter we examine the two core requirements of the Filter Permutator: its optimality for homomorphic evaluation and its security as a symmetric encryption scheme. The investigations on the homomorphic results prove its excellent behavior relatively to the third generation FHE, and they draw a pattern usable for all kinds of FHE. The security analysis gives the limits on the design that can be considered for the homomorphic efficiency. Moreover, the security concerns develop a concrete understanding of the construction, and propose to stand on the practical security of instances.

The results presented in this chapter essentially come from the article [MJSC16]. Otherwise it is specified, except the second section which relates to personal material conceived for the understanding of the manuscript.

4.1 Homomorphic Behavior of Standard Constructions

Let us begin this chapter by highlighting the potential difficulty to efficiently evaluate a function homomorphically, even in the case of a function used in a standard symmetric scheme. We use the denomination *standard* for symmetric schemes that are proposed and used for fast encryption scheme, as the Advanced Encryption Standard (AES [DR02]) or more recently schemes proposed to the eSTREAM or CAESAR competitions. We consider these schemes rather than instantiations of Pseudo-Random Functions or Pseudo-Random Permutations whose security is based on standard (number-theory based) assumptions because they are more relevant for a primitive which needs to be performed a lot and quickly for a very common use as symmetric encryption. Therefore standard symmetric schemes are designed to be evaluated quickly and without requiring an important storage (the expansion factor, the ratio between the ciphertext size and the corresponding plaintext size, is close to 1). Then it makes hope that homomorphically evaluating the functions of these schemes should be easy, but it is not the case. Most of the techniques used to efficiently compute a function in clear, as for a symmetric encryption scheme, are not compatible with homomorphic evaluation. Take as example a look-up table, it is used to tabulate the input/outputs of a function which is used plenty of time in one single encryption. It corresponds to a switch on the value of the input, homomorphically it can be evaluated only by verifying the equality between the encrypted input and each particular case of the switch, with the result of each equality test being afterward multiplied by an encryption of the corresponding output. In this example, all the branches of the computations have to be performed (and summed) homomorphically, which makes it very inefficient contrarily to computing this step in clear. The same inefficiency appear for additionners, and generally for all optimizations using the knowledge of an intermediate value to cut a branch of computation (as identifying 0 in a chain of AND gates or 1 in a chain of OR gates). To sum up, the efficiency of the homomorphic

evaluation of a symmetric scheme does not depend overwhelmingly in the efficiency of this scheme as a standard symmetric key cipher. Instead it rather depends on the homomorphic growth obtained when evaluating entirely its decryption circuit.

Various schemes have been considered for homomorphic frameworks, first based on AES mostly as a proof of concept for homomorphic evaluation and then on better suited schemes. Some of these schemes have been identified as compatible with homomorphic evaluation *i.e.* with decryption circuit providing a low error-growth. Finally some schemes have been designed for the hybrid framework, in order to provide candidates for outsourcing computation applications. We separate these works in three families; the first one consisting in block ciphers, the second one on stream ciphers and the third one symmetric schemes related to the LWE problem. For these three families we briefly describe the most adapted construction, we study the involved error-growth, and we conclude on the homomorphic behavior of this kind of constructions.

4.1.1 Homomorphically Evaluating a Block Cipher

Various block ciphers have been homomorphically evaluated, always with the second generation of FHE. First the AES scheme has been homomorphically evaluated [GHS12; CLT14], then ciphers more adapted as Prince [DSES14] and SIMON [LN14] and finally ciphers designed for homomorphic frameworks as LowMC [ARS+15], that we describe here.

The LowMC (standing for Low Multiplicative Complexity) family of block ciphers is a very parameterizable family designed for FHE, MPC and ZK as it is optimized to have the lower number of AND gates by bit of ciphertext or to obtain the smallest AND depth. These two metrics are more accurate in the MPC setting than in the third generation FHE as examined in the precedent chapter. We describe a round of LowMC, focusing on the multiplicative depth as it imports for second generation FHE.

A round is defined on blocks of size n as the composition of (in temporal order) an Sbox layer, a linear layer, a constant addition and a key addition where:

- The Sbox layer consists of 3-bit Sboxes with multiplicative depth 1, or the identity.
- The linear layer corresponds to a product with a matrix randomly chosen in $GL_n(\mathbb{F}_2)$ which is set public, it does not add multiplicative depth.
- The constant addition consists in adding a random binary vector of length n which is set public and which does not require to compute a product.
- The key addition consists in adding an n-bit vector to the state, where this vector is computed as the product of the key and a random binary matrix (which is set public) of size $n \times k$ and rank $\min\{n, k\}$ where k is the key size parameter. This operation does not require to homomorphically evaluate a product neither, as the product with the public matrix enables to use only additions, and accordingly for the round key addition.

To sum up, the multiplicative depth of one round is only 1, and the number of rounds necessary for a given security parameter is evaluated relatively to the known attacks. Based on current cryptanalyses [DLMW15; DEM16], it requires 12 rounds for a security of 80 bits, and 14 rounds for a security of 128 bits, giving the multiplicative depth to consider.

Examining the error-growth when LowMC is evaluated, the multiplicative depth record of 12 or 14 made it the best symmetric cipher to use for homomorphic frameworks using

second generation FHE when it was published. Nevertheless, as noticed [CCF+16] when it was evaluated with the homomorphic library HElib [HS14], the high number of additions involves an error-growth similar to 1 or 2 multiplicative levels. It imposes to allow at least 14 or 16 levels of noise only to evaluate LowMC (corresponding to Step 3 of the hybrid framework). The high number of additions comes from the linear layer, where the use of random matrices requires at each round and for each of the n ciphertexts to perform the homomorphic addition of around $n/2$ ciphertexts. For the third generation FHE, we saw in Chapter 3 that different ways for evaluating a circuit give very different error-growths. Here, the exact expression of a final ciphertext in terms of the key bits and the bits of the first round is too complicated (in theory, considering only the λ bits of the key as variables, the expression of a final ciphertext bit would be an expression with algebraic normal form of approximatively $2^{\lambda-1}$ monomials). As we cannot use this expression, we can only use the circuit to compute a ciphertext obtained after a few rounds, generally no more than one round, to determine the optimal way to homomorphically compute this ciphertext. Therefore, after one round (or a few rounds if the circuit obtained by combining a few rounds is still simple enough) there are no more fresh ciphertexts that can be used to guarantee a small error. As a consequence, the multiplicative part of the error (term y) can be a polynomial with degree equal to the number of rounds, and the repetitive use of additions increases the plaintext norm which involves a very high error-growth for further products.

On the homomorphic side various lessons can be learnt from this cipher and more generally relatively to block-ciphers. First, the iteration of rounds makes very difficult to use fresh ciphertexts during the whole homomorphic evaluation. It drastically limits the advantages of the third generation. Then, the minimal number of rounds necessary to guarantee security seems to be lower bounded, as in LowMC determined by the complexity bounds (which depend on r) of the known attacks, or proved for theoretic constructions as [LR88]. The multiplicative depth of a round being at least 1 to guarantee that it is not linear, and the existence of a minimal number of rounds, imply a lower bound on the multiplicative depths of all block-ciphers. It leads to think that even for second generation FHE the block cipher approach may not be the optimal one. On another side, the fixed number of rounds guarantees that all ciphertexts of a block contain a similar amount of noise (which is also compatible with some chaining modes as CBC mode), that can be bounded, and therefore easily studied for further applications. Finally the particular design of LowMC shows that the number of XOR cannot be totally neglected in a homomorphic framework (independently of the generation). Compensating a low number of AND gates by a huge number of XOR gates still gives a consequent error-growth.

4.1.2 Homomorphically Evaluating a Stream Cipher

In 2015; independently of [MJSC16] proposing the Filter Permutator construction, the work [CCF+16] considers the use of an additive IV-based stream-cipher for hybrid homomorphic frameworks and describes a generic construction applicable to almost all stream ciphers. The authors instantiated this generic construction with the cipher Trivium [CP08] recommended by the eSTREAM project and with Kreyvium, an extension of this lightweight cipher designed to reach a security level of 128 bits.

The generic construction they described consists of a homomorphic encryption scheme with binary plaintext space, an expansion function G mapping strings of IV length to arbitrary length, and a fixed size function F. During the encryption, the IV is extended by the function

G, and each bit of the keystream is an output of the function F taking as inputs the key and a part of the output of G. This keystream is finally XORed to the plaintexts to give the ciphertexts. When the construction is homomorphically evaluated, the function G does not require homomorphic operations as it only uses the IV, contrarily to the function F. Evaluating F requires the homomorphic encryption of the symmetric key to produce the homomorphic keystream. The final XOR with the homomorphic keystream adds a small noise; consequently in this construction the multiplicative depth comes entirely from the function F and the error-growth comes mostly from this function.

For this construction, the authors of [CCF+16] focused on additive IV-based keystream generators that can be decomposed in three functions: resynchronization, transition and filtering. The resynchronization function takes as inputs the key and the IV and outputs an n-bit initial state. The transition function ϕ computes the next internal state. Finally, the filtering function f computes a keystream segment from the current internal state. Note that the multiplicative depth of a circuit computing m keystream bits can be as high as the sum of the multiplicative depth of the resynchronization, filtering, and m times the multiplicative depth of the transition function.

The stream-ciphers chosen to instantiate this construction are Trivium, and its extension Kreyvium, due to their low-depth circuit. Trivium is a synchronous stream cipher with key and IV of 80 bits, an internal state of 288 bits, a transition function of degree 2 and a filtering function of degree 1. Kreyvium is an extension of Trivium designed to provide 128 bits of security, it consists of a key and an IV of 128 bits, an internal state of 544 bits, a transition function of degree 2 and a filtering function of degree 1.

We focus on the homomorphic error-growth involved in the evaluation of these two ciphers. The keystream length that can be produced with a circuit of multiplicative depth d is proved ([CCF+16], Propositions 1 and 2), it enables to stand on the number of ciphertexts than can be efficiently produced using second generation FHE. Concretely Trivium enables to produce 57 ciphertexts with multiplicative depth 12, or 136 bits with multiplicative depth 13, and Kreyvium enables to produce 46 or 125 ciphertexts for the same multiplicative depths. The multiplicative depth is increasing if more ciphertexts are produced, and therefore it becomes not competitive with LowMC anymore. To produce more ciphertexts the encryption process has to be reinitialized with a different IV. The increasing multiplicative depth during the encryption process, and more generally the increasing complexity of the circuit, does not lead to a low error-growth for third generation FHE. In these ciphers, there are no fresh ciphertexts that can be used when the transition function has been sufficiently applied, therefore the good properties of the homomorphic product of this generation cannot be used. As for the second generation, the slowly increasing complexity also enables to produce a few low noise ciphertexts in this context.

Various lessons can be learnt from this line of work, one of the most important being the use of a keystream and the use of an encryption process containing a key-dependent part, and an IV-dependent part. However, Trivium and Kreyvium deviate from the generic construction, as the function F is not fed by a part containing only key-dependent bits and a part containing only IV-dependent bits; in both ciphers the internal state contains both kinds of bits and is updating accordingly. As a transition function increasing the homomorphic noise is iterated, the quantity of homomorphic noise increases with the size of the keystream. Then, for all ciphers using this strategy, a limited quantity of ciphertexts usable in the homomorphic framework can be computed, with potentially lower noise than with the block cipher approach. Note that the barrier impeaching the homomorphic error-growth of the

produced ciphertexts to remain low is the error-growth involved by the transition function, the update of the internal register. This is the barrier that the Filter Permutator overcomes, as explained later in this chapter.

4.1.3 Homomorphically Evaluating an LWE-related Cipher

In the work [FHK16], the authors present three symmetric encryption schemes based on lattice problems, more precisely on learning problems. Even if these schemes are less standard on a symmetric encryption perspective, they benefit from the connexion to assumptions on lattices, assumptions that are already assumed for the security of the hybrid framework. Standard LWE cannot be used directly as a symmetric primitive, as the error could lead to a different rounding for decryption than for encryption, therefore the error has to be deterministic to avoid this correctness issue. An intuition to use these deterministic LWE encryption schemes in homomorphic framework comes from the logarithmic depth of the decryption circuit of this kind of schemes. This logarithmic depth is the crucial point for bootstrapping and therefore to reach full homomorphism, then this decryption circuit could be considerate as a valid candidate in hybrid frameworks.

The three encryption schemes studied in this work have security reduction respectively to LPN, LWR and LWE (previously defined in Section 2.3.2), and they are described in a very general way as defined in [BV11; LS12], to give a variety of instances covering the integer setting and the ring setting. For simplicity we use the same notation for these schemes: x is a plaintext, $\mathbf{S}$ the secret matrix, $\mathbf{a}$ a random vector for each ciphertext and y the ciphertext corresponding to x. We briefly describe these schemes, for a detailed description and analysis we refer to [FHK16].

- LPN version:

  An error correcting code is used with encoding function E and decoding function D, a delinearization function F involving layers of multiplication is used to avoid the Arora-Ge attack [AG11]. The encryption of x is then $S.\mathsf{Enc}(x, \mathbf{S}) = (\mathbf{a}, E(x) + F(\mathbf{aS}) + e)$, where e is the error from the Bernoulli distribution defining the LPN problem. The decryption is therefore $S.\mathsf{Dec}(y, \mathbf{S}) = D(F(\mathbf{aS}) + y)$.

- LWR version:

  The encryption of x is $S.\mathsf{Enc}(x, \mathbf{S}) = (\mathbf{a}, x + \lfloor \mathbf{aS} \rceil_p)$, where $\lfloor \cdot \rceil_p$ is a rounding function. Here q and p are powers of two, the rounding function consists in deleting the lower bits and keeping only the $\log q - \log p$ upper bits. The decryption is then $S.\mathsf{Dec}(y, \mathbf{S}) = y - \lfloor \mathbf{aS} \rceil_p$.

- LWE version:

  This scheme is derived from the precedent one, the distribution of $\mathbf{a}$ being tweaked to rely on the LWE assumption. From a matrix $\mathbf{S}$ and an error distribution χ (for e) the distribution $D_{\mathbf{S}}$ is defined such that the probability $Pr[D_{\mathbf{S}} = \mathbf{a}]$ is proportional to $Pr[|\lfloor \frac{p}{q}(\lfloor(\lfloor \mathbf{aS} \rceil_p + e)\rceil)\rceil_p - \frac{p}{q}(\lfloor \lfloor \mathbf{aS} \rceil_p + e \rceil)| < \frac{1}{4}]$. In a few words the distribution avoids to round on a different value. Then encryption and decryption are defined as in the previous scheme; only the distribution of $\mathbf{a}$ is different.

We focus in the error-growth involved in the homomorphic evaluation of these decryption circuits. The authors of [FHK16] considered the second generation FHE and more particularly

the optimizations using batching in HElib. For this generation, the multiplicative depth is crucial; for the LPN-based scheme the multiplicative depth is the sum of the multiplicative depth of the scalar product, delinearization and decoding. Without delinearization and with a 3-repetition code decoded by a majority vote the authors obtain a multiplicative depth of 5 which is better than for previously mentioned constructions. Nevertheless, a delinearization of multiplicative depth at least 3 being needed to avoid the Arora-Ge attack and the (up to date) not so well understood security of this construction makes believe that the total multiplicative depth is comparable to the other constructions. Both LWR and LWE-based schemes use an inner product and a rounding, the inner product gives a multiplicative depth of 1, but the rounding is more costly. The authors consider to keep the ciphertext without performing homomorphically the rounding part, as the plaintext is recoverable in the upper bits, but this evaluation does not give a ciphertext usable for the framework we consider. Evaluating the decryption circuit forces to extract the plaintext from the upper bits and therefore it implies a multiplicative depth of $\log q$ [GHS12] as the ExtractDigits function used in HElib or in subroutines used for bootstrapping.

Considering the third generation FHE, up to our knowledge, there is no known LPN scheme instantiation (with code and delinearization fixed) with decryption circuit providing both low error-growth and established security. We note that if an instance with a delinearization function and a decoding function similar to the ones of Section 3.3.3 is considered secure, it would give a good candidate for homomorphic frameworks. For the other two schemes, homomorphically evaluating an inner product and a rounding with the third generation FHE corresponds to the bootstrapping of [AP14] where a GSW-like encryption scheme is used to bootstrap an LWE ciphertext. It leads to avoid this kind of construction for the framework we consider for two reasons. First, bootstrapping is quite costly in time and data, then the error-growth is proportional to different parameters as $\log q$, preventing to provide a low error-growth. Note that $\log q$ cannot be too small as this quantity is related to the size of the ciphertexts, to the amount of homomorphic operations doable and to the security. The cost in time can be discussed, based on the works [DM15; CGGI16] both realizing bootstrapping in less than a second through GSW-like schemes, but they are gate-bootstrappings: after each gate of a particular type bootstrapping needs to be performed. The techniques they use cannot be used in all contexts as the ciphertexts necessitate to follow some properties. A low starting error is mandatory and it is kept low using third generation FHE only as an intermediate step, then homomorphic transciphering as key-switching and mod-switching (similar to the one described in Section 3.1.2) are performed. All this process is performed at each homomorphic gate involving noise, to give a ciphertext following the same properties and therefore usable for another gate. Homomorphic evaluation by gate-bootstrapping leads to a different approach where Alice computation has to be described gate by gate (rather than with operations), each gate being way more expensive in time to evaluate with bootstrapping than without, but without any other kind of bootstrapping needed afterward. As the time cost by gate is lower without bootstrapping and that the restriction to particular gates (as only NAND gates for [DM15]) does not seem adequate with the framework allowing all kind of computations for Alice, we do not consider these improvement usable in this context. On a technical side, the final error of these bootstrapping evaluations still contain $\log q$ and other terms in the involved error-growth.

Among the lessons that can be learnt from this line of work, these constructions are less efficient than standard symmetric constructions (data cost mostly) but potentially more adapted for the hybrid framework we consider. The state of the art shows error-growth

comparable to the one of the adapted ciphers of the previously presented two families. However, the efficiency of these schemes in our context is linked to bootstrapping, which is still a costly algorithm to perform, with logarithmic depth and always non trivial error-growth. An improvement in the efficiency of bootstrapping, as constant depth or reduced number of operations in practice, would not change only the efficiency in hybrid framework, it would generally impact fully homomorphic encryption efficiency.

4.2 First Attempt

In view of this state-of-the-art, a natural direction would be to try combining the achievements of the two standards families, block ciphers and stream ciphers. That is, to design a cipher inheriting from the constant noise property offered by block ciphers, and the lower noise levels of stream ciphers (due to the lower algebraic degree of their outputs). The goal of this section is to introduce a prototype of construction reaching these two achievements. This first attempt serves as a transition between standard constructions and constructions targeting optimal homomorphic evaluation.

First, let us have a closer look on the generic construction of [CCF+16], where a function G expends the IV and a function F generates the keystream from the key and from the output of G. Implemented with standard stream ciphers this structure is not preserved. As example the internal registers of some constructions are updated by a nonlinear function, with a content depending both of the IV and of the key for each state. Homomorphically the IV-part can be updated with nonlinear functions without increasing the noise, but the impact is different if the key part is updated with the same function. To provide a low error-growth, the update of the key part should be as low as possible. Using the IV-based keystream generator notation, it consists in finding a transition function ϕ as simple as possible for the part containing the key-dependent bits, but strong enough to ensure security when the filtering function f is applied. It leads to the following first attempt of stream cipher with low homomorphic error-growth.

Let us consider a simple filtered Fibonacci LFSR (without considering the current security of these constructions), in order to witness the impact of the transition function on the homomorphic error-growth. This construction consists of:

- A size-N register,
- an affine update function g (the feedback function of the LFSR),
- a filtering function f.

At each clock cycle the register is updated, and a keystream bit is obtained by applying F to the current register. As the key is loaded at the initial state, we can study the homomorphic error-growth by examining separately the error-growth of the ciphertexts in the current register and then the error-growth involved by f homomorphically applied to this register. We begin by examining the error of the ciphertexts corresponding to the current internal state.

Proposition 4.2.1. *Let $a_{i,t}$ be the bit at the i-th position of the register at time t, and its corresponding ciphertext $\mathbf{C}_{i,t}$ (i.e. the homomorphic ciphertext corresponding to the plaintext $a_{i,t}$ when we consider the homomorphic evaluation). Assume that $\mathbf{C}_i = \mathbf{C}_{i,0}$ for $0 \leq i \leq N-1$*

are ciphertexts of a GSW-like HE scheme with same subgaussian parameter σ and $c = 1$. Then for all $t \geq 0$ and for all i, $(0 \leq i < N)$, the error parameter σ' of $\mathbf{C}_{i,t}$ satisfies:

$$\sigma' \leq \sigma\sqrt{N}.$$

Proof. At each clock cycle the register is updated by the affine function g, therefore for any strictly positive t, each ciphertext $\mathbf{C}_{i,t}$ can be computed from the ciphertexts $\mathbf{C}_{i,t-1}$ using homomorphic additions. As the ciphertexts encrypt bits, there is no need to add twice the same ciphertext, canceling rather than adding twice gives the same result, and does not increase the error. To avoid useless additions we use a simple structure of lists, to track the ciphertext to add for each position of the internal state. Indeed, for each register position we can associate a list of integers between 0 and $N-1$, standing for the initial ciphertexts to add in order to get the ciphertext related to this position, and this list evolves with t. For positions $0 < j < N$, the list at time $t > 0$ is the one of the position $j-1$ at time $t-1$. At position 0 the list is obtained by keeping only the values happening an odd number of time in the concatenation of the lists at time $t-1$ of the taps of the feedback. For each i, (with $0 \leq i < N$), and $t \geq 0$ the ciphertext $\mathbf{C}_{i,t}$ is associated to a list of at most N numbers standing for the initial ciphertexts, hence $\mathbf{C}_{i,t}$ can be computed by evaluating the homomorphic addition of at most N independent ciphertexts with same subgaussian parameter σ and $c = 1$. Using Lemma 3.3.1, it gives the final result.

□

The filtering function F is then homomorphically applied to the ciphertexts corresponding to the current state to give an encryption of a keystream bit. As the ciphertexts corresponding to the encrypted internal state are obtained by additions of at most N initial ciphertexts, the parameter c for homomorphic combs can be as high as N or $\lceil N/2 \rceil$ using sign optimization. Then, the error-growth after applying F can be $\mathcal{O}(N^d)$ (see Lemma 3.3.2 and Lemma 3.3.4) where d is the degree of F. As F is a nonlinear filtering function, this error cannot be used in the context of homomorphic frameworks.

To conclude on this first attempt, it consists in a simple stream cipher design, relying on a linear transition. Homomorphically it reveals a particular case where the additions make the homomorphic error-growth too important. Therefore it shows that even a linear cost for the homomorphic update is too much to realize a homomorphic evaluation with a very low error. The next step is then to find an update process which makes a homomorphic evaluation with constant noise, or more adequately a zero-noise homomorphic update. That is the realization of the Filter Permutator; using a public permutation - more precisely a rearrangement - of the internal register to update it.

4.3 Filter Permutator Design and Instantiation

Here, we present the design of the Filter Permutator, a few selected instances and the potential variations of the design. This section introduces the components of the main construction, and serves as a setting point before evaluating its homomorphic behavior and its security.

4.3.1 General Design

We introduce a new stream cipher construction, next called Filter Permutator (by analogy with filter generators). Its main design principle is to filter a constant key register with a

variable (public) bit permutation. More precisely, at each cycle, the key register is (bitwise) permuted with a pseudo-randomly generated permutation, and we apply a non-linear filtering function to the output of this permuted key register. The main advantage of this construction is to always apply the non-linear filtering directly on the key bits, which allows to maintain the noise level of our outputs constant. Conceptually, this type of construction seems appealing for any FHE scheme.

The general structure of the Filter Permutator is depicted in Figure 4.1. It is composed of three parts:

- A register where the key is stored,
- a (bit) permutation generator parametrized by a Pseudo Random Number Generator (PRNG) [BM84; KL07] (which is initialized with a public IV),
- a filtering function which generates a keystream.

The Filter Permutator can be compared to a filter generator, in which the LFSR is replaced by a permuted key register. In other words, the register is no longer updated by means of the LFSR, but with pseudorandom bit permutations. More precisely, at each cycle (*i.e.* each time the filtering function outputs a bit), a public pseudo-random permutation is applied to the register and the permuted key register is filtered. Eventually, the encryption (*resp.* decryption) with a Filter Permutator simply consists in XORing the bits output by the filtering function with those of the plaintext (*resp.* ciphertext).

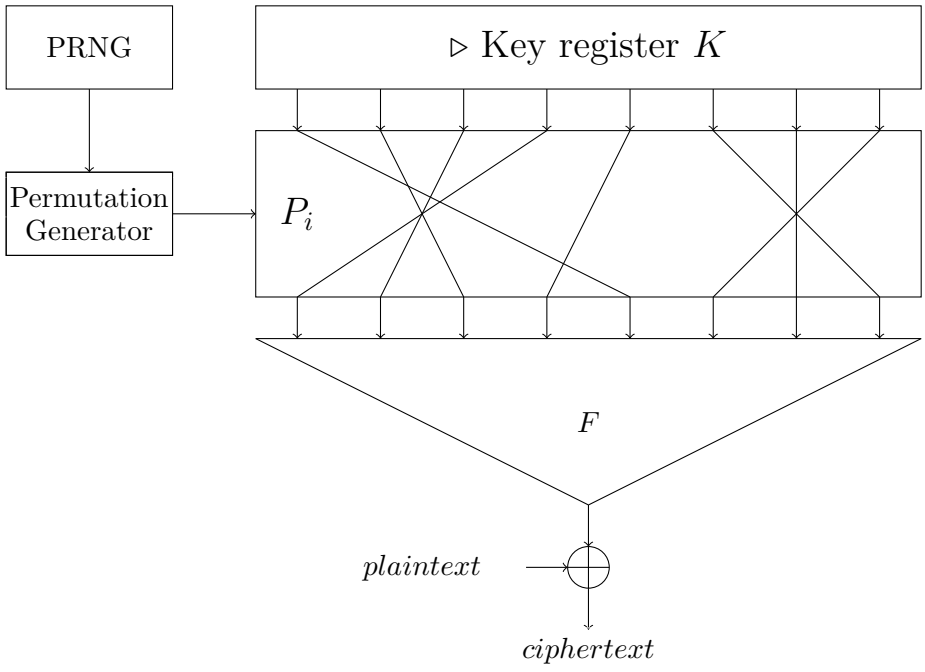

Figure 4.1: Filter Permutator construction.

4.3.2 FLIP Family of stream ciphers

We give an instantiation of the Filter Permutator with a particular structure oriented to efficient homomorphic evaluation with third generation FHE, and with proposed instances for current security levels; 80 and 128 bits of security. This third generation oriented instantiation is called FLIP, a name derived from a pronounceable abbreviation of the Filter Permutator.

4.3.2.1 FLIP Construction

Based on the general design and definitions of Boolean functions from Chapter 2, we specify the FLIP family of stream ciphers as a Filter Permutator using the following components. A key register of size N, where N is also the number of variables of F, is filled with a key randomly chosen in the set of binary words of length N and Hamming weight $N/2$. The PRNG is a forward secure PRNG [BY03] based on the AES-128 (*e.g.* as instantiated in the context of leakage-resilient cryptography [SPY13]). The permutation generator, generating rearrangements of the key register is instantiated with the Knuth shuffle [Knu97] (or Fisher-Yates shuffle). This algorithm guarantees to give the same probability to all permutations if used with a random number generator. The filtering function F is the N-variable Boolean function defined by the direct sum of three Boolean functions f_1, f_2 and f_3 of respectively n_1, n_2 and n_3 variables, such that:

- $f_1(x_1, \cdots, x_{n_1}) = L_{n_1}$,
- $f_2(x_{n_1+1}, \cdots, x_{n_1+n_2}) = Q_{n_2/2}$,
- $f_3(x_{n_1+n_2+1}, \cdots, x_{n_1+n_2+n_3})$ is the direct sum of nb triangular functions T_h, *i.e.* such that each T_h acts on different and independent variables, that we denote as ${}^{nb}\Delta^h$.

Hence, we have $F : \mathbb{F}_2^{n_1+n_2+n_3} \to \mathbb{F}_2$ the Boolean function such that:

$$F(x_1, \cdots, x_{n_1+n_2+n_3}) = L_{n_1} \oplus Q_{n_2/2} \oplus \bigoplus_{i=1}^{nb} T_h,$$

a Boolean function obtained by direct sum of monomials.

4.3.2.2 FLIP Instances

We give here the filtering functions of the 4 proposed instances of the FLIP family proposed in the Eurocrypt 2016 publication [MJSC16] in Table 4.1. Each function is defined by 4 parameters n_1, n_2, nb and h as described above: a linear function of n_1 variables, a quadratic function of n_2 variables and nb triangular functions of degree h. Notice that each of these functions being obtained by direct sum of monomials only, they can also be represented by their direct sum vector.

4.3.3 Design Tweaks

Due to the novelty of this stream cipher construction, some characteristics of the design may not be optimal yet, for a security concern or for a homomorphic concern. We discuss here two particularities of the Filter Permutator instantiated as FLIP that could be modified by design tweaks, in order to improve the security or the homomorphic evaluation efficiency.

Name	N	n_1	n_2	nb	h	λ
FLIP-530	530	42	128	8	9	80
FLIP-662	662	46	136	4	15	80
FLIP-1394	1394	82	224	8	16	128
FLIP-1704	1704	86	238	5	23	128

Table 4.1: FLIP filtering function instances, N is the total number of variables, n_1 is the number of variables over the linear part, n_2 is the number of variables over the quadratic part, nb is the number of triangular functions, h is the degree of the triangular functions and λ is the security parameter.

4.3.3.1 Invariant on F Input

Generally, security analyses of filtered registers are based on standard cryptanalysis and design tools, similarly the security of the FLIP designs can be based on properties of Boolean functions. These properties are generally computed assuming a uniform input distribution, yet, for Filter Permutators this condition is not strictly respected since the Hamming weight of the key register is constant during the encryption, as a rearrangement keeps invariant the Hamming weight. For the FLIP family the Hamming weight is fixed (set to $N/2$ in order to avoid weak keys, but even without this condition, it would be fixed to an unknown value). The impact of this non-uniform distribution is studied at the end of this chapter for the cryptanalysis concern of FLIP and at the end of Chapter 5 for the Boolean functions point-of-view.

Note that in case the Filter Permutator turns out to have weaknesses specifically due to the non-uniform distribution of F function's inputs, there are tweaks that could be used to mitigate their impact. The simplest one is to apply a public whitening to the input bits of the non-linear parts of F (using additional public PRNG outputs), which has no impact on the homomorphic error-growth. The adversary could then bias the F function's inputs based on his knowledge of the whitening bits, but to a lower extent than with fixed Hamming-weight keys. Another tweak consists in applying F to a reduced part of the key register, or equivalently to define F as a direct sum containing a null function. Then the exact Hamming weight in input of the bits influencing F output cannot be known, this incertitude can be enough to avoid the attacks applying on constant Hamming weight input. Alternatively, one could add a (more or less complex) linear layer before the non-linear part of F, which would then make the Filter Permutator conceptually more similar to filter generators, and (at least for certain layers) only imply moderate cost from the FHE point-of-view. These strategies can also be combined, leading to think that the invariant created by the zero cost homomorphic update can be rectified without damaging the good homomorphic behavior of the primitive.

4.3.3.2 Indirect Sums

Before analyzing the FHE properties of Filter Permutators, we finally suggest FLIP designs based on indirect sums as another interesting topic for evaluation, since they lead to quite different challenges. Namely, the main motivations to use direct sums in the FLIP family are the possibility to assess their cryptographic properties based on existing tools and their good

behavior relatively to homomorphic error-growth, as shown in Section 3.3.3.1. With direct sums, the homomorphic evaluation benefits of the sum of independent errors, whereas for indirect sums this independence is not guaranteed, neither the use of homomorphic combs on freshly encrypted ciphertexts. By contrast, Filter Permutator designs based on indirect sums seem harder to analyze (both for designers and cryptanalysts). This is mainly because in this case, we can make the inputs of the Boolean functions vary, but also the Boolean functions themselves. Using a filtering function F which is not a direct sum of monomials enables to use other constructions of Boolean functions, and functions with fewer variables involving a better homomorphic error-growth. More plastic constructions can even imply variations in the functions used for each keystream bit. For illustration, we can specify "multi-FLIP" designs, next denoted as b-FLIP designs, such that we compute b instances of FLIP in parallel, each with the same filtering function but with different permutations, and then XOR the b computed bits in order to produce a keystream bit. We can conjecture that such b-FLIP designs could lead to secure stream ciphers with smaller states, and suggest 10-FLIP$(10, 20, {}^1\Delta^{20})$ and 15-FLIP$(15, 30, {}^1\Delta^{30})$ as exemplary instances for 80 and 128 bits security.

4.4 Homomorphic Results

The main purpose of the Filter Permutator is to guarantee the production of low noise ciphertexts in a hybrid framework. In this part we detail how low this error can be, considering the third generation FHE or even the second. As we are considering an evaluation avoiding bootstrappings (rely on SWHE rather than FHE formally), we need to control the magnitude of the error and keep it below a critical level to ensure the correctness of a final ciphertext. This noise management is crucial for the applications, it is directly linked with the quantity of computation that the server can do for the client. We now study the error-growth stemming from the homomorphic evaluation of FLIP. In this case, all the ciphertexts used by the server in the computation step will have a same starting error. The knowledge of this starting error (defined by some parameter) and its growth for additions and multiplications (in a chosen order) is enough to determine the amount of computations that can be performed correctly by the server.

Note that independently of the generation of the used FHE, the error-growth of a ciphertext produced by a Filter Permutator construction can be determined by examining the error-growth involved by the filtering function F. Indeed, when the decryption is homomorphically evaluated, the permutations are in clear as the input of the PRNG is public, then the ciphertexts of the key bits are reordered without increasing the noise before the evaluation of F. After the evaluation of F, the unique XOR to a freshly encrypted ciphertext (or even zero noise ciphertext as explained in Section 3.4) can be neglected relatively to the noise involved by the nonlinear function F. Therefore to conclude on the homomorphic behavior of the FLIP ciphers, we focus on the error-growth involved by the filtering function, relatively to the FHE generation considered. We first study it in a general way, focusing more on the asymptotic behavior than on the particular instances, and then we present concrete results based on implemented instances.

Algorithm	Reference	Multiplicative depth	Security
SIMON-32/64	[LN14]	32	64
Trivium-12	[CCF+16]	12	80
Trivium-13	[CCF+16]	13	80
LowMC-80	[ARS+15]	11	80
FLIP − 530	[MJSC16]	$\lceil \log 9 \rceil = 4$	80
AES-128	[GHS12; CLT14]	40	128
SIMON-64/128	[LN14]	44	128
Prince	[DSES14]	24	128
Kreyvium-12	[CCF+16]	12	128
Kreyvium-13	[CCF+16]	13	128
LowMC-128	[ARS+15]	12	128
FLIP − 1394	[MJSC16]	$\lceil \log 16 \rceil = 4$	128

Table 4.2: Multiplicative depth of different symmetric ciphers.

4.4.1 General Results

4.4.1.1 FLIP and Third Generation FHE

As the filtering functions used for the FLIP instances are all direct sums of monomials, the study of homomorphic error-growth of Section 3.3.3.1 directly gives the error-growth of the ciphertexts produced when FLIP (decryption) circuit is homomorphically evaluated with a GSW-like HE scheme. From ciphertexts $\mathbf{C}^H(\mathsf{sk}_i^S)$ with associated error parameter σ, the final ciphertext used in the hybrid framework $\mathbf{C}^H(m)$ has an error parameter of $\mathcal{O}(\sigma y\sqrt{N})$. This is the same error parameter as the one involved by a multiplicative chain of N fresh ciphertexts, therefore we can limit the noise after the homomorphic evaluation of a decryption to a level of the same order of magnitude as for a single homomorphic multiplication. Consequently, we essentially make the impact of the symmetric encryption scheme as small as possible.

Note that if the design of FLIP is generalized, with instances defined for all $\lambda > 0$, different asymptotic error-growths are possible, depending on the relation between λ and N. A constant register size will imply a constant error-growth, then a degree-d polynomial increasing of the register size will imply an error-growth of half this degree.

4.4.1.2 FLIP and Second Generation FHE

The FLIP design has not been oriented for the second generation FHE, but the quite low degree of the proposed instance makes it interesting to compare it with the ciphers studied for their behavior relatively to the second generation FHE. The main measure for the compatibility with this generation being the multiplicative depth of the decryption circuit, the degree of the filtering function enables us to compare with the other ciphers as presented in Table 4.2. The multiplicative depths listed in this table are the ones given in their respective paper, rather than a comparison on the same library. The table is not complete, more specifically the results of [FHK16] have not been instantiated, otherwise the corresponding multiplicative depth will be slightly lower than the one of LowMC. The values relative to LowMC are the one presented in the original paper, the security of these version has been reduced then [DLMW15; DEM16].

Security λ	n	$\log q$
80	256	80
128	512	120

Table 4.3: (R)LWE parameters used in our applications.

Note that if the design of FLIP is generalized, with instances defined for all $\lambda > 0$, the degree of the filtering functions could evolve polynomially in λ. Then the multiplicative depth could be logarithmic in λ, enabling efficient homomorphic frameworks using the second generation for all security levels.

4.4.2 Concrete Results

4.4.2.1 Performances for Third Generation Schemes

As all previous symmetric schemes used in a hybrid homomorphic framework have been evaluated with a second generation scheme, we have no indicated library to compare the results of the FLIP instances to others. Therefore we implement the third generation evaluation of these instances on our own code, giving other measures to prove the efficiency of the design.

For the security parameters choices, we follow the analysis of Lindner and Peikert [LP11] for the hardness of LWE and RLWE, considering distinguishing and decoding attacks using BKZ [SE94; CN11]. We assume that the distribution χ in the considered LWE instances is the discrete Gaussian distribution with mean 0 and standard deviation σ. First we compute the best root Hermite factor δ of a basis (see [GN08]) computable with complexity 2^λ from the conservative lower bound of [LP11]:

$$\log(\delta) = 1.8/(110 + \lambda). \tag{4.1}$$

The distinguishing attack described in [RS10; LP11] is successful with advantage ε by finding vectors of length $\alpha\frac{q}{\sigma}$ with $\alpha = \sqrt{\ln(1/\varepsilon)/\pi}$. The length of the shortest vector that can be computed is $2^{2\sqrt{n \log q \log \delta}}$, leading to the inequation:

$$\alpha\frac{q}{\sigma} < 2^{2\sqrt{n \log q \log \delta}}. \tag{4.2}$$

Given $\sigma \geq 2\sqrt{n}$ from Regev's reduction [Reg05], we can choose parameters for n and q matching equation (4.2) for the considered security parameter λ. The parameters we select for our application are summarized in Table 4.3.

Contrary to other works published in the context of symmetric encryption schemes for efficient FHE [GHS12; CCF+16; ARS+15], our primary focus is not on the performances (see SHIELD [KGV14] for efficient implementation of Ring-GSW) but rather on the error-growth. As pointed out in [CCF+16], in most of these previous works, after the decryption process the noise inside the ciphertexts was too high to perform any other operation on them, whereas it is the main motivation for a practical use of FHE. In the following, we consequently provide experimental results about this error-growth in the ciphertexts after different operations evaluated on the Ring GSW scheme. As the link between subgaussian parameter, ciphertext error and homomorphic computation is not direct, we make some choices for representing these results focusing on giving intuition on how the error behaves.

The choice of the Ring GSW setting rather than Batched GSW is for convenience. It allows to deal with smaller matrices and faster evaluations, providing the same confirmation

Ring (n, ℓ)		FLIP	Fresh		H.Add		H.Mul		H.Eval(FLIP)	
			$\log e$	%	$\log e$	%	$\log e$	%	$\log e$	%
256	80	FLIP − 530	13, 07	17 %	13, 96	18%	19, 82	25%	24, 71	31%
512	120	FLIP − 1394	14, 68	12 %	15, 14	13%	23, 27	20%	28, 77	24%

Table 4.4: Experimental error-growth of Ring-GSW. Fresh, H.Add, H.Mul and H.Eval(FLIP) respectively stands for the noise e measure after a fresh homomorphic encryption, the homomorphic addition of two fresh ciphertexts, the homomorphic multiplication of two fresh ciphertexts and the homomorphic evaluation of FLIP on fresh ciphertexts. The first value is the log of the error e inside the corresponding ciphertexts and the percentage represents the proportion of the noise with respect to the capacity of decryption (*i.e.* $\ell - 2$).

on the heuristic error-growth. We give the parameters n and ℓ defining the polynomial ring and fix $\sigma = 2\lceil\sqrt{n}\rceil$ for the error distribution of the ciphertexts $\mathbf{C}^H(\mathsf{sk}_i^S)$.

An efficient way of measuring the error-growth within the ciphertexts is to compute the difference by applying the rounding $\lfloor\cdot\rceil_2$ in H.Dec between various ciphertexts with known plaintext. This difference (for each polynomial coefficient or vector component) corresponds to the amount of noise contained in this ciphertext. The correctness requires this quantity to be inferior to $2^{\ell-2}$. Then, considering its logarithm in base 2, it enables to have an intuitive and practical measure of the ciphertext noise: this quantity grows with the homomorphic operations until this log equals $\ell - 2$. Concretely, in our experiments we encrypt polynomials being $m = 0$ or $m = 1$, compute on the constant coefficient the quantity $e = |(\langle \mathbf{s}, \mathbf{c}_\ell \rangle - m2^{\ell-1}) \bmod q|$, and give its logarithm. We give another quantity in order to provide intuition about the homomorphic computation possibilities over the ciphertexts, by simply computing a percentage of the actual level of noise relatively to the maximal level $\ell - 2$.

Remark 4.4.1. *The quantity exhibited by our measures is roughly the subgaussian parameter of the distribution of the error contained in the ciphertexts. Considering the simpler case of a real Gaussian distribution $\mathcal{N}(0, \sigma^2)$, the difference that we compute then follows a half normal distribution with mean $\sigma\frac{\sqrt{2}}{\sqrt{\pi}}$.*

We based our prototype implementation on the NTL library combined with GMP and the discrete Gaussian sampler of BLISS [DDLL13]. We report in Table 4.4 experimental results on the error-growth for different RLWE and FLIP parameters, based on an average over one thousand of samples. The implementations were done by Anthony Journault, they correspond to proof of concepts rather than optimized code, they are experimental and not public. These results illustrate the behavior of FLIP relatively to the third generation error-growth, and not timing performances. This homomorphic evaluation is compatible with the recent library of [CGGI16], well optimized, and which could give very competitive timings.

The results confirm the quasi-additive error-growth when considering the specific metric of GSW given by the asymptotic bounds. The main conclusion of these results is that the error inside the ciphertexts after a homomorphic evaluation of FLIP is of the same order of magnitude as the one after a multiplication. The only difference between these noise increases is a term provided by the root of the symmetric key register size. Therefore, assuming a register size constant or linear in λ, with the FLIP construction the error-growth is roughly

Algorithm	Security	Nb	L	Number of Slots	Latency (sec)	Throughput (bits/min)
Trivium-12	80	45 (max)	12	600	1417.4	1143.0
	80	45 (max)	19	720	4420.3	439.8
Trivium-13	80	136 (max)	13	600	3650.3	1341.3
	80	136 (max)	20	720	11379.7	516.3
Kreyvium-12	128	42 (max)	12	504	1715.0	740.5
	128	42 (max)	19	756	4956.0	384.4
Kreyvium-13	128	124 (max)	13	682	3987.2	1272.6
	128	124 (max)	20	420	12450.8	286.8
LowMC-128	$? \leq 128$	256	13	682	3608.4	2903.1
	$? \leq 128$	256	20	480	10619.6	694.3
FLIP − 530	80	1	5	378	4.72	4805.08
	80	1	12	600	17.39	2070.16
FLIP − 1394	128	1	6	630	14.53	2601,51
	128	1	13	720	102.51	421.42

Table 4.5: Timings of the homomorphic evaluation of several instances of the Boolean function of FLIP using HElib on an Intel Core i7-3770. The other results are taken from [CCF+16]. L and Number of Slots are HElib parameters which stand respectively for the level of noise and the number of bits packed in one ciphertext. (Nb * Number of Slots) corresponds to the number of decrypted bits.

the basic multiplicative error-growth of two ciphertexts. Hence, we can conclude that Filter Permutators as FLIP release the bottleneck of evaluating symmetric decryption, and lead the further improvement of the calculus delegation framework to depend overwhelmingly on improvements of the homomorphic operations.

4.4.2.2 Performances for Second Generation Schemes

Despite our new constructions are primarily designed for 3rd-generation FHE, a look at Table 4.5 suggests that also from the multiplicative depth point of view, FLIP instances bring good results compared to their natural competitors such as LowMC [ARS+15] and Trivium/Kreyvium [CCF+16]. For completeness, we finally investigated the performances of some instances of FLIP for second generation FHE schemes using HElib, as reported in Table 4.5, where the latency is the amount of time (in seconds) needed to homomorphically decrypt (Nb * Number of Slots) bits, and the throughput is calculated as (Nb * Number of Slots * 60)/latency.

As in [CCF+16], we have considered two noise levels: a first one that does not allow any other operations on the ciphertexts, and a second one where we allow operations of multiplicative depth up to 7. Note that the (max) parenthesis in the Nb column recalls that for Trivium/Kreyvium, the homomorphic capacity decreases with the number of keystream bits generated, which therefore bounds the number of such bits before re-initializing. We observe that for 80-bit security, our instances outperform the ones based on Trivium. As for 128-bit security, the gap between our instances and Kreyvium is limited (despite the larger state of FLIP), and LowMC has better throughput in this context. Note also that our results correspond to the evaluation of the F function of FLIP (we verified that the time

needed to generate the permutations only marginally affects the overall performances of homomorphic FLIP evaluations). We finally mention that these results should certainly not be viewed as strict comparisons, since obtained on different computers and for relatively new ciphers for which we have limited understanding of the security margins (especially for LowMC [DLMW15; DEM16] and FLIP). So they should mainly be seen as an indication that besides their excellent features from the FHE capacity point-of-view, Filter Permutators inherently have good properties for efficient second generation FHE implementations as well.

4.5 Symmetric Security Analysis

The security analysis assesses the consideration of the Filter Permutator as the symmetric encryption scheme in the hybrid framework. It also enables to study the limits of this design and consequently to examine the boundaries of its homomorphic performances. In the following section we provide a security analysis of the FLIP ciphers, as the design is recent and that no reduction to a well studied assumption is known up to now, the security analysis is based on the complexity of the attacks known to apply to this construction.

More precisely, as symmetric encryption scheme used in the hybrid framework it would be nice to guarantee a strong notion of indistinguishability as in [BDJR97] or more particularly for stream-ciphers as in [BG07]. It could be obtained by relying on the Pseudorandom Function quality of a particular formulation of FLIP cipher, nevertheless as there is no known reduction between FLIP and another well-studied scheme or assumption, this would be a hypothesis with little meaning. Indeed, this hypothesis would be tailored for a scheme, and breaking this hypothesis will not solve a well-studied mathematical problem. Therefore in the following we do not examine FLIP in term of proven security, but in term of concrete cryptanalysis, our heuristic approach consists investigating the complexity of an attack on this scheme, investigating all attacks known to potentiality apply. In our context of hybrid framework, the more adapted model is known ciphertext model, we will consider it in the following section.

For this purpose, we will assume that no additional weaknesses arise from its PRNG and bit permutation generator. In this respect, we note that our forward secure PRNG does not allow malleability, so it should be hard to obtain a collision in the chosen IV model better than with the birthday probability. This should prevent collisions on the generated permutations. Besides, the Knuth shuffle [Knu97] (or Fisher-Yates shuffle) is an algorithm allowing to generate a random permutation on a finite set. This algorithm has the interesting property of giving the same probability to all permutations if used with a random number generator. As a result, we expect that any deviation between a bit permutation based on a Knuth shuffle fed with the PRNG will be hard to exploit by an adversary. Our motivation for this assumption is twofold. First, it allows us to focus on whether the filter Permutator construction is theoretically sound. Second, if such a choice was leading to an exploitable weakness, it remains possible to build a pseudorandom permutation based on standard cryptographic constructions [LR88].

As a consequence, the attacks to consider target the filtering function, we present these attacks in three parts. First we examine classical (or generic) attacks against stream ciphers, based on state-of-the-art tools. Then we study the attacks combining a guess-and-determine technique with these classical attacks. Finally we investigate the attacks using the Hamming weight invariance of the internal state together with the previous strategies.

Remark 4.5.1. *Since the permutation generation part of FLIP has only birthday security (with respect to the size of the PRNG), it implies that it is only secure up to 2^{64} PRNG outputs when implemented with the AES-128. Generating more keystream bits using block-ciphers with larger block size should be feasible. However, in view of the novelty of the FLIP instances, our claims are only made for this limited (birthday) data complexity so far, which should not be limiting for the intended FHE applications. We leave the investigation of their security against attacks using larger data complexities as a scope for further research. Besides, we note that using a PRNG based on a tweakable block cipher [LRW11] (where a part of the larger IV would be used as tweak) could be an interesting track to reduce the impact of a collision on the PRNG output in the known IV model, which we also leave as an open research direction.*

4.5.1 Classical Attacks

Since the Filter Permutator shares similarities with a filter generator, it is natural to start our investigations with the typical attacks considered against such types of stream ciphers. For this purpose, we next study the applicability of algebraic attacks and correlation attacks, together with more specialized cryptanalyses that have been considered against stream ciphers. Note that the attacks considered in the rest of this section frequently require to solve systems of equations and to implement a Gaussian reduction. Our complexity estimations will consider Strassen's algorithm for this purpose and assume $\omega = \log 7$ to be the exponent in a Gaussian elimination. Admittedly, approaches based on Groebner bases [Fau99] or taking advantage of the sparsity of the matrices [Wie86] could lead to even faster algorithms. We ignore them for simplicity in these investigations, considering security margins when instantiating. Note also that we only claim security in the single-key setting.

4.5.1.1 Algebraic Attacks

These attacks were first introduced by Courtois and Meier in [CM03] and applied to the stream cipher Toyocrypt. Their main idea is to build an over-defined system of equations with the initial state of the LFSR as unknown, and to solve this system with Gaussian elimination. More precisely, by using a nonzero function g such that both g and $h = gF$ have low algebraic degree, an adversary is able to obtain T equations with monomials of degree at most $AI(f)$. It is easily shown that g can be taken equal to the annihilator of F or of $F \oplus 1$, *i.e.* such that $gF = 0$ or $g(F \oplus 1) = 0$. After a linearization step, the adversary obtains a system of T equations in $D = \sum_{i=0}^{\mathsf{AI}(F)} \binom{N}{i}$ variables. Therefore, the time complexity of the algebraic attack is $\mathcal{O}(D^\omega)$, that is, $\mathcal{O}(N^{\omega AI(f)})$.

4.5.1.2 Fast Algebraic Attacks

They are a variation of the previous algebraic attacks introduced by Courtois at Crypto [Cou03a]. Considering the relation $gF = h$, their goal is to find and use functions g of low algebraic degree e, possibly smaller than $AI(f)$, and h of low but possibly larger degree d. Then, they lower the degree of the resulting equations by an off-line elimination of the monomials of degrees larger than e (several equations being needed to obtain each one with degree at most e). Following [ACG+06], this attack can be decomposed into four steps:

1. The search for the polynomials g and h generating a system of $D + E$ equations in

$D + E$ unknowns, where:

$$D = \sum_{i=0}^{d} \binom{N}{i}, \quad \text{and } E = \sum_{i=0}^{e} \binom{N}{i}.$$

This step has a time complexity in:

$$\mathcal{O}\left(\sum_{i=0}^{d} \binom{n}{i} + \sum_{i=0}^{e} \binom{n}{i}\right)^{\omega}.$$

2. The search for linear relations which allow the suppression of the monomials of degree more than e. This step has a time complexity in $\mathcal{O}(D \log^2(D))$.

3. The elimination of monomials of degree larger than e using the Berlekamp-Massey algorithm. This step has a time complexity in $\mathcal{O}(ED \log(D))$.

4. The resolution of the system. This step has a time complexity in $\mathcal{O}(E^{\omega})$.

Given the FAI of F, ignoring Step 1 which is trivial for our choice of F, the time complexity of this attack is:

$$O(D \log^2 D + E^2 D + E^{\omega}) \approx \mathcal{O}(N^{\mathsf{FAI}}).$$

4.5.1.3 Correlation Attacks.

In the following we call "correlation attacks" the one trying to distinguish the output sequence of a stream cipher from a random sequence, by exploiting the bias δ of the filtering function. Note that it is different from the classical correlation attacks introduced in [Sie84]. For the classical definition, the attack on filtered registers relies on the correlation of the keystream with the output of one LFSR only. We use the denomination of correlation attacks here as we consider the correlation between the keystream and a reduced part of the key. It this case, these attacks corresponds to distinguishing attacks, then more techniques are necessary for the key recovery. Therefore will assume that the data complexity of a key recovery attack is more important than the one of the corresponding distinguishing attack.

We can easily rule out such attacks by considering a (much) simplified version of Filter Permutator where the bit permutations P_i's would be made of two independent permutations P_i^1 and $P_i^{2,3}$ (respectively acting on the $n_1 + h$ bits of the linear part and the $n_2 + n_3 - h$ bits of the non-linear part of F). Suppose for simplicity that P_i^1 is kept constant t times, then the output distribution of F has a bias δ to a balanced output. In this case, a correlation attack would have a data complexity of:

$$\mathcal{O}(\delta^{-2}), \text{ with } \delta = \frac{1}{2} - \left(\frac{\mathsf{NL}(F)}{2^N}\right).$$

For simplicity, we will consider this conservative estimation in our following selection of security parameters. Yet, we note that since the permutation P_i of a Filter Permutator is acting on all the N bits of the filter F, the probability that the linear part of F is kept invariant by the permutations t times is in fact considerably small, and it is linked to the resiliency. For the functions we consider, the linear part gives all the resiliency, and even if the complexity bound of the attack does not depend on the resiliency, a non trivial resiliency

is here a good feature. Indeed, an m-resilient function ensures that no bias on the output distribution can be used if the adversary obtains a set where less than m variables are fixed. Therefore, a non-trivial resiliency implies a consequent data cost for the adversary, who needs to collect a bunch of equations with the same variables fixed in order to mount a correlation attack or a similar attack. Accordingly, we decide to guarantee a resiliency of at least $\lambda/2$.

4.5.1.4 BKW-like Attack.

The BKW algorithm was introduced in [BKW00] as a solution to solve the LPN problem using smart combinations of well-chosen vectors and their associated bias. Intuitively, our stream-cipher construction simplified as just explained (with two independent permutations P_i^1 and $P_i^{2,3}$ rather than a single one P_i) also shares similarities with this problem. Indeed, we could see the linear part as the parity of an LPN problem and the non-linear one (with a small bias) as a (large) noise. It is the principle of the more general fast correlation attacks [MS88] (explained in Section 4.5.1.6). Adapting the BKW algorithm to our setting amounts to XOR some linear parts of F in order to obtain vectors of low Hamming weight, and then to consider a distinguishing attack with the associated bias. Denoting w the target Hamming weight, x the log of the number of XORs and δ the bias, the resulting attack (which can be viewed as an extension of the previous correlation attack) has data complexity $\mathcal{O}(2^w\delta^{-2(x+1)})$. We detail the translation of our keystream correlated problem into a LPN problem in the following two paragraphs.

Decomposing F in a linear and non-linear part, we can study our filtering function by analogy with LPN and therefore consider the impact of BKW on our construction. Let $\mathbf{s} \leftarrow_\$ \{0,1\}^N$, a LPN sample is a couple $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s}\rangle + \nu)$ such that $\mathbf{a} \leftarrow_\$ \{0,1\}^N$ and $\nu \leftarrow_\$$ Bernoulli(ε). At each cycle of the filter Permutator, the permutation on the linear part of F of ℓ bits is analogous to the random choice of $\mathbf{a}$, with the restriction that its Hamming weight is fixed to ℓ. Then the non-linear part of F can be considered as the ν part of a LPN sample, such that the output bit follows a Bernoulli distribution with parameter $\varepsilon = \frac{\mathsf{NL}(F)}{2^N}$. As in our case the $\mathbf{a}$ distribution is restricted and the output bits are produced from dependent distributions, we cannot formally reduce the Filter Permutator key recovery to the search-LPN problem. Nevertheless, we can evaluate the computational cost of a strategy similar to the BKW algorithm to recover the key, based on the LF1 algorithm complexity [LF06].

Namely, writing N as $a * b$, the main point of the attack is to find a lot of groups of 2^a well-chosen vectors such that $\mathbf{a}_1 \oplus \cdots \oplus \mathbf{a}_{2^a} = \mathbf{e}_j$. With 2^a a small number, the bias introduced by XORing 2^a LPN samples is not too small, enabling to recover $\mathbf{s}_j$ from a majority vote over the different groups of 2^a vectors, since $\langle \mathbf{s}, \mathbf{a}_1 \oplus \cdots \oplus \mathbf{a}_{2^a}\rangle = \langle \mathbf{s}, \mathbf{e}_j\rangle = \mathbf{s}_j$. For our construction, the case $a = 1$ is impossible: as the Hamming weight of each $\mathbf{a}_i$ is the same, no difference can give a vector of Hamming weight 1. Therefore, at least two XORs are needed to obtain a new vector with Hamming weight 1. Let $\delta = 0.5 - \varepsilon$ be the bias of the original vectors. This implies that the bias of such new vectors is δ^3. To distinguish this bias and to recover $\mathbf{s}_j$, we therefore need $\mathcal{O}(\delta^{-6})$ operations. Such an attack can be extended by finding vectors such that the sum is $\mathbf{e}_i \oplus \mathbf{e}_j$, which leads to perform at least one XOR to obtain a targeted vector if $\ell > 2$. The computational cost of recovering one sum is then $\mathcal{O}(\delta^{-4})$. It leads to a complexity $\mathcal{O}(N\delta^{-4})$ to recover the whole key. For the case where $\ell = 2$, there is no need to perform XORs: the $\mathbf{a}_i$'s are already of Hamming weight 2 and the corresponding attacks are therefore the correlation attacks described above. But we can extend this attack considering vectors of Hamming weight $w \leq \ell$, and number of XORs x, leading to the (conservative)

complexity of $\mathcal{O}(2^w \delta^{-2(x+1)})$.

4.5.1.5 Higher-Order Correlation Attacks

These attacks were introduced by Courtois [Cou03b] and exploit the so-called XL algorithm. They look for good correlations between F and an approximation g of degree $d > 1$, in order to solve a linearized system based on the values of this approximation. The value ε is defined such that g is equal to F with probability greater than $1 - \varepsilon$. Such attacks have a (very conservative) time complexity estimate:

$$\mathcal{O}\left(\binom{N}{D}^{\omega}(1-\varepsilon)^{-m}\right), \text{ where } D \geq d \text{ and } m \geq \frac{\binom{N}{D}}{\binom{N}{D-d}}.$$

4.5.1.6 Other attacks.

Besides the previous attacks that will be taken into account quantitatively when selecting our concrete instances of FLIP designs, we also investigated the following other cryptanalyses. First, *fast correlation attacks* were introduced by Meier and Staffelbach at Eurocrypt 1988 [MS88]. A recent survey can be found in [Mei11]. The attack is divided into two phases. The first one aims at looking for relations between the output bits a_i of the LFSR to generate a system of parity-check equations. The second one uses a fast decoding algorithm (*e.g.* the belief propagation algorithm) in order to decode the words of the code $z_i = F(a_i)$ satisfying the previous relations, where the channel is a binary symmetric channel with error probability $p = \mathsf{NL}(F)/2^N$. Note that the BKW-like attack can be described accordingly, the first phase corresponds to the generation of noisy vectors of low Hamming weight (the $\mathbf{e}_j$ for example), the second phase corresponds to the majority vote. There are many attack variants of this strategy, with various decoding algorithms, giving time-data trade-offs around $\mathcal{O}(2^w \delta^{-2(x+1)})$. So we assume that the previous (conservative) complexity estimates rule out these variation as well.

Second, *weak keys* (*i.e.* keys of low or high Hamming weights) can produce a keystream sufficiently biased to determine this Hamming weight, and then to recover the key among the small amount of possible ones. The complexity of such attacks can be computed from the resiliency of F. However, since our N parameter will typically be significantly larger than the bit-security of our filter Permutator instances, we suggest to restrict the key space to keys of Hamming weight $N/2$ to rule out this concern. For this purpose, master keys can simply be generated by applying a first (secret) random permutation to any stream with Hamming weight $N/2$.

Third, *augmented function attacks* are attacks focusing on multiple outputs of the function rather than one. The goal is to find coefficients $j_1, \cdots, j_r$ such that a relation between the key and the outputs $s_{i+j_1}, \cdots, s_{i+j_r}$ can be exploited. This relation can be a correlation (as explained in [And95]) or simply algebraic [FM07]. In both cases, a prerequisite is that the relation holds on a sufficient number of i. As each bit output by FLIP depends on a different permutation, we believe that there is no exploitable relation between different outputs.

Eventually, *cube attacks* were introduced by Dinur and Shamir at Eurocrypt 2009 [DS09] as a variant of algebraic attacks taking advantage of the public parameters of a cryptographic protocol (plaintext in block ciphers, IV in stream cipher) in order to generate a system of equations of low degree. However in filter Permutator constructions, the only such public

parameter is the seed of the PRNG allowing to generate the pseudo-random bit permutations P_i. Since controlling this seed hardly allows any control of the F function's inputs, such attacks do not seem applicable. A similar observation holds for conditional differential cryptanalysis [KMN10] and for integral/zero-sum distinguishers [BC11; KW02].

4.5.2 Guess-and-Determine Attacks

Note that the relevance of guess-and-determine attacks on the FLIP cipher and more generally on the Filter Permutator construction was not considered for the first instantiations. After a presentation of work in progress at the *Journées Codages et Crypto 2015*, private communications between the authors of [MJSC16] and the ones of [DLR16a] tackle this issue. These discussions resulted in an attack published at CRYPTO [DLR16b] on a preliminary instance of the filtering function containing a single triangular function and a reduced quadratic part, and on the published version of [MJSC16] with the current instances and a section devoted to guess-and-determine impact on Boolean functions.

The principle of the guess-and-determine attack consists in guessing ℓ bits of the key in order to cancel some monomials. In our context, as the key register has not even a linear update, the guess of a key bit enables to know the value of a position in the register during all the encryption. Therefore, when the values of ℓ bits are guessed, at each clock cycle, the filtering function is then F' a $N-\ell$ bits function, with a shape depending on the part of the function where the guessed bits appear. The attack presented in [DLR16b] targets the degree of F' as the presence of a single triangular function implies a few number of monomials of degree higher than 2. With a few number of variables guesses to be 0 (the number of monomials of degree higher than 2, as if a variable of a monomial is equal to 0 the monomial is canceled), the adversary collects all the equations of degree 2, and try to solve this quadratic system by linearizing it. If the guess was correct, the key is recovered, with a time complexity corresponding to the time required for solving a quadratic system in $N-\ell$ variables multiplied by 2^ℓ potential guesses, and a data complexity depending on the probability of obtaining the function F' from the function F.

Various lessons can be learn from this attack, first a guess-and-determine strategy applies on the particular register update of the Filter Permutator, second the practicability of this kind of strategies depends on the probability of getting a particular function F' by doing ℓ guesses and on the Boolean criteria of this function F'. The attack described above targets the degree of F', which is a sub-case of the algebraic attack; it can be generalized in a straightforward manner to the other attacks. More generally, the guess-and-determine strategy allows an adversary to focus on a filtering function restricted to a subset of variables. This weaker function can then be cryptanalyzed, *e.g.* analyzed with the four aforementioned attacks, *i.e.* the algebraic attack, the fast algebraic attack, the correlation/BKW-like attacks and the higher-order correlation attack. In the following study we neglect the data complexity of obtaining the function F', and we prove conservative bounds based on the cryptographic criteria of the Boolean function F'. Then, the complexity of a guess-and-determine attack against a function F of N variables is $\min_\ell\{2^\ell \min C(F')\}$ where the second minimum is taken over all the functions F' that can be obtain by a guess of ℓ of the N variables of F (that is, over 2^ℓ times N choose ℓ potential guesses). $C(F)$ is the complexity of the best of the four attacks considered on the filtering function F. The case $\ell = 0$ corresponds to attack the scheme without guess-and-determine.

In the following, we bound the minimal complexity over these four attacks considering the

weakest functions obtained by guessing. To do so, we introduce some notation and criteria allowing us to specify the cryptographic properties of Boolean functions obtained by guessing ℓ variables, we call such criteria ''recurrent criteria". For a Boolean criterion (take as example the algebraic immunity $\mathsf{AI}(F)$), we define its recurrent version of order ℓ ($\mathsf{AI}[\ell]F$ for our example) as the minimum value it takes over all the functions F' than can be obtained by a guess of ℓ of the N variables of F. A formal definition is given in Chapter 5, where the behavior of particular functions relatively to these criteria is studied. The following lemmata enable to give a lower bound on the minimal complexity of a guess-and-determine attack combined with a standard attack, the complexity depends on the number of guesses and the recurrent criteria of the filtering function.

Lemma 4.5.2 (Guess-And-Determine & Algebraic Attack)**.** *Let F be a Boolean function in N variables and $C_{GDAA}(F)$ (respectively $C_{AA}(F)$) be the minimum complexity of the Guess-and-Determine with Algebraic Attack, denoted GDAA, (respectively Algebraic Attack) on F, then :*

$$C_{GDAA}(F) \geq \min_{0\leq\ell\leq\lambda}\left[2^{\ell}\binom{N-\ell}{\mathsf{AI}[\ell](F)}^{\omega}\right].$$

Proof. As guessing ℓ binary variables has a worst case cost of 2^{ℓ}, by definition:

$$C_{GDAA}(F) = \min_{\ell}(2^{\ell}\min C_{AA}(F')),$$

with the minimum taken over all guesses of ℓ variables, with $0 \leq \ell \leq \lambda$. We can then use the complexity of the algebraic attack, and the definition of the recurrent algebraic immunity:

$$C_{GDAA}(F) = \min_{\ell}\left[2^{\ell}\min_{F'}\left(\sum_{i=1}^{\mathsf{AI}(F')}\binom{N-\ell}{i}\right)^{\omega}\right] \geq \min_{\ell}\left[2^{\ell}\binom{N-\ell}{\mathsf{AI}[\ell](F)}^{\omega}\right].$$

Note that the inequality is the final result of the lemma; a precedent result ([MJSC16]) gives a more conservative bound where the linearized system is considered with fewer variables. Even if F' has a lower number of variables than $N-\ell$, the different permutations give a system with all the $N-\ell$ guessed variables, or restricting the number of variables consequently increases the data complexity. The conservative bound is obtained by considering $N[\ell]$: the minimal number of variables over all the functions F' obtained by guessing ℓ of the N variables, this number of variables being obtained after removing the variables not influencing the function. This bound is therefore:

$$\min_{\ell}\left[2^{\ell}\min_{F'}\binom{N(F')}{\mathsf{AI}(F')}\right].$$

Focusing on the binomial, as for all Boolean functions the algebraic immunity is inferior to the ceil of half the number of its variables, all these binomial coefficients are on the left part of Pascal's triangle, guarantying for fixed ℓ and for all functions F':

$$\binom{N(F')}{\mathsf{AI}(F')} \geq \binom{N(F')}{\mathsf{AI}[\ell](F)} \geq \binom{N[\ell](F)}{\mathsf{AI}[\ell](F)}.$$

As it applies for all F', it applies for the minimal value, giving a lower bound of:

$$\min_{\ell}\left[2^{\ell}\binom{N[\ell]}{\mathsf{AI}[\ell](F)}\right].$$

□

Lemma 4.5.3 (Guess-and-determine & Fast Algebraic Attacks)**.** *Let F be a boolean function in N variables and $C_{GDFAA}(F)$ be the minimum complexity of the Guess-and-determine with Fast Algebraic Attacks on F, then :*

$$C_{GDFAA}(F) \geq \min_{0\leq\ell\leq\lambda}\left[2^{\ell}\left(\binom{N-\ell}{\mathsf{AI}[\ell]}\log^2\binom{N-\ell}{\mathsf{AI}[\ell]} + (N-\ell)^2\binom{N-\ell}{\mathsf{AI}[\ell]} + (N-\ell)^{\omega}\right)\right].$$

Proof. By definition:

$$C_{GDFAA}(F) = \min_{\ell}(2^{\ell}\min_{F'} C_{FAA}(F')).$$

Then for all Boolean functions f, $C_{FAA}(f) = \min[(D\log^2 D + E^2D + E^{\omega})]$ where:

- The minimum is taken over all Boolean functions g and h such that $fg = h$.
- $d = \deg(h)$ and $e = \deg(g)$.
- $D = \sum_{i=1}^{d}\binom{N}{i}$ and $E = \sum_{i=1}^{e}\binom{N}{i}$.
- ω is the exponent appearing in the complexity required for solving a linear system.

We need to bound d and e for all guesses, therefore we use a property of the algebraic immunity (applying when g and h are non zero and distinct functions):

$$fg = h \Rightarrow fg = fh \Rightarrow f(g+h) = 0,$$

and by definition of $\mathsf{AI}(f)$, $deg(g+h) \geq \mathsf{AI}(f)$, hence $\max(d,e) \geq \mathsf{AI}(f)$.

As $C_{FAA}(f)$ is defined as a minimal value over all choices of g and h (not null and distinct) such that $fg = h$, we can restrict the choices to $1 \leq e \leq d$ with $d \geq \mathsf{AI}(f)$. Therefore we get:

$C_{FAA}(f) = \min(D\log^2 D + E^2D + E^{\omega}) \geq \min\left(\binom{N}{d}\log^2\binom{N}{d} + \binom{N}{e}^2\binom{N}{d} + \binom{N}{e}^{\omega}\right)$,

$C_{FAA}(f) \geq \binom{N}{\mathsf{AI}(f)}\log^2\binom{N}{\mathsf{AI}(f)} + \binom{N}{1}^2\binom{N}{\mathsf{AI}(f)} + \binom{N}{1}^{\omega}$.

Therefore, for the complexity of the fast algebraic attack with guess-and-determine, the number of variables in the system being reduced by ℓ and we get:

$$C_{GDFAA}(F) \geq \min_{0\leq\ell\leq\lambda}\left[2^{\ell}\left(\binom{N-\ell}{\mathsf{AI}[\ell]}\log^2\binom{N-\ell}{\mathsf{AI}[\ell]} + (N-\ell)^2\binom{N-\ell}{\mathsf{AI}[\ell]} + (N-\ell)^{\omega}\right)\right].$$

Note that, as for the previous lemma, a more conservative bound can be derived, replacing $N-\ell$ by $N[\ell]$, using the same techniques as the proof of the previous lemma.

□

For the correlation attacks, BKW-like attacks or similar strategies, we bound the whole attack complexity by the quantity of data necessary to distinguish a bias, considering this quantity necessary independently of the attack variant considered.

Lemma 4.5.4 (Guess-and-determine & CA/BKW-like Attack)**.** *Let F be a Boolean function in N variables and $C_{GDCA/BKW}(F)$ be the minimum complexity of the Guess-and-determine with Correlation/BKW Attack on F, then :*

$$C_{GDCA/BKW}(F) \geq \min_{0\leq\ell\leq\lambda}\{2^{\ell}\delta[\ell](F)\},$$

where $\delta[\ell]$ is the recurrent bias to $1/2$, related to $\mathsf{NL}[\ell]$.

Proof. Without considering the resiliency but only the nonlinearity criterion, for all functions f we have $C_{CA}(f) \geq \delta^{-2}$, and $C_{BKW}(f) \geq \delta^{-2}$ where C_{CA} and C_{BKW} stand for the (data) complexity of the correlation attack and for the BKW attack; δ is the bias to $1/2$, defined for generic attacks. Then applying the definitions of $C_{GDCA/BKW}(F)$ and $\delta[\ell]$ gives the result. □

The higher order correlation attack considers the best approximation of fixed degree of a Boolean function, which corresponds to determine its nonlinearity of order greater than 1, that we denote NL_d and accordingly δ_d and the recurrent versions.

Lemma 4.5.5 (Guess-and-determine and HOC)**.** *Let F be a Boolean function in N variables and C_{GDHOC} be the minimum complexity of the Guess-and-determine with High Order Correlation Attack on F, then :*

$$C_{GDHOC}(F) = \min_{0\leq\ell\leq\lambda}\left(2^{\ell}\min_{1\leq d\leq\deg(F)}\left[\binom{N-\ell}{D}^{\omega}\left(\frac{1}{2}+\delta_d[\ell]\right)\right]\right),$$

where $d \leq D \leq \frac{N-\ell}{2}$, and $m \geq \frac{\binom{N-\ell}{D}}{\binom{N-\ell}{D-d}}$, for each ℓ.

Proof. By definition:

$$C_{GDHOC}(F) = \min_{\ell}(2^{\ell}\min_{F'}C_{HOC}(F')).$$

Then for all Boolean functions f in n variables, $C_{HOC}(f) = \min_{1\leq d\leq\deg(F)}[\binom{n}{D}^{\omega}(1-\varepsilon)^{-m}])$, where:

- $d \leq D \leq n$,
- $\varepsilon = \frac{d_H(f,g)}{2^n}$ with g a Boolean function of degree at most d,
- $m \geq \frac{\binom{n}{D}}{\binom{n}{D-d}}$.

First, in our context we bound the term $(1-\varepsilon)$ using the definition of the nonlinearity of order d and δ_d, which gives $(1-\varepsilon) \geq \frac{1}{2}+\delta_d[\ell]$.

Then, the number of variables is reduced to $N-\ell$ for the algebraic system to solve as ℓ variables are fixed.

Finally we bound the term $\binom{N-\ell}{D}$ and m in consequences. Following [Cou03b] we assume $D << N$ and more precisely $D \leq \frac{N-\ell}{2}$, to avoid the decrease of the binomial coefficient affecting the complexity bound whereas this particular D does not give an attack in practice.

Putting all together we obtain the final result. Note that as for Lemma 4.5.2 and Lemma 4.5.3 a more conservative bound can be obtained considering $N[\ell]$.

□

Note that on particular functions, the parameters relatively to the recurrent criteria can be efficiently bounded, making these complexity bounds simpler and practical to give security guarantees; it will be developed in Section 5.2.2.

4.5.3 Behavior relatively to Fixed Hamming Weight

The non-standard design of the FLIP ciphers is not common to use for Boolean filtering functions: the updating process of the internal state consists in permuting the coordinates. Therefore, the Hamming weight of the internal state is constant during the whole encryption. In the four proposed instances, the Hamming weight of this register is forced to $n/2$ where n is the size of the register (n is larger than the security parameter λ, enough to ensure that $\binom{n}{n/2} \geq 2^\lambda$).

For classical filtered pseudo-random generators (for example filtered Linear Feedback Shift Registers), when the next-state function reaches all elements in $\mathbb{F}_2^n$ or $\mathbb{F}_2^n \setminus \{0\}$, the main criteria, AI, FAI, and NL, (for functions defined over $\mathbb{F}_2^n$) are relevant. Indeed, as the input is the whole space, designers can ensure that there are no extra relations on the filtering function inputs. However, if all possible inputs are not all reachable by the next-state function as in our case, then an adversary can use this restriction and the security does not directly rely on the classical criteria defined for Boolean functions over the whole $\mathbb{F}_2^n$, because the internal state itself does not reach all possible values. Then, the security analysis has to be adapted to the adequate model: the stream cipher function is only evaluated on entries from $E_{n,\frac{n}{2}}$, and the security needs to be studied relatively to the robustness of Boolean functions over $E_{n,\frac{n}{2}}$. The study of Boolean functions over $E_{n,\frac{n}{2}}$ and the security of FLIP relatively to fixed Hamming weight input presented here comes from the work [CMR17].

The purpose of this section is to stand on the attacks applying when the adversary uses the restriction of the input space jointly with previously defined attacks: classical ones and with guess-and-determine. We describe these attacks based on parameters of the proposed filtering functions relatively to fixed input weight criteria. We call ''restricted criteria" cryptographic criteria on Boolean functions which input is restricted to a subset of the whole $\mathbb{F}_2^n$. The denomination ''fixed input weight" refers to the particular case where we consider the subsets of $\mathbb{F}_2^n$ where all elements have the same Hamming weight. The restricted criteria we consider are more precisely the adaptation of the classical criteria of AI, NL and balancedness. We restrict our study to parameters relatively to these criteria, as the others are not well understood enough up to now or less applicable in the context of a fixed Hamming weight input. In the three last sections of Chapter 5 we properly define these criteria for all subset of $\mathbb{F}_2^n$ and investigate them. Here we informally introduce these criteria for constant Hamming weight input sets, and we consider the parameters of the filtering functions relatively to these criteria in order to provide a security analysis. For the sets $E_{n,k}$ and for the quantities of AI, NL and bal, we note $\mathsf{AI}_{E_{n,k}}$, $\mathsf{NL}_{E_{n,k}}$ and $\mathsf{bal}_{E_{n,k}}$ the corresponding quantities where the functions are considered only on the restricted set $E_{n,k}$ rather than $\mathbb{F}_2^n$. In the following, we explain three attacks applying on the cipher FLIP, we give lower bounds on the complexity of these attacks depending on two main parameters: the fixed Hamming weight input parameter (relatively to a criterion) of the filtering function and the Hamming weight of the key.

4.5.3.1 Algebraic Attack based on Restricted Algebraic Immunity

Assuming that an attacker wants to recover some internal state of the cipher rather than distinguishing the keystream, she could improve the so-called algebraic attacks. We briefly recall the principle defined in Subsection 4.5.1 to adapt it to the restricted algebraic immunity. The main idea for algebraic attacks (and fast algebraic attacks) is to build an over-defined system of equations with the initial state of the stream cipher as unknown, and to solve this system with Gaussian elimination. More precisely, by using a nonzero function g such that both g and $h = gf$ have low algebraic degree, an adversary is able to obtain T equations with monomials of degree at most $AI(f)$. Function g can be taken equal to an annihilator of f or of $f \oplus 1$, *i.e.* such that $gf = 0$ or $g(f \oplus 1) = 0$. The fast version consists in finding a function g with low degree and a function h with degree slightly higher than $\mathsf{AI}(f)$ which are solutions of the equation $h = gF$, providing an algebraic system easier to solve. In our context of fixed Hamming weight, the functions have to be non zero on $E_{N,k}$ and we care only at the relations holding on this set, even if they do not hold on the entire $\mathbb{F}_2^N$. Then the data complexity will drop to $D' = \sum_{i=0}^{\mathsf{AI}_k(f)} \binom{N}{i}$, the number of independent equations needed to mount a solvable algebraic system.

4.5.3.2 Correlation Attack based on Restricted Nonlinearity

In our model we target only the function F generating the keystream, in this sense the classical correlation attacks cannot work because there is no way we can do a divide-and-conquer technique as in Siegenthaler's attack [Sie84]. In this case only fast correlation attacks [MS88] could have smaller complexity than an exhaustive search in our model. Therefore we refer as correlation attack in our setting the following attack (similarly to the description of the correlation attack in Subsection 4.5.1). The attacker first computes the linear approximations l_k of f restricted to $E_{N,k}$ where $\mathsf{NL}_{E_{N,k}}(f)$ is small. Approximating the keystream equations by their linear approximation, she builds a linear system and relies it to a decoding problem. When no particular code structure is used, this system can be seen by the attacker as an instance of the Learning Parity with Noise problem, where the noise parameter is $\varepsilon_k = \frac{\mathsf{NL}_{E_{N,k}}}{\binom{N}{k}}$, (and bias $\delta_k = 0.5 - \varepsilon_k$). Standard algorithms can be used to solve this instance, as BKW [BKW00] or LF [LF06] algorithms giving an attack with data complexity of $\mathcal{O}(2^w \delta_k^{-2(x+1)})$ where the parameters w and x depend on the algorithm used and the number of variables used in l_k.

4.5.3.3 Distinguishing Attack based on Restricted Balancedness

For any stream cipher, it is important to guarantee that the keystream has good statistical properties (*i. e.* looks like a random sequence), to avoid the possibility for an attacker to distinguish the keystream from a random sequence. That is a reason why Boolean functions used in cryptography should be balanced and therefore, in a model where the Hamming weight is known, Boolean functions should be balanced on the corresponding set $E_{N,k}$. Indeed, let us denote $p_k = Pr_{x \in E_{N,k}}[f(x) = 1] = \frac{1}{2} - \varepsilon_k$. Then if there exists k such that $\varepsilon_k \neq 0$, then there exists a distinguisher on the function f for this weight. The amount of data needed to detect the bias ε_k is equal to ε_k^{-2}; If we consider all entries of f, we scale the probability of having a word of weight k, so the amount of data needed for our distinguisher to detect a bias is therefore:

$$\min_k \left(\frac{1}{{\varepsilon_k}^2} \times \frac{2^N}{\binom{N}{k}} \right)$$

As FLIP cipher always applies the filtering function on entries of constant Hamming weight k, there is no need to scale the probability; k being fixed to $N/2$, we care about $\varepsilon_{N/2}$ and $\mathsf{bal}_{E_{N,N/2}}$. However, balancedness for all weights is also important in this setting; a guess-and-determine technique consists in fixing some entries, modifying the weight. Therefore we study the balancedness criterion for Hamming weights k close to $N/2$, giving the lower bound on the data complexity of this attack: ε_k^{-2}.

4.5.4 Instances and Security

In this part we investigate the current security provided by the proposed instances, relatively to the previously described attacks. For these instances designed to provide 80 and 128 bits of security we first compute the bounds on the complexity of the standard and guess-and-determine attacks. Secondly, from [CMR17], we examine their robustness against the attacks using the input Hamming weight restriction.

4.5.4.1 Instances relatively to Standard, and Guess-and-determine Attacks

Table 4.6 lists the minimal bounds on the complexity of the attacks we described, for classical attacks with or without using a guess-and-determine strategy. These bounds are very close to the security level targeted as the functions have been chosen to have parameters tailored to the security analysis relying on these two classes of attacks. These instances are then naturally contrasted. On the one hand, the bounds taken are very conservative with respect to the considered attacks: if these attacks were the best ones, more aggressive instances could be proposed (*e.g.* in order to reduce the key size). On the other hand, Filter Permutators are based on non-standard design principles, and the security analysis could be incomplete, which naturally suggests the need of security margins. As an example, the results of the attacks based on restricted criteria at the end of this chapter, show that the security margins enabled to avoid attacks unknown when the instances were published. Overall, we believe the proposed instances are a reasonable trade-off between efficiency and security based on our current understanding of Filter Permutators, and therefore are good targets for further investigations. Their rational design and security margins enable to instantiate a proof of concept; the better understanding of the Filter Permutator and the applicable attacks coming with time will allow to reduce or modify the instances.

The complexity bounds are considered very conservative in this table for various reasons. First, the complexity of the standard attacks might be underestimated, as they are not studied exactly in the framework of our context but based on the work they appeared, generally adapted to filtered registers. Then, applying the impact of guess-and-determine attacks, we combine the worst cases of the minimal number of variables of F' and minimal AI reachable (the same technique is reproduced for the FAA and HOC attack), whereas these cases are sometimes mutually exclusive. Moreover, the data complexity of the guess-and-determine strategy is neglected, otherwise it could invalidate some guesses that correspond to the exhibited bounds, as the weakest functions can be obtained by the less probable guesses. Finally, we use rough bounds for the parameter of the functions considered, as the recurrent criteria have been recently introduced and therefore they do not enable to use exact values

Instance	N	AA	ℓ	FAA	l	CA/BKW	ℓ	HOC	ℓ	λ
FLIP − 530	530	95	56	81	0	86	72	94	55	81
FLIP − 662	662	91	52	81	52	80	72	90	48	80
FLIP − 1394	1394	156	112	140	40	134	120	155	109	134
FLIP − 1704	1704	149	105	137	105	133	124	128	74	128

Table 4.6: Attack complexities for the different FLIP instances. AA stands for algebraic attacks, FAA stands for fast algebraic attacks, CA stands for correlation or BKW-like attacks, HOC stands for higher-order correlation attacks, and ℓ stands for the optimal number of bits guessed *i.e.* leading to the best complexity for guess-and-determine attacks. For the CA/BKW column, we reported the minimum complexity between the correlation attack and the BKW-like attack. Eventually, λ stands for the security parameter of F and is simply taken as the minimum between AA, FAA,CA/BKW and HOC complexities.

Instance	N	*Bound*	D	$D^{\log(7)}$
FLIP-530	530	4	$2^{31.6}$	$2^{88.7}$
FLIP-662	662	6	$2^{46.7}$	$2^{131.1}$
FLIP-1394	1394	6	$2^{53.2}$	$2^{150.2}$
FLIP-1704	1704	8	$2^{70.6}$	$2^{198.2}$

Table 4.7: Lower bounds on $\mathsf{AI}_{E_{N,k}}$ of FLIP instances and complexity of the algebraic attack, N refers to the number of variables, *Bound* is a lower bound on $\mathsf{AI}_{E_{N,k}}$, D refers to the number of variables after linearization and $D^{\log(7)}$ to the corresponding attack complexity.

relatively to these criteria. In conclusion, a better understanding on the Filter Permutator, a more precise investigation of the algorithmic implementation of these attacks in this context and further research on the recurrent parameters could guarantee a consequently higher level of security for these instances.

4.5.4.2 Instances and Restricted Algebraic Immunity

For the FLIP family of stream cipher, we cannot conclude on the exact algebraic immunity of FLIP instances (regarding the restricted input). However, we will use a lower bound on $\mathsf{AI}_{E_{N,k}}$ proven later in Section 5.3 (using a partition of each filtering function and their structure of direct sum) to examine the potential of the attack based on $\mathsf{AI}_{E_{N,k}}$ for the instances we consider. With the lower bounds on $\mathsf{AI}_{E_{N,k}}$ for the four instances, we calculate a lower bound on the complexity of an algebraic attack on FLIP which is $\left(\sum_{i=1}^{\mathsf{AI}_{E_{N,k}}} \binom{N}{i}\right)^{\log 7}$. The results and the bounds giving us the complexity are shown in Table 4.7.

It is important to notice that the lower bounds here may be not tight and the algebraic immunity (on constant Hamming weight inputs) of FLIP instances could be as great as the general $\mathsf{AI}(F)$. Then it remains to determine the exact $\mathsf{AI}_{E_{N,N/2}}$ of FLIP instances, which we

Instance	N	v_0	ℓ	N_1	v_1
FLIP-530	530	$[107, 464]$	40	450	$[142, 383]$
FLIP-662	662	$[136, 556]$	40	582	$[189, 453]$
FLIP-1394	1394	$[221, 1239]$	64	1266	$[296, 1094]$
FLIP-1704	1704	$[266, 1492]$	64	1576	$[363, 1321]$

Table 4.8: Lower bound on $\mathsf{NL}_{E_{N,k}}$ for FLIP instances, N refers to the number of variables, v_0 the range of weight k for which $\frac{\mathsf{NL}_{E_{N,k}}(f)}{\binom{N}{k}} \geq 0.499$. ℓ refers to the number of canceled monomials of the quadratic part by the guess strategy, N_1 and v_1 are the corresponding number of variables and range of weights.

could not determine by computation due to the high number of variables of these functions.

Moreover, it remains not clear whether or not a guess-and-determine attack targeting the algebraic immunity could apply. Considering all possible guesses leads to functions where the lower bound decreases enough to contemplate an attack, but different aspects impeach us to exhibit an attack. Firstly, the guessing technique conducts to cancel or diminish the degree of some monomials. When a monomial is canceled, some variables unguessed inside could be ones, and therefore the considered function is evaluated on an input for which we cannot know exactly the Hamming weight. Secondly, the probability of obtaining a targeted weight on a targeted simpler function enough times (*i.e.* disposing of enough keystream bits with a coherent set of permutations) quickly decreases. Finally, the time complexity of the attack exhibited with the lower bound (Table 4.7) is out of reach when no guesses are made;,computational trials make us believe that the additional complexity cost of fixing ℓ variables of the filtering function (additional cost of 2^ℓ) compensate the potential decrease of $\mathsf{AI}_{E_{N,k}}$ of the obtained weaker function.

4.5.4.3 Instances and Restricted Nonlinearity

The high number of variables in the four instances makes impossible to compute exactly the nonlinearity on constant weight inputs. Nevertheless a lower bound from Section 5.4 (using the direct sum structure of the filtering functions and parameters of components simple enough) enables to state on a minimal $\mathsf{NL}_{E_{N,k}}$ of the instances and then to stand on the attack using the restricted nonlinearity. In Table 4.8 we summarize the results.

The results in Table 4.8 show that for $k = N/2$, $\mathsf{NL}_{E_{N,k}}(f)$ is high enough to ensure that the system of equations defining the LPN problem will be unsolvable with data complexity inferior to 2^λ as the error is considered as coming from a Bernoulli distribution with mean p following the relation $0.499 \leq p \leq 0.5$. Even with a guess-and-determine attack (with simulated results in the right part of the table), the $\mathsf{NL}_{E_{N,k}}$ of the various functions and the number of variables are too big to lead to a concrete cryptanalysis with an attack scenario based on $\mathsf{NL}_{E_{N,k}}$.

4.5.4.4 Instances and Restricted Balancedness

We compute the bias for each weight on toy versions of FLIP, and it appears that for all not extreme k (k close to 0 or N) the Boolean function is not balanced. Nevertheless the

Instance	N	v_0	ℓ	N_1	v_1	N_2	v_2	N_3	v_3
FLIP-530	530	$[78, 482]$	40	490	$[134, 446]$	450	$[70, 409]$	250	$[30, 190]$
FLIP-662	662	$[102, 621]$	40	622	$[178, 585]$	582	$[97, 547]$	242	$[29, 185]$
FLIP-1394	1394	$[207, 1325]$	64	1330	$[348, 1266]$	1266	$[203, 1205]$	594	$[69, 514]$
FLIP-1704	1704	$[257, 1643]$	64	1640	$[429, 1582]$	1576	$[254, 1519]$	610	$[70, 533]$

Table 4.9: Balancedness (with constant weight inputs) bias for FLIP filtering function instances, and modified instances. N is the number of variables of the instance, ℓ is the number of variables guessed to be 0. v_i stands for the range of weights with bias $< 2^{-\ell}$ for the various strategies of guesses i and N_i the number of variables of the resulting function, with v_0 the attack without guesses of the N variables function.

calculated biases are not exploitable to distinguish the output from a random sequence, regarding that we cannot have more than 2^{80} or 2^{128} bits of keystream. Using the direct sum structure of filtering functions, we can exactly compute the values $|\{x \mid f(x) = 1, \mathsf{wH}(x) = k\}|$ for all k for the four instances and the bias from a random sequence. In Table 4.9 we summarize our computation results, providing for which weights k the bias is inferior to $2^{-\frac{\lambda}{2}}$. As the impact of guess-and-determine attack on the restricted balancedness is not known, for the 4 instances we study this criterion for three particular guesses. The first guess consists in canceling (forcing to be 0) $\lambda/2$ linear variables, the second one $\lambda/2$ variables of the quadratic part and the third one $\lambda/2$ variables of the highest degree monomials. These three strategies represent the best deterioration that an attacker can obtain on one part of a FLIP filtering function. We assume that if none of these strategies reveal a sufficient bias for a concrete cryptanalysis then no hybrid approach will lead to an efficient attack.

Interpreting the results of Table 4.9, as $k = N/2$ is in the ranges of the v_0 column for the 4 instances of f, we conclude that we cannot apply our distinguisher based on the balancedness criterion as it will require more than 2^{λ} keystream bits. Even considering a combination with guess-and-determine attack, the biases on the simpler functions obtained are insufficient to mount a concrete attack.

4.5.4.5 Conclusions on Attacks based on Restricted Criteria

As a preliminary conclusion, the lower bounds exhibited for algebraic immunity, nonlinearity, and balancedness do not reveal any concrete attack on the four instances of the FLIP cipher. In this section the security analysis focusing on the Boolean function is made in the more relevant model: taken into account that the entries of the filtering function only belong to $E_{n,\frac{n}{2}}$ and not the whole $\mathbb{F}_2^n$.

On one side, we get only lower bounds on the parameters relative to the criteria of nonlinearity and algebraic immunity, leading to lower bounds on the attack complexity rather than practical bounds. On the other side, if the bounds for $\mathsf{AI}_{E_{N,k}}$ were tight, a modification of a guess-and-determine attack could be a potential attack. In conclusion, these attacks based on a better understanding of the Filter Permutator construction are a step in favor of the general security of this construction, and of the particular security of the proposed instances. As for the two other classes of attacks, a lot of improvements on the bounds on the

attack complexities and on the parameters relative to the recently defined Boolean criteria are possible. It would enable to obtain a tighter relation between provable lower bounds and concrete attack complexities, enabling to determine more efficient instances.

Chapter 5

New Criteria on Boolean Functions

In this part, we investigate new Boolean criteria, and new constructions of Boolean functions, both inspired by the design of efficient homomorphic frameworks. The restrictions on functions to produce low-error when they are homomorphically evaluated lead to consider very simple functions, and their behavior relatively to the standard cryptographic criteria. The particular design of the Filter Permutator motivates two other kinds of criteria. The recurrent criteria handle the behavior of a function when a determined part of its inputs are known. The restricted-input criteria generalize the standard criteria to functions whose inputs are restricted to a particular subset of $\mathbb{F}_2^n$.

First, we examine simple functions with good cryptographic properties and progress towards optimal constructions of low-cost functions. Then, we investigate the recurrent criteria, and more specifically the behavior of low-cost functions relatively to these criteria. Finally, we study the restricted-input criteria, separately for algebraic immunity, nonlinearity and balancedness.

Contents

In the cryptographic constructions, Boolean functions are generally used as components of symmetric encryption schemes, and have then been studied relatively to this context. We can differentiate the notions of Boolean functions and vectorial Boolean functions, the first one considering only the functions with one bit of output, which is the class of Boolean functions that we study in this chapter. The second one, larger, is more relevant in the context of block cipher constructions. Since the eighties, Boolean functions have been widely used in the context of filtered LFSR or combined registers, and more generally in most of the stream cipher constructions. In this context, the cryptographic properties required for security have been identified and extensively studied [Car10], corresponding to Boolean criteria as: high algebraic degree, high algebraic immunity, high fast algebraic immunity, high nonlinearity, balancedness, high resiliency, nonzero linear structure, *etc.* For efficient homomorphic frameworks we saw in Chapter 3 that the context is different, implying new restrictions on constructions of Boolean functions. More particularly, the commonly accepted cryptographic criteria are not adapted to state on the security of the Filter Permutator, therefore new criteria and new constructions of Boolean functions are required. The goal of this chapter is to introduce these new criteria, to propose new constructions, and to study both of them. These investigations enable to know the parameters of some functions relatively to specific Boolean criteria, which serve to derive the complexities of the best known attacks on a particular instance and to assess its security. This study also enables to consider the constructions guarantying security together with the efficiency concern of the homomorphic evaluation.

In order to introduce and study these new criteria and constructions, we proceed in the following order. First, we take care of the low-cost functions, the ones which produce low-error when they are homomorphically evaluated. Therefore, as low-cost functions, we consider the functions with a low number of monomials, more precisely we consider the ones with a linear (in the number of variables N) number of monomials in its algebraic normal form. Among these functions, we look for the ones with good behavior relatively to the standard cryptographic criteria for filtering functions: high (fast) algebraic immunity, high nonlinearity, balancedness and high resiliency. We study the robustness of particular constructions relatively to these criteria, progressing towards optimal functions in the low-cost context. Then, we introduce the recurrent criteria, extending the standard notions to multiple reduced functions, as considered in a guess ans determine attack. We study more particularly the recurrent algebraic immunity for all functions, and give particular examples. We explore the behavior of the family of Boolean functions obtained by direct sums of monomials, relatively to all recurrent criteria linked to a known attack. Finally, we investigate the behavior of Boolean functions which are evaluated only on a subset of $\mathbb{F}_2^n$, more precisely we study the restricted algebraic immunity, the restricted nonlinearity and the restricted balancedness. Based on the work [CMR17], we present a general study of these three criteria. To do so we study the parameters, or quantities, of the functions relatively to these criteria. For these quantities, we investigate their behavior for all sets, then we study more particularly their behaviors relatively to the sets formed by all inputs of a fixed Hamming weight. Finally, we focus on the fragility of some functions commonly used in cryptography which revealed to be weak in this context.

The results of the first two sections come from the article [MJSC16]. Otherwise it is specified, except the final theorem of Section 5.1.2, and Section 5.2.1 which are supplementary personal results not published yet. The results of the last three sections, corresponding to the main part of this chapter, come from [CMR17], otherwise it is specified.

5.1 Low-cost Functions

Generally, the Boolean functions used in cryptography are used for symmetric encryption and studied more particularly in adequateness with the corresponding applications. Indeed, most of the stream ciphers use filtering functions of less than 30 bits, and a lot of other functions act on registers of 64, 80, 128 or 256 bits. These functions are also chosen for their implementation, in order to be evaluated fast (in clear). Therefore, the functions known for their good behavior relatively to standard cryptographic criteria, are not the most indicated for homomorphic evaluation. First, we need to consider functions in higher number of variables, as we are looking for functions whose parameters alone guarantee the security of the scheme. It imposes to consider functions with a few hundred variables rather than less than 30 variables. Then, optimal constructions for a particular criterion are known for infinite values of N, as functions with optimal algebraic immunity for all N or functions with optimal nonlinearity for all even N. Even if a lot of work on Boolean criteria have be done, most of these constructions do not fit our low-cost consideration. A meaningful example is the family of majority functions, which have optimal algebraic immunity but a number of monomials which is exponential in N . Therefore in this section we investigate the behavior of functions with a low number of monomials, and the simple constructions, which guarantee good enough parameters for the different criteria we consider.

5.1.1 Low-cost Functions and Standard Criteria

A promising direction to obtain low-cost functions is to bound the number of monomials in the algebraic normal form of the function: it gives an upper bound on the number of additions and multiplications needed to compute it. Nevertheless, considering a very low number of monomials is quite restrictive, and not many constructions of Boolean functions enable to do so. The direct sum, a secondary construction, permits to build functions with few monomials and non trivial parameters. Indeed, the number of monomials is the sum of the number of monomials of the two components; and for the parameters, they can be computed from the parameters of the two added functions.

We begin our study by investigating the behavior of a direct sum relatively to the main Boolean criteria of a filtering function: (fast) algebraic immunity, nonlinearity and resiliency:

Lemma 5.1.1 (Direct sum properties)**.** *Let F be the direct sum of f and g with n and m variables respectively. Then F has the following cryptographic properties:*

1. *Algebraic Immunity:* $\max(\mathsf{AI}(f), \mathsf{AI}(g)) \leq \mathsf{AI}(F) \leq \mathsf{AI}(f) + \mathsf{AI}(g)$.
2. *Fast Algebraic Immunity:* $\mathsf{FAI}(F) \geq \max(\mathsf{FAI}(f), \mathsf{FAI}(g))$.
3. *Resiliency:* $\mathsf{res}(F) = \mathsf{res}(f) + \mathsf{res}(g) + 1$.
4. *Non Linearity:* $\mathsf{NL}(F) = 2^m\mathsf{NL}(f) + 2^n\mathsf{NL}(g) - 2\mathsf{NL}(f)\mathsf{NL}(g)$.

Proof. 1. We begin by the upper bound, $\mathsf{AI}(F) \leq \mathsf{AI}(f) + \mathsf{AI}(g)$:

Let h_1 be any annihilator of f or $1 + f$ of degree $\mathsf{AI}(f)$ and h_2 be any annihilator of g or $1 + g$ of degree $\mathsf{AI}(g)$, then:

$$h_1 \cdot h_2 \cdot F = 0 \text{ or } h_1 \cdot h_2 \cdot (1 + F) = 0.$$

The last concern on the function $h_1 \cdot h_2$ to state on the algebraic immunity of F is to guaranty that this function is not null (a core problem for proofs on the algebraic immunity). As f and g are functions in different variables, there exist annihilators h_1 and h_2 in the respective variables of f and g only. Then the monomials of highest degree of h_1 and h_2 are disjuncts or both equal to 1; in both cases $h_1 \cdot h_2$ is not null.

Finally, $\deg(h_1 \cdot h_2) = \mathsf{AI}(f_1) + \mathsf{AI}(f_2)$, so $\mathsf{AI}(f) \leq \mathsf{AI}(f_1) + \mathsf{AI}(f_2)$.

Now we prove the lower bound, $\mathsf{AI}(F) \geq \max(\mathsf{AI}(f), \mathsf{AI}(g))$:

We proceed by contradiction, let suppose $\mathsf{AI}(F) < \max(\mathsf{AI}(f), \mathsf{AI}(g))$. Without loss of generality we take $\max\{\mathsf{AI}(f), \mathsf{AI}(g)\} = \mathsf{AI}(f)$.

By definition of the algebraic immunity there exists h such that $F \cdot h = 0$ or $(1+F) \cdot h = 0$ with $\deg(h) = \mathsf{AI}(F)$. We first consider the case $F \cdot h = 0$.

Then, $F(x)h(x) = 0$ for all $x \in \mathbb{F}_2^{n+m}$, so *a fortiori* when the last m variables are equal to zero, denoting all x as $(x', 0, \cdots, 0)$ with $x' \in \mathbb{F}_2^n$:

$$F(x_1, \cdots, x_n, 0, \cdots, 0)h(x_1, \cdots, x_n, 0, \cdots, 0) = f(x')h(x', 0, \cdots, 0) = 0.$$

Hence $h(x_1, \cdots, x_n, 0, \cdots, 0)$ is a function in n variables and degree strictly less than $\mathsf{AI}(f)$ annihilating f, therefore this function is null or we get a contradiction. Note that $h(x_1, \cdots, x_n, 0, \cdots, 0)$ being null does not imply the nullity of h, therefore we consider all the potential values of the m last variables:

$$\forall u \in \mathbb{F}_2^m, \quad 0 = h(x_1, \cdots, x_n, u) \cdot (x_1, \cdots, x_n, u) = h(x_1, \cdots, x_n, u) \cdot (f + \varepsilon),$$

where $\varepsilon \in \{0, 1\}$.

Consequently, for all $u \in \mathbb{F}_2^m$, $h(x_1, \cdots, x_n, u)$ is an n-variables function of degree strictly less than $\mathsf{AI}(f)$ annihilating f or $f + 1$ or the null function. If at least one of the functions is a non null annihilator then it contradicts the supposition on $\mathsf{AI}(F)$. otherwise, all these functions are null, implying that the $n + m$ variables function h is null, which is also a contradiction.

We considered the case $F \cdot h = 0$; the other case where h annihilates $F + 1$ is proved with the same reasoning, changing F into $F + 1$ and f into $f + 1$ in the previous equations. It concludes this item of the proof.

2. The proof follows the same techniques as the precedent proof; if the FAI of the direct sum is strictly less than the one of one of its components, then it enables to find a function h in n (or m) variables contradicting the definition of FAI.

3. For this item and the next one the proof is given for instance in [Car10] (page 125), for self containment we give here a detailed proof.

   Let consider the Walsh transform of F, defined for all $u \in \mathbb{F}_2^{n+m}$. For each element u we use the following partition: we denote a the first n bits and b the last m bits, then a and b are uniquely defined. Denoting u as (a, b), enables to derive the following equalities:

$$\begin{aligned}\hat{F}_\chi(u) &= \sum_{x\in\mathbb{F}_2^{n+m}} (-1)^{F(x)+u\cdot x} \\ &= \sum_{(y,z)\in\mathbb{F}_2^n\times\mathbb{F}_2^m} (-1)^{f(y)+g(z)+a\cdot y+b\cdot z} \\ &= \sum_{y\in\mathbb{F}_2^n} \left(\sum_{z\in\mathbb{F}_2^m} (-1)^{f(y)+a\cdot y+g(z)+b\cdot z} \right) \\ &= \sum_{y\in\mathbb{F}_2^n} (-1)^{f(y)+a\cdot y} \left(\hat{g}_\chi(b)\right) \\ &= \hat{f}_\chi(a)\ \hat{g}_\chi(b).\end{aligned}$$

The expression of the Walsh transform of F as the product of the Walsh transforms of f and g can be used together with Theorem 2.4.8. As the Walsh transform is equal to 0 for all vectors of Hamming weight less than or equal to the resiliency, we study the value of the Walsh transform of F for any vector u of Hamming weight less than or equal to $\mathsf{res}(f)+\mathsf{res}(g)+1$.

For all vectors $u\in\mathbb{F}_2^{n+m}$, written as (a,b), the restriction $\mathsf{w_H}(u)\leq \mathsf{res}(f)+\mathsf{res}(g)+1$ can be separated in two cases: either $\mathsf{w_H}(a)\leq \mathsf{res}(f)$, or $\mathsf{w_H}(a)> \mathsf{res}(f)$.

In the first case, $\mathsf{w_H}(a)\leq \mathsf{res}(f)$ then $\hat{f}_\chi(a)=0$, giving $\hat{F}_\chi(u)=0$. In the second case, $\mathsf{w_H}(a)> \mathsf{res}(f)$ then $\mathsf{w_H}(b)\leq \mathsf{res}(g)$, giving $\hat{g}_\chi(b)=0$, and finally $\hat{F}_\chi(u)=0$.

From these two cases, we conclude:

$$\mathsf{res}(F)\geq \mathsf{res}(f)+\mathsf{res}(g)+1.$$

To get the equality it is sufficient to consider two elements, one for f and one for g. As f is not $\mathsf{res}(f)+1$ resilient, we can choose an element $a\in\mathbb{F}_2^n$ such that $\mathsf{w_H}(a)=\mathsf{res}(f)+1$ and $\hat{f}_\chi(a)\neq 0$. Similarly, we can choose an element $b\in\mathbb{F}_2^m$ such that $\mathsf{w_H}(b)=\mathsf{res}(g)+1$ and $\hat{g}_\chi(b)\neq 0$.

Therefore, we can choose an element $u\in\mathbb{F}_2^{n+m}$, $u=(a,b)$ such that:

$$\mathsf{w_H}(u)=\mathsf{res}(f)+\mathsf{res}(g)+2,\ \text{and}\ \hat{F}_\chi(u)\neq 0.$$

Ensuring $\mathsf{res}(F)<\mathsf{res}(f)+\mathsf{res}(g)+2$, finalizing this item of the proof.

4. Using the second part of Definition 2.4.9, the nonlinearity can be obtained from the maximum of the Walsh transform:

$$\mathsf{NL}(F)=2^{n+m-1}-\frac{1}{2}\max_{u\in\mathbb{F}_2^{n+m}}|\hat{F}_\chi(u)|.$$

From the proof of the precedent item (on the resiliency), we know:

$$\hat{F}_\chi(u)=\hat{f}_\chi(a)\hat{g}_\chi(b),$$

therefore:

$$\begin{aligned}\max_{u\in\mathbb{F}_2^{n+m}} \hat{F}_\chi(u) &= \max_{a\in\mathbb{F}_2^n} \hat{f}_\chi(a) \cdot \max_{b\in\mathbb{F}_2^m} \hat{g}_\chi(b),\\ &= (2^n - 2\mathsf{NL}(f)) \cdot (2^m - 2\mathsf{NL}(g)).\end{aligned}$$

Finally, it gives:

$$\begin{aligned}\mathsf{NL}(F) &= 2^{n+m-1} - \frac{1}{2}\left(2^{n+m} - 2^{n+1}\mathsf{NL}(g) - 2^{m+1}\mathsf{NL}(f) + 4\mathsf{NL}(f)\mathsf{NL}(g)\right),\\ &= 2^m\mathsf{NL}(f) + 2^n\mathsf{NL}(g) - 2\mathsf{NL}(f)\mathsf{NL}(g),\end{aligned}$$

concluding the proof.

□

Based on this lemma, we can hope to find functions with good cryptographic properties and few monomials using direct sum constructions. However, these constructions require to find functions in a few variables good, or even optimal, for a particular criterion. Indeed, for the 4 criteria we study, the direct sum construction enhance the parameter of the better function: the parameter of the final function is at least as high as the one of the better function. Then, finding a strong function in a few variables and with a few monomials for each criterion enables to build a function in more variables with still a quite low number of monomials, and good parameters relatively to the 4 criteria. Functions with a low number of monomials (linear in the number of variables) and optimal relatively to resiliency and nonlinearity are well-known in the (Boolean functions) literature, and we will present them slightly later. For the algebraic immunity and fast algebraic immunity criteria, way less is known in this direction; the functions known for optimal algebraic immunity are the majority functions (Definition 2.4.18) or variants and the Carlet-Feng functions [CF08]. Both are functions with ANF containing a high number of monomials (exponential in the number of variables), then inappropriate for the low-cost constructions we consider. To circumvent this issue, in [MJSC16] we introduce the triangular functions: functions with a low number of monomials (proportional to the square root of the number of variables), and good algebraic and fast algebraic immunities.

We examine the different parameters of a triangular function (Definition 2.4.16)in the following lemma. We particularly focus on the algebraic immunity, as it is the core motivation of this particular construction.

Lemma 5.1.2 (Triangular functions properties)**.** *Let k be a non null positive integer and let T_k the k-th triangular function. Then the following properties hold:*

1. *Algebraic Immunity:* $\mathsf{AI}(T_k) = k$.
2. *Fast Algebraic Immunity:* $\mathsf{FAI}(T_k) = k + 1$.
3. *The Nonlinearity follows the recursive formula defined by:*
   (i) $\mathsf{NL}(T_1) = 0$,
   (ii) $\mathsf{NL}(T_{k+1}) = (2^{k+1} - 2)\mathsf{NL}(T_k) + 2^{k(k+1)/2}$.
4. *resiliency:* $\mathsf{res}(T_k) = 0$.

1. We give here a proof simpler than the one presented in [MJSC16]. We proceed by induction on k, showing that for all $k \in \mathbb{N}^*, \mathsf{AI}(T_k) = k$.

   For $k = 1$ we have $T_1 = x_1$ of algebraic immunity $\mathsf{AI}(x_1) \geq 1$ by definition. Hence $1 + x_1$ is a degree 1 annihilator of x_1, giving $\mathsf{AI}(T_1) = 1$.

   For $k \to k + 1$, by the induction hypothesis we have that $\mathsf{AI}(T_k) = k$. We want now to determine $\mathsf{AI}(T_{k+1})$ where $T_{k+1} = T_k + \Pi$ where Π is the product of the $k + 1$ variables which are not in T_k. Π is a monomial of degree $k + 1$, therefore its algebraic immunity is 1 (Corollary 2.4.11). By Lemma 5.1.1 we get:

   $$k \leq \mathsf{AI}(T_{k+1}) \leq k + 1.$$

   We prove the induction step by contradiction: suppose that $\mathsf{AI}(T_{k+1}) = k$, then $\exists h$ such that $\deg(h) = k$ and $h \cdot T_{k+1} = 0$ or $h \cdot (1 + T_{k+1}) = 0$. Hence we have two different cases, we first consider the case $h \cdot T_{k+1} = 0$.

   For readability, let use note $x_1 \cdots x_n$, the n variables of T_k and $y_1, \cdots, y_{k+1}$ the $k + 1$ variables of Π. For all $(x, y) \in \mathbb{F}_2^{n+k+1}$ the following holds:

   $$h(x_1, \cdots, x_n, y_1, \cdots, y_{k+1})(T_k(x_1, \cdots, x_n) + \Pi(y_1, \cdots, y_n)) = 0.$$

   Therefore, fixing the $k + 1$ variables of the Π part, for all $y \in \mathbb{F}_2^{k+1}$ we get:

   $$h(x_1, \cdots, x_n, y)(T_k(x_1, \cdots, x_n) + \varepsilon) = 0,$$

   where $\varepsilon = 1$ if $y = (1, \cdots, 1)$ and $\varepsilon = 0$ for all the other values.

   Then, for all y, $h(x_1, \cdots, x_n, y)$ is a n-variable function of degree less than or equal to k (due to our supposition on the degree of h), annihilator of T_k or annihilator of $1 + T_k$. Using the induction hypothesis, either this function is null, or its degree is exactly k. If for all y this function is null, the entire function h is null, which contradicts our assumption on h. Otherwise, for at least one value of y, $h(x_1, \cdots, x_n, y)$ is not null, hence its ANF contains at least one monomial of degree k in the n variables of T_k. Therefore, the ANF of h contains at least one monomial with exactly k of the n variables of T_k, this monomial does not contain any other variables as we assume $\deg(h) = k$; giving that h's ANF contains at least one degree k monomial in the n variables of T_k. Note that if there is more than one monomial of this kind, they are distinct, by definition of the ANF.

   The existence of a monomial of degree k only in the n variables of T_k in h's ANF enables to get the contradiction. Indeed, for each monomial of this kind, the product of h times Π, produces then a monomial of degree $2k + 1$ which cannot be canceled by any monomial of $h \cdot T_k$, giving $h \cdot (T_k + \Pi) \neq 0$, contradicting the hypothesis.

   Now, we consider the case $h \cdot (T_{k+1} + 1) = 0$, the previous reasoning holds, changing T_k to $1 + T_k$. Therefore, we can conclude, the assumption $\mathsf{AI}(T_{k+1}) = k$ leads to a contradiction; then $\mathsf{AI}(T_{k+1}) = k + 1$.

   It finalizes the proof by induction, giving the final result.

2. We proceed by contradiction, let assume that $\mathsf{FAI}(T_k) < k + 1$.

By definition we have:

$$\min\{2AI(T_k), \max\{\deg(g \cdot T_k) + \deg(g), 3\deg(g)\} \mid 1 \leq \deg(g) < \mathsf{AI}(T_k)\} < k+1.$$

Applying the first item of Lemma 5.1.2 (proven just above) we have:

$$\min\{2k, \max\{\deg(g \cdot T_k) + \deg(g), 3\deg(g)\} \mid 1 \leq \deg(g) < k)\} < k+1.$$

Hence, it implies that:

$$\exists g \mid \deg(g \cdot T_k) + \deg(g) < k+1 \quad \text{and} \quad 3 \cdot \deg(g) < k+1 \quad \text{and} \quad 1 \leq \deg(g) < k.$$

In the following we prove that such g does not exist: either the restrictions on the degree of g or the properties of the algebraic immunity are violated.

We begin by considering $T_k + 1$, which is an annihilator of $g \cdot T_k$. Then, $T_k(gT_k + g) = 0$, it implies that $g \cdot T_k + g = 0$ or $\deg(g \cdot T_k + g) \geq k$.

If $g \cdot T_k + g = 0$, we have either $g = 0$ or $\deg(g) \geq k$, both giving in contradiction with the restrictions on the degree of g.

If $\deg(gT_k + g) \geq k$ then we get $\deg(g) \geq k$ or $\deg(gT_k) \geq k$, where the first inequality leads to a contradiction. The second inequality, $\deg(gT_k) \geq k$, combined with the restriction on the degree of g, $\deg(g) \geq 1$, leads to $\deg(g \cdot T_k) + \deg(g) \geq k+1$ which is in contradiction with the first condition on g.

To sum up, the assumption $\mathsf{FAI}(T_k) < k+1$ always leads to a contradiction, so $\mathsf{FAI}(T_k) \geq k+1$.

For the exact value of the fast algebraic immunity, let us denote by x_ℓ one of the variables contained in the monomial of degree k. We can take the degree-one function $g = 1 + x_\ell$. The product of g by T_k cancels the highest monomial of T_k, giving $\deg(g \cdot T_k) \leq k$. As the second highest degree monomial of T_k (for $k > 1$) does not contain x_ℓ, $g \cdot T_k$ has a monomial of degree k. These two remarks gives:

$$\deg(g \cdot T_k) + \deg(g) = k+1.$$

Note that in these cases, $\max\{\deg(g \cdot T_k) + \deg(g), 3\deg(g)\}$ is equal to the first value if k is not less than 2.

To sum up, for $k > 1$ in the previous paragraph we exhibit a function ensuring $\mathsf{FAI}(T_k) \leq k+1$. For the particular case $k = 1$, as $\mathsf{AI}(T_1) = 1$, the definition of the FAI imposes $\mathsf{FAI}(T_1) = 2$. Combining the two inequalities, we conclude: $\mathsf{FAI}(T_k) = k+1$.

3. Here we proceed by induction.

   For $k = 1$, T_1 is a linear function so considering T_1 as its own best affine approximation we get $\mathsf{NL}(T_1) = 0$.

   For $k \to k+1$, we use the fourth item of Lemma 5.1.1, using the following decomposition:

$$\forall k \in \mathbb{N}^*, \quad T_{k+1} = T_k + \Pi,$$

   where Π is the monomial of degree $k+1$ in the last $k+1$ variables.

Denoting n the number of variables of T_k (note that $n = k(k+1)/2$), we obtain the following equalities:

$$\begin{aligned}\mathsf{NL}(T_{k+1}) &= \mathsf{NL}(T_k + \Pi),\\ &= 2^{k+1}\mathsf{NL}(T_k) + 2^n\mathsf{NL}(\Pi) - 2\mathsf{NL}(T_k)\mathsf{NL}(\Pi),\\ &= (2^{k+1} - 2)\mathsf{NL}(T_k) + 2^n, \text{ as } \mathsf{NL}(\Pi) = 1 \text{ for all } k \geq 1.\end{aligned}$$

It concludes the induction, and finishes the proof of this item.

4. We proceed by induction, let suppose that for $k \geq 1, \mathsf{res}(T_k) = 0$.

   For $k = 1$, $T_1 = x_1$; it is a linear non null $1-$variable function, therefore balanced and with resiliency 0.

   For $k \to k+1$, The induction hypothesis gives that T_k has resiliency 0, then as by construction, for $k \geq 1$, T_{k+1} is the direct sum of T_k and a monomial of degree $k+1$, we can use the third item of Lemma 5.1.1. As a monomial function of degree strictly greater than one is unbalanced, its resiliency is -1. The lemma gives then:

$$\mathsf{res}(T_{k+1}) = 1 + \mathsf{res}(T_k) + (-1) = \mathsf{res}(T_k) = 0.$$

   In conclusion, for all $k \in \mathbb{N}^*$ we conclude: $\mathsf{res}(T_k) = 0$.

Some important particularities of the triangular functions can be observed, on its simplicity and its behavior relatively to the algebraic immunity. First, a triangular function is a direct sum of monomials, hence it can be represented by a direct sum vector (of k ones), and computed by a Boolean circuit of $N-1$ gates, where N is the number of variables and k the degree of the function. Regarding the algebraic immunity criterion, this construction maximizes the ratio between algebraic immunity and number of monomials in the ANF. Indeed, an ANF of ℓ monomials implies an algebraic immunity less than or equal to ℓ, as each monomial can be canceled by a degree one function. The triangular function has not an optimal algebraic immunity then, as this parameter is equivalent to its number of monomials, which is proportional to the root of N. Nevertheless, it can be observed that for the same parameter of algebraic immunity, a triangular function can be evaluated more efficiently than a majority function. For this parameter being equal to k, T_k can be evaluated by a circuit of $(k-1)k/2$ AND gates and $k-1$ XOR gates, with a multiplicative depth of $\log k$. In comparison, the circuit represented in Figure 3.2 requires $(k+1)k-3$ AND gates and $(2k-1)(k-1)$ XOR gates (counting one AND and two XOR for one MUX).

The triangular function is an important step towards our low-cost construction with high cryptographic parameters. More precisely this low-cost function provides good algebraic and fast algebraic immunities with very few monomials. It cannot be directly used as a filtering function as its nonlinearity and resiliency are too low. Nevertheless, the direct sum construction enables to complement this function with already know low-cost functions in order to get functions strong and low-cost, that we call FLIP functions. For the nonlinearity criterion, the Dickson function in $2n$ variables are bent functions of degree 2 and their ANF contain only n monomials. Regarding the resiliency criterion, the elementary symmetric function in n variables of degree 1 is $n-1$ resilient, we note that it is the optimal function in terms of monomials and resiliency. Siegenthaler's bound gives that an n-variable function is

at most $n-1$ resilient and the equality implies that the function is affine; the parameter $n-1$ implies that all variables have to appear in the ANF, therefore the symmetric function we consider minimizes the number of monomials relatively to the optimal resiliency.

We conclude this part by giving the parameters of the FLIP functions, relatively to the cryptographic criteria we consider and the number of monomials in the ANF. Note that the repetitive use of the direct sum construction gives a direct sum of monomials, an N-variable function which can always be evaluated in at most $N-1$ AND and XOR gates.

Lemma 5.1.3 (FLIP functions properties)**.** *Let F bean N-variable FLIP function defined by n_1, n_2, nb and h or equivalently by the direct sum vector of length h:$[n_1+nb, n_2/2+nb, nb, \cdots, nb]$. Then the following properties hold:*

1. *Number of monomials of F: $n_1 + n_2/2 + nb \cdot h$.*
2. *Algebraic Immunity: $\mathsf{AI}(F) = h$.*
3. *Fast Algebraic Immunity: $\mathsf{FAI}(F) \geq h+1$.*
4. *Nonlinearity: $NL(F) \geq 2^{N-1} - 2^{N-n_2/2-nb-1}$.*
5. *resiliency: $\mathsf{res}(F) = n_1 + nb - 1$.*

Proof. The proof follows quite directly the ideas of the previous two lemmata: Lemma 5.1.1 and Lemma 5.1.2, we briefly recall here the core ideas.

1. Summing all the elements of the direct sum vector we obtain the number of monomials: $n_1 + n_2/2 + nb \cdot h$. We can also count that the ANF of L_{n_1} has n_1 monomials, the ANF of $Q_{n_2/2}$ contains $n_2/2$ monomials and each of the nb triangular functions has an ANF with h monomials.

2. We can write F as the direct sum of T_h and another function, which gives the lower bound $\mathsf{AI}(F) \geq h$ using the algebraic immunity items of Lemma 5.1.1 and Lemma 5.1.2. As F is a degree h function, we conclude $\mathsf{AI}(F) = h$.

3. Lemma 5.1.2 guarantees $\mathsf{FAI}(T_h) = h+1$; applying Lemma 5.1.1 to the same decomposition in direct sum as the previous item we obtain $\mathsf{FAI}(F) \geq h+1$.

4. Note that the exact nonlinearity of L_{n_1}, $Q_{n_2/2}$ and of each triangular function T_h can be computed and the exact value of $\mathsf{NL}(F)$ can be recursively computed using Lemma 5.1.1. In this lemma we give only an upper bound here for readability, considering only the nonlinearity provided by the degree two monomials.

   Denoting G the part of F containing all the degree two monomials (so G is a $n_2 + 2nb$ variables function) and H the remaining part, we can write F as the direct sum of G and H and apply Lemma 5.1.1:

$$\begin{aligned}
\mathsf{NL}(F) &= 2^{n_2+2nb}\,\mathsf{NL}(H) + 2^{N-n_2-2nb}\,\mathsf{NL}(G) - 2\mathsf{NL}(G)\,\mathsf{NL}(H) && (5.1)\\
&= (2^{n_2+2nb} - 2\mathsf{NL}(G))\,\mathsf{NL}(H) + 2^{N-n_2-2nb}\,\mathsf{NL}(G) && (5.2)\\
&\geq 2^{N-n_2-2nb}\,\mathsf{NL}(G) && (5.3)\\
&\geq 2^{N-1} - 2^{N-n_2/2-nb-1}. && (5.4)
\end{aligned}$$

We obtain Equation (5.3) because the nonlinearity of a n variables function is positive and smaller than 2^{n-1}, ensuring here that the first summand is positive. Equation (5.4) comes from the fact that G (a Dickson function) is bent; it enables to conclude for this item.

5. We can obtain this result only considering the resiliency of monomial functions and Lemma 5.1.1. The resiliency of a monomial function is 0 if the monomial has degree 1, -1 otherwise. Therefore, applying Lemma 5.1.1, the $n_1 + nb$ degree one monomials contribute to a resiliency of $n_1 + nb - 1$ and the other monomials do not modify the resulting resiliency; giving the final result: $\mathsf{res}(F) = n_1 + nb - 1$.

□

This lemma totally quantifies the parameters of the FLIP functions relatively to the cryptographic Boolean criteria, it enables to link the mathematical properties of these low-cost functions to their security in the contexts we consider.

5.1.2 Exact Algebraic Immunity of Direct Sums of Monomials

The previous part shows the good behavior of direct sums of monomials as low-cost cryptographic Boolean functions. More particularly, Lemma 5.1.1 enables to derive the exact nonlinearity and resiliency of all functions of this class, nevertheless the exact algebraic immunity is only proven for the sub-cases of triangular functions and FLIP functions. In this part we prove the exact algebraic immunity of all functions obtained by direct sums of monomials. The exact algebraic immunity is not known for a large variety of families of functions, this study increases this knowledge for a small family. The number of "shapes" (only the distinct direct sum vectors) of direct sum functions of N variables corresponds to the number of integer partitions of N, and for all N this number can be derived exactly from Euler's pentagonal number theorem.

In order to prove the exact value of the algebraic immunity of a direct sum of monomials we need the following steps. First, we use the result on the algebraic immunity of a triangular function: $\mathsf{AI}(T_k) = k$. Then, we introduce a proposition connecting the algebraic immunity of two functions. Finally, we give the theorem standing on the result, linking all direct sums of monomials to triangular functions.

The triangular function T_k has as associated direct sum vector the vector of length k with all m_i being equal to 1. In the previous part we prove $\mathsf{AI}(T_k) = k$, giving an AI equal to the number of monomials of this function. The algebraic immunity of FLIP functions shows that we can extend the knowledge of the algebraic immunity of the triangular function to some other direct sums of monomial. Indeed, for the functions obtained by a direct sum of a triangular function of degree d and another function of degree less than or equal to d, the algebraic immunity is exactly d. However, this result does not cover all direct sums of monomials, for example it cannot apply to a function obtained by withdrawing a monomial of a triangular function, neither to a direct sum of d monomials of degree d. To take care of all these functions, in [CMR17] we introduce a proposition linking the algebraic immunity of two related functions. The proposition guarantees that the AI of the function in fewer variables is less than or equal to the AI of the first function, or equivalently that the AI of the function in more variables is at least as high as the AI of the second function. In the particular case of direct sums of monomials this relation can be explained using their direct

sum vector. The second vector is simply obtained by decreasing by 1 one of the m_i (with $i > 1$ and $m_i > 0$) and adding one to m_{i-1}; it illustrates the contraction of two variables in one. Using repetitively this proposition, all direct sums of monomials can be linked to a triangular function, providing an upper bound or a lower bound on its algebraic immunity.

Proposition 5.1.4. *Let $f(x_1, x_2, x_3, \cdots, x_n)$ be a Boolean function in n variables such that there exists two variables (x_1 and x_2 without loss of generality) satisfying:*

$$\forall x \in \mathbb{F}_2^{n-2}\ f(0,0,x) = f(0,1,x) = f(1,0,x)$$

Let $F(X, x_3, \cdots, x_n)$ be the Boolean function in $n-1$ variables defined by :

$$\forall x \in \mathbb{F}_2^{n-2}\ F(1,x) = f(1,1,x) \text{ and } F(0,x) = f(0,0,x)$$

If $\mathsf{AI}(f) \leq d$ then $\mathsf{AI}(F) \leq d$.

Proof. Formally we prove that if there exists a non null function g in n variables of degree $\leq d$ such that $fg = 0$ (respectively $(f+1)g = 0$) over the whole $\mathbb{F}_2^n$ then there exists a non null function G in $n-1$ variables of degree $\leq d$ such that $FG = 0$ (respectively $(F+1)G = 0$) over $\mathbb{F}_2^{n-1}$.

First we decompose g in a unique way:

$$g = g(x_1, x_2, \cdots, x_n) = g_1 x_1 + g_2 x_2 + g_{mix} x_1 x_2 + g_{none},$$

where:

- $g_{mix} x_1 x_2$ contains all monomials with both x_1 and x_2,
- $g_1 x_1$ contains all monomials with x_1 and without x_2,
- $g_2 x_2$ contains all monomials with x_2 and without x_1,
- g_{none} contains all monomials without x_1 and without x_2.

As g is non null there exists at least one $\bar{x} \in \mathbb{F}_2^{n-2}$ such that for at least one of the four entries $(0,0,\bar{x}), (0,1,\bar{x}), (1,0,\bar{x})$ or $(1,1,\bar{x})$ the function g is not null. Therefore we realize a disjunction of cases, for the four possible values of (x_1, x_2) we build a different function G. In each case G is an annihilator of F (respectively $F+1$) in $n-1$ variables of degree less than or equal to d based on the fact that the function in $n-2$ variables defined $\forall x \in \mathbb{F}_2^{n-2}$ as $g(x_1, x_2, x)$ is non null and with degree less than or equal to d.

- Case $g(1,1,x)$ is not the null function:

  $$\forall x \in \mathbb{F}_2^{n-2}, \quad g(1,1,x) = g_1 + g_2 + g_{mix} + g_{none},$$

  then, we can define G as

  $$G(X,x) = (g_1 + g_2 + g_{mix})X + g_{none}.$$

  Therefore, $\forall x \in \mathbb{F}_2^{n-2}$:

  if $X = 0$ then $F(X,x)G(X,x) = f(0,0,x)g_{none} = f(0,0,x)g(0,0,x) = 0$,

  else $X = 1$ then $FG = f(1,1,x)(g_1 + g_2 + g_{mix} + g_{none}) = f(1,1,x)g(1,1,x) = 0$.

  In both cases G is an annihilator of F (respectively of $1+F$), non null because $g(1,1,x)$ is non null and of degree less than or equal to d as the degree of g_1 and g_2 are upper bounded by $d-1$, the degree of g_{mix} is upper bounded by $d-2$ and the degree of g_{none} is upper bounded by d.

- Case $g(1,0,x)$ is not the null function:
  we define $G(X,x) = g_1 + g_{none} + (g_2 + g_{mix})X$, then $\forall x \in \mathbb{F}_2^{n-2}$:
  if $X = 0$, then
  $$F(X,x)G(X,x) = f(0,0,x)(g_1 + g_{none}) = f(1,0,x)g(1,0,x) = 0,$$
  else $X = 1$, then
  $$F(X,x)G(X,x) = f(1,1,x)(g_1 + g_2 + g_{mix} + g_{none}) = f(1,1,x)g(1,1,x) = 0.$$
  We can conclude similarly than the previous item.
- Case $g(0,1,x)$ is not the null function:
  we define $G(X,x)$ as $g_2 + g_{none} + (g_1 + g_{mix})X$; we only switch the impact of x_1 and x_2 from the previous item.
- Case $g(0,0,x)$ is not the null function:
  we define $G(X,x) = (g_1 + g_2 + g_{mix})X + g_{none}$ as for the first item and as $g(0,0,x)$ is not null in this case we can conclude in the same way.

We conclude that for all f with this property relatively to two variables there exists a function G with the described property, and therefore $\mathsf{AI}(f) \leq d$ then $\mathsf{AI}(F) \leq d$.

□

In this section we care only on direct sums of monomials, however we highlight the general character of the precedent proposition: this contraction technique can be used for functions with a higher number of monomials. Other contractions could lead to generic techniques to prove the exact algebraic immunity of family of functions based on the knowledge of the AI of simple functions. In the following we use this proposition as a tool to quantify the algebraic immunity of direct sums of monomials, we express the exact value using the coefficients of the direct sum vector representation. To obtain interesting bounds from the proposition on a function f, we need to target functions with AI known and we need to find a path from our n-variable function to the target function. For the case of direct sums of monomials, we can illustrate it through the direct sum vector, showing the link between the vectors of f and F and the shape of the vectors for which the corresponding AI is known.

As an illustration, let us consider f a Boolean function obtained by direct sums of monomials, and the associated direct sum vector $\mathbf{m}_f = [m_1, \cdots, m_k]$. We can apply the proposition on all m_i such that $i > 1$ and $m_i > 0$, giving:

$$\mathbf{m}_F = [m_1, \cdots, m_{i-1} + 1, m_i - 1, \cdots, m_k] \text{ and } \mathsf{AI}(f) \leq d \Rightarrow \mathsf{AI}(F) \leq d.$$

The direct sums of monomials for which the AI is known are the triangular functions, and the direct sums of a triangular function and a function of lower (or similar) degree, corresponding to the following vectors $\mathbf{m}_{T_k}$ and $\mathbf{m}_H$:

$$\mathbf{m}_{T_k} = [m_1, \cdots, m_k] = [1, \cdots, 1], \text{ and } \mathsf{AI}(T_k) = k.$$

$$\mathbf{m}_H = [m_1, \cdots, m_k], \text{where } \forall i \in [1,k]\ m_i \geq 1, \text{ and } \mathsf{AI}(H) = k.$$

For direct sums of monomials, a way to determine the exact algebraic immunity of a function consists in modifying its direct sum vector using the previous proposition until we get a vector of one of the two shapes just above. It finally gives the exact algebraic immunity if we identify an annihilator reaching the bound given by the proposition. Therefore we can now enunciate the theorem on the algebraic immunity of direct sums of monomials.

Theorem 5.1.5 (Algebraic Immunity of Direct Sums of Monomials)**.** *Let $f \in \mathbb{F}_2^n$ be a Boolean function obtained by direct sums of monomials with associated direct sum vector $\mathbf{m}_f = [m_1, \cdots, m_k]$, its algebraic immunity is:*

$$\mathsf{AI}(f) = \min_{0 \leq d \leq k} \left(d + \sum_{i=d+1}^{k} m_i \right).$$

Proof. First, we prove the inequality:

$$\mathsf{AI}(f) \leq \min_{0 \leq d \leq k} \left(d + \sum_{i=d+1}^{k} m_i \right).$$

For this inequality it is sufficient to show that there exists a non null annihilator of f of degree $d + \sum_{i=d+1}^{k} m_i$ for all d such that $0 \leq d \leq k$. We fix d, and express f as a direct sum of two functions f_1 and f_2, with direct sum vectors:

$$\mathbf{m}_{f_1} = [m_1, \cdots, m_d], \text{ and } \mathbf{m}_{f_2} = [0, \cdots, 0, m_{d+1}, \cdots, m_k].$$

From $\mathbf{m}_{f_1}$ we have $\deg(f_1) = d$, therefore the degree-d function $f_1 + 1$ is an annihilator of f_1. Then, for each of the monomials of f_2, we choose a variable x_j appearing in this monomial. As $1 + x_j$ is a degree-1 annihilator for all monomial functions containing x_j, f_2 the direct sum f_2 of $\sum_{i=d+1}^{k} m_i$ monomials admits as annihilator the product of the annihilator of each individual monomial. Hence, the function obtained by multiplying the appropriate $1 + x_j$ is an annihilator of f_2, as f_2 is obtained by direct sums of monomials all $1 + x_j$ are distinct therefore their product has degree $\sum_{i=d+1}^{k} m_i$. Denoting this function h which is non zero by construction we obtain the following:

$$(f_1 + f_2) \cdot (f_1 + 1)h = 0, \text{ with } (f_1 + 1)h \neq 0, \text{ and } \deg((f_1 + 1)h) = d + \sum_{i=d+1}^{k} m_i.$$

Therefore, considering all d such that $0 \leq d \leq k$ gives the inequality. Note that this inequality shows, for the case of direct sums of monomials, that the algebraic immunity is upper bounded both by the number of monomials (case $d = 0$) and the degree of f (case $d = k$).

Then, we prove the second inequality:

$$\mathsf{AI}(f) \geq \min_{0 \leq d \leq k} d + \sum_{i=d+1}^{k} m_i.$$

Let us denote by e the integer between 0 and k giving the minimal value of the sum, we take e as the smallest element if multiple integers lead to the minimal value.

So, for all integers j such that $1 \leq j \leq e$:

$$e - j + \sum_{i=e-j+1}^{k} m_i \geq e + \sum_{i=e+1}^{k} m_i,$$

which is equivalent to:

$$\sum_{i=e-j+1}^{e} m_i \geq j.$$

This inequality holds for all j such that $1 \leq j \leq e$; for $j = 1$ it guarantees that m_e is non null, for $j = 2$ it gives that $m_e + m_{e-1} \geq 2$, so one and so forth until $m_1 + \cdots + m_e \geq e$. Therefore it guarantees that we can repetitively apply Proposition 5.1.4 on f to obtain a function with direct sum vector having all m_i non null for $1 \leq i \leq e$, which already gives $\mathsf{AI}(f) \geq e$. A constructive way can consist in contracting all but one monomials of each degree less than or equal to e, beginning by the degree-e monomials. It would lead to a function F with direct sum vector:

$$\mathbf{m}_F = \left[\left(\sum_{i=1}^{e} m_i\right) - e + 1, \cdots, 1, m_{e+1}, \cdots, m_k\right].$$

Now, let us consider the monomials of degree higher than e; for all integers j such that $1 \leq j \leq k - e$:

$$e + j + \sum_{i=e+j+1}^{k} m_i \geq e + \sum_{i=e+1}^{k} m_i,$$

which is equivalent to:

$$j \geq \sum_{i=e+1}^{e+j} m_i.$$

This inequality holds for all j such that $1 \leq j \leq k-e$; for $j = 1$ it guarantees that $m_{e+1} \leq 1$, for $j = 2$ it gives that $m_{e+1} + m_{e+2} \leq 2$, so one and so forth until $m_{e+1} + \cdots + m_k \leq k - e$. Therefore it guarantees that they are no more monomials than positions between e and any position of degree higher than e. So we can repetitively apply Proposition 5.1.4 on f to obtain a function with direct sum vector containing $\sum_{i=e+1}^{k} m_i$ consecutive 1 from m_{e+1} and zeros for higher positions (the zeros are then deleted for a correct representation of the direct sum vector). A constructive way can consist in contracting each monomial to the first empty position of the vector, from m_{e+1} to m_k. It would lead to a function F with direct sum vector of length $e + \sum_{i=e+1}^{k} m_i$:

$$\mathbf{m}_F = [(m_1, \cdots, m_e, 1, \cdots, 1].$$

Together with the reasoning on the lower positions of the vector, this result shows that the AI of f can be linked through the repetitive use of Proposition 5.1.4 to the AI of a function F with direct sum vector of length $e + \sum_{i=e+1}^{k} m_i$:

$$\mathbf{m}_F = [m_1, \cdots, m_e, 1, \cdots, 1], \text{ such that } m_i > 0 \; \forall i \in [1, e].$$

As $\mathsf{AI}(F) = e + \sum_{i=e+1}^{k} m_i$, Proposition 5.1.4 gives that $\mathsf{AI}(f) > e - 1 + \sum_{i=e+1}^{k} m_i$, proving the second inequality, and finishing the proof.

□

Remark 5.1.6. *This theorem gives the exact algebraic immunity of all functions obtained by direct sums of monomials, it also characterizes all functions whose algebraic immunity is exactly the number of monomials in the ANF. Indeed, these functions are direct sums of monomials, otherwise similarly to the first part of the proof of the theorem, we can construct functions $1+x_j$ which annihilate more than one monomial, giving a final annihilator of degree strictly less than the number of monomials. Among the functions with algebraic immunity equal to the number of monomials in their ANF, by construction the triangular functions are the ones with the minimal number of variables for a fixed algebraic immunity.*

5.2 Recurrent Criteria

In [MJSC16], the motivation to study the recurrent criteria on Boolean functions is the guess-and-determine attack as presented in [DLR16b]. The efficiency of the attack can be bounded using the parameters relatively to the standard cryptographic criteria of the functions that can be obtained by fixing the values of some of the variables. The complexity of the attack depends on the worst parameters among all the Boolean functions that can be obtained from F by fixing ℓ of its N variables (each one to 0 or 1); it defines the notion that we call *recurrent criterion of order ℓ*. As example, the recurrent algebraic immunity of order ℓ of F is the minimal algebraic immunity taken over all functions obtained by fixing ℓ among N variables of F, therefore a minimum taken over 2^ℓ times N choose ℓ values, and denoted by $\mathsf{AI}[\ell](F)$.

According to the security requirements for the Filter Permutator (see Section 4.5.2) in this study we focus on the extension of a limited number of standard criteria to their recurrent versions, the one which are connected with a known attack. Therefore we examine the recurrent algebraic immunity, the recurrent nonlinearity and the recurrent nonlinearity of higher order. Except for the first one, we investigate the recurrent parameters only for a family of functions: the direct sums of monomials. Note that we do not examine the recurrent balancedness whereas balancedness is an important cryptographic criterion; in fact, this notion coincides with the resiliency: the function with worst balancedness when ℓ inputs are fixed is unbalanced if $\mathsf{res}(F) < \ell$ and balanced otherwise.

5.2.1 Definitions and General Bounds

5.2.1.1 Definitions

We give here a formal definition of the recurrent algebraic immunity, recurrent nonlinearity and recurrent nonlinearity of higher order, of a function. These are the three quantities to examine for the attacks considered in Section 4.5.2.

Definition 5.2.1 (Recurrent Algebraic Immunity)**.** *Let F be a Boolean function of N variables and ℓ an integer such that $0 \leq \ell < N$, we define the Recurrent Algebraic Immunity of F of order ℓ as:*

$$\mathsf{AI}[\ell](F) = \min_{\pi \in S_N} \left(\min_{v \in \mathbb{F}_2^\ell} \left[\mathsf{AI}\left(F(x_1, \cdots, x_n) \mid (x_{\pi(1)}, \cdots, x_{\pi(\ell)}) = v \right) \right] \right),$$

where S_N is the group of permutations of N elements.
Note that the minimum is finally taken over $2^\ell \binom{N}{\ell}$ functions defined over $N - \ell$ variables.

In the following part we give bounds on the parameters relative to this criterion and exhibit functions reaching these bounds. The recurrent algebraic immunity is important on a security perspective, it enables to stand on the security of both algebraic and fast algebraic attacks combined with guess-and-determine techniques.

We define in a very similar way the recurrent nonlinearity and recurrent nonlinearity of higher oder. To avoid misunderstandings, we denote the order of nonlinearity as an index (bottom, right) whereas the recurrent order is denoted between square brackets. As the standard nonlinearity is the nonlinearity of order one, the NL_1 notation is replaced by NL in the other sections.

Definition 5.2.2 (Recurrent Nonlinearity (of Higher Order))**.** *Let F be a Boolean function of N variables and ℓ an integer such that $0 \leq \ell < N$, we define the recurrent nonlinearity (of order d) of F of recurrent order ℓ as:*

$$\mathsf{NL}_d[\ell](F) = \min_{\pi \in S_N} \left(\min_{v \in \mathbb{F}_2^\ell} \left[\mathsf{NL}_\mathsf{d} \left(F(x_1, \cdots, x_n) \mid (x_{\pi(1)}, \cdots, x_{\pi(\ell)}) = v \right) \right] \right).$$

This quantity can be sufficient to stand on the security relatively to correlation-like attacks of various orders combined with guess-and-determine techniques.

Note that for all recurrent quantities the recurrent order 0 corresponds to the standard quantity, then the parameters of a function relatively to a recurrent criterion can be as high as in the standard case, and as low as for a trivial function, depending on the recurrent order.

5.2.1.2 Bounds on the Recurrent Algebraic Immunity

In this part we investigate the minimal and maximal values of the recurrent algebraic immunity for all functions F, in term of the algebraic immunity of F and the recurrent order ℓ. Then, finding functions reaching these extrema enable us to stand on the good candidates for cryptographic constructions affected by (fast) algebraic attacks combined with guess-and-determine attacks.

Lemma 5.2.3 (Extremum Values of the Recurrent Algebraic Immunity)**.** *Let F be a Boolean function in N variables, and ℓ an integer such that $0 \leq \ell < N$, the recurrent algebraic immunity of F follows:*

$$\mathsf{AI}(F) - \ell \leq \mathsf{AI}[\ell](F) \leq \mathsf{AI}(F).$$

Proof. First we prove the upper bound $\mathsf{AI}[\ell](F) \leq \mathsf{AI}(F)$. By definition of the algebraic immunity, we can take a function g in N variables such that:

$$gF = 0 \text{ or } g(F+1) = 0, \text{ where } \deg(g) = \mathsf{AI}(F) \neq 0.$$

As g is not null, for all ℓ less than N there exists a set I of ℓ indexes between 1 and N and an element $v \in \mathbb{F}_2^\ell$ such that the $(N-\ell)$-variable function $g(x_1, \cdots, x_n)|(x_{I_1}, \cdots, x_{I_\ell}) = v$ is non null. Without loss of generality we take I as the integer set from $N - \ell + 1$ to N.

If g annihilates F on $\mathbb{F}_2^N$ we get:

$$F(x_1, \cdots, x_{N-\ell}, v) g(x_1, \cdots, x_{N-\ell}, v) = 0, \text{ and } g(x_1, \cdots, x_{N-\ell}, v) \neq 0.$$

Otherwise g annihilates $F+1$, then we get the same equation with $1+(x_1,\cdots,x_{N-\ell},v)$. Combining these two equations gives:

$$\mathsf{AI}(F(x_1,\cdots,x_{N-\ell},v)) \leq \deg(g(x_1,\cdots,x_{N-\ell},v),$$

by definition of the degree,

$$\deg(g(x_1,\cdots,x_{N-\ell},v)) \leq \deg(g),$$

by definition of the recurrent algebraic immunity,

$$\mathsf{AI}[\ell](F) \leq \mathsf{AI}(F(x_1,\cdots,x_{N-\ell},v)),$$

finally giving the upper bound

$$\mathsf{AI}[\ell](F) \leq \mathsf{AI}(F).$$

Then, we prove the lower bound $\mathsf{AI}(F)-\ell \leq \mathsf{AI}[\ell](F)$, proceeding by contradiction. Let us suppose $\mathsf{AI}[\ell](F) < \mathsf{AI}(F)-\ell$, then there exist:

- an element $v \in \mathbb{F}_2^\ell$,
- an integer set I of ℓ elements between 1 and N,

such that the $(N-\ell)$-variable F' defined over elements

$$(x_1,\cdots,x_N) \mid (x_{I_1},\cdots,x_{I_\ell}) = v$$

has an algebraic immunity strictly smaller than $\mathsf{AI}(F)-\ell$.

Without loss of generality we set $I=[N-\ell+1,N]$, giving

$$F'(x_1,\cdots,x_{N-\ell}) = F(x_1,\cdots,x_{N-\ell},v_1,\cdots,v_\ell), \text{ with } \mathsf{AI}(F') < \mathsf{AI}(F)-\ell.$$

Then, we create an annihilator of F or $F+1$ of degree strictly smaller than $\mathsf{AI}(F)$. To do so, we first consider a simpler case: $\ell=1$ and $v=0$. In this case, if g is a non null annihilator of F' (or $F'+1$) we have that $(1+x_N)g$ is a non null annihilator of F (or $F+1$). Indeed, if $x_N=0$ then g annihilates F(or $F+1$) where F' is defined, otherwise $x_N=1$ then $1+x_N$ is null. By construction $(1+x_N)g$ cannot be null and is therefore an annihilator of F (or $F+1$) of degree $\deg(g)+1$.

We generalize this strategy for all ℓ such that $0 \leq \ell < N$ and all $v \in \mathbb{F}_2^\ell$; let g be an $(N-\ell)$-variable function of degree $\mathsf{AI}(F')$ annihilating F', then the following holds:

$$\left(\prod_{i=1}^{\ell}(1+x_{N-\ell+i}+v_i)\right) gF = 0.$$

If g is an annihilator of $F'+1$ instead, the above equation is still correct for $F+1$ instead of F. In both cases, it gives a non null annihilator of degree $\mathsf{AI}(F')+\ell$, strictly less than $\mathsf{AI}(F)$, which gives a contradiction.

Therefore we can conclude that $\mathsf{AI}(F)-\ell \leq \mathsf{AI}[\ell](F)$, finishing the proof.

□

Note that this result corresponds to the Proposition 1 in [DGM04], written differently.

These two bounds are tight, we exhibit cases where the lower bound is tight, focusing on the family of majority functions. Majority functions have optimal algebraic immunity, then it makes interesting the study of their behavior relatively to recurrent algebraic immunity; wondering if they are still optimal in this context or not. We show in the following proposition that they reach the lower bound of the precedent lemma.

Proposition 5.2.4. *Let N be a positive odd integer and ℓ an integer such that $0 \leq \ell < N$, then the following holds for the majority functions:*

$$\mathsf{AI}[\ell](Maj_N) = \max\left(0, \left\lceil \frac{N}{2} \right\rceil - \ell\right).$$

Proof. From the definition of the majority function (Definition 2.4.18) we know that:

$$Maj_N(x) = 1 \iff Hw(x) \geq \left\lceil \frac{N}{2} \right\rceil.$$

We consider the function obtained from Maj_N by fixing the last ℓ variables to 1, noted F, a $(N-\ell)$-variable function:

$$\forall y \in \mathbb{F}_2^{N-\ell},\ F(y) = Maj_N(y_1, \cdots, y_{N-\ell}, 1, \cdots, 1),$$

which is equivalent to:

$$F(y) = 1 \iff Hw(y) \geq \left\lceil \frac{N}{2} \right\rceil - \ell.$$

Then, let us consider the elementary symmetric function in $N-\ell$ variables of degree $\lceil N/2 \rceil - \ell$ that we denote $\sigma_{\lceil N/2 \rceil - \ell}$.

Now we focus on the Hamming weight of the elements $y \in \mathbb{F}_2^{N-\ell}$: if $\mathsf{wH}(y) < \lceil N/2 \rceil - \ell$ then $F(y) = \sigma_{\lceil N/2 \rceil - \ell}(y) = 0$, otherwise $\mathsf{wH}(y) \geq \lceil N/2 \rceil - \ell$ then $(F+1)(y) = 0$.

Combining these two cases:

$$\forall y \in \mathbb{F}_2^{N-\ell}\ (F+1)\sigma_{\lceil N/2 \rceil - \ell}(y) = 0.$$

Therefore, $\sigma_{\lceil N/2 \rceil - \ell}$ is a non null annihilator of $F+1$ if $\lceil N/2 \rceil - \ell > 0$, giving in this case

$$\mathsf{AI}[\ell](F) \leq \lceil N/2 \rceil - \ell.$$

Note that if this function is null, we are in the case where $F = 1$ giving $\mathsf{AI}(F) = 0$. The lower bound of Lemma 5.2.3 enables to conclude:

$$\mathsf{AI}[\ell](Maj_N) = \max\left(0, \left\lceil \frac{N}{2} \right\rceil - \ell\right).$$

□

We can also exhibit functions whose recurrent algebraic immunity follows the upper bound of Lemma 5.2.3. For example, the direct sum of nb triangular functions of degree k has the same recurrent algebraic immunity for all orders ℓ such that $0 \leq \ell < nb$ (it is a direct corollary of Theorem 5.1.5). Moreover, some functions can reach the upper bound for all recurrent orders as illustrated by the next remark .

Remark 5.2.5. *The elementary symmetric function of degree* 1*, noted* σ_1*, reaches the upper bound of Lemma 5.2.3 for all recurrent order. Indeed, for all* N *this non constant linear functions is such that* $\mathsf{AI}(\sigma_1) = 1$*, and as* $\mathsf{res}(\sigma_1) = N - 1$*, all functions obtained from* σ_1 *by fixing at most* $N - 1$ *of its variables is still balanced. The algebraic immunity of a balanced function being at least* 1*, it gives:*

$$\forall \ell \text{ such that } 0 \leq \ell < N, \quad \mathsf{AI}[\ell](\sigma_1) = \mathsf{AI}(\sigma_1) = 1.$$

Note that the examples for the lower and upper bounds of Lemma 5.2.3 show a variety of behaviors for the recurrent algebraic immunity. From the algebraic immunity upper bound we can derive the upper bound of $\lceil (N-\ell)/2 \rceil$ for its recurrent analog. Some of the functions we exhibit reach this upper bound for particular values of ℓ, a natural question consists in asking if there exists functions of N variables reaching this bound for all ℓ such that $0 \leq \ell < N$. It would lead to the concept of optimal recurrent algebraic immunity, functions with all sub-functions obtained by fixing variables have optimal algebraic immunity. For $N = 2$, it is the case of σ_1 as shown in the precedent remark, however we do not know constructions for all N satisfying this property, and defer it to future investigations.

5.2.2 Recurrent Criteria for Direct Sums of Monomials

In this part we examine the recurrent criteria specifically for functions obtained by direct sums of monomials. As the FLIP functions are a sub-part of this family, upper bounds on the parameters of these functions enable to study the concrete security of the instantiations of the Filter Permutator.

We begin by defining two criteria of particular interest for the direct sums of monomials:

Definition 5.2.6 (Recurrent Criteria for Direct Sums of Monomials)**.** *For a Boolean function* F *of* N *variables obtained by direct sums of monomials and with associated direct sum vector* $\mathbf{m}_F = [m_1, m_2, \cdots, m_k]$ *we define the criteria:*

- $\mathbf{m}_F^*$ *as the number of nonzero values of* $\mathbf{m}_F$*,*
- $\delta_d = \frac{1}{2} - \frac{\mathsf{NL}_d(F)}{2^N}$ *as the bias to* $1/2$ *of the best degree* d *approximation of* F*.*

We accordingly define $\mathbf{m}_F^*[\ell]$ *and* $\delta_d[\ell]$*, as the minimal value (relatively to these criteria) taken over the* N *choose* ℓ *times* 2^ℓ *functions obtained from* F *by fixing* ℓ *variables.*

These criteria are adapted to quantify the impact of guessing some bits on the cryptographic properties of a Boolean function obtained by direct sums. $\mathbf{m}_F$, $\mathbf{m}_F^*$ are easily computable from the description of F, δ_d and $\delta_d[\ell]$ can be computed recursively using Lemma 5.1.1. We first give an upper bound on $\mathbf{m}_F^*[\ell]$, this quantity being easy to derive from the direct sum vector notation, and strongly connected to $\mathsf{AI}[\ell](F)$.

Lemma 5.2.7. *Let* F *be a Boolean function obtained by direct sums of* M *monomials with associated direct sum vector* $\mathbf{m}_F = [m_1, m_2, \cdots, m_k]$*, and* ℓ *an integer such that* $0 \leq \ell < M$ *then the following holds for* $\mathbf{m}_F^*[\ell]$*:*

$$\mathbf{m}_F^*[\ell] \geq \mathbf{m}_F^* - \left\lfloor \frac{\ell}{\min_{1 \leq i \leq k} m_i} \right\rfloor.$$

Proof. First, note that for each variable fixed, the function obtained is still a direct sum of monomials, and at most one of the m_i is reduced. Indeed, a variable fixed to 0 cancels one monomial, reducing by one the coefficient m_i where the variable appears. A variable fixed to 1 changes a monomial of degree d into a monomial of degree $d-1$; if $d > 1$ then it adds one to m_{i-1} and reduce m_i by one, otherwise $d = 1$ and it only reduces m_1 by one.

Then, as each fixed variable reduces at most one m_i and by 1, $\mathbf{m}_F^*$ cannot be reduced by more than the maximal number of m_is than can be reduced to 0 by fixing ℓ variables. It ensures:

$$\mathbf{m}_F^*[\ell] \geq \mathbf{m}_F^* - \max_{\pi \in S_k} \left[j \mid \left(\sum_{i=1}^{j} m_{\pi(i)} \right) \leq \ell \right],$$

where S_k denotes the group of permutations of $k = \deg(F)$ elements.

This maximum corresponds to the maximal number of m_is that can be changed in 0 with ℓ variables fixed, as fixing these ℓ variables to 0 does not increase any other m_i this maximum is reached, transforming this bound in an equality. This maximum can be computed by summing the successive minima of $\mathbf{m}_F$ until the sum exceeds ℓ, the number of successive minima subtracted to $\mathbf{m}_F^*$ giving the exact value of $\mathbf{m}_F^*[\ell]$. We follow our analysis to give an upper bound on $\mathbf{m}_F^*[\ell]$ easier to compute and express.

We can bound this maximum:

$$\max_{\pi \in S_k} \left[j \mid \left(\sum_{i=1}^{j} m_{\pi(i)} \right) \leq \ell \right] \leq \left\lfloor \frac{\ell}{\min_{1 \leq i \leq k} m_i} \right\rfloor.$$

Using this bound (corresponding to the worst case where all m_is are equals) we get the final bound:

$$\mathbf{m}_F^*[\ell] \geq \mathbf{m}_F^* - \left\lfloor \frac{\ell}{\min_{1 \leq i \leq k} m_i} \right\rfloor.$$

□

Note that for all Boolean functions obtained by direct sums of monomials F, $\mathbf{m}_F^* \leq \mathsf{AI}(F)$ as a direct consequence of Theorem 5.1.5. Then as fixing variables of F does not change its quality of direct sums of monomials, $\mathbf{m}_F^*[\ell]$ gives a lower bound on $\mathsf{AI}[\ell](F)$ for this family of functions. The advantage of $\mathbf{m}_F^*[\ell]$ is to be easily computed or bounded; instead, the exact $\mathsf{AI}[\ell]$ can be determined using Theorem 5.1.5. It requires to consider the minimum over all functions obtained by fixing ℓ variables; each one being obtained by direct sums of monomials, the theorem gives its exact AI. Nevertheless this strategy supposes to compute a minimum over at most 2^ℓ times N choose ℓ minima, which can be quite costly.

We then give a lower bound on $\delta_1[\ell]$, related to the recurrent nonlinearity, this quantity appearing in the complexity bounds of the attacks we described related to correlation-like attacks combined with guess-and-determine attacks. For readability in the rest of this section we refer as δ for δ_1 and $\delta[\ell]$ for $\delta_1[\ell]$.

Lemma 5.2.8. *Let F be a Boolean function obtained by direct sums of M monomials with associated direct sum vector $\mathbf{m}_F = [m_1, m_2, \cdots, m_k]$, and ℓ an integer such that $0 \leq \ell < M$ then the following holds for $\delta[\ell]$:*

$$\delta[\ell](F) \;\leq\; \delta(F) \cdot 2^\ell.$$

Proof. First, we study the parameter δ on similar functions, noting that for all functions $0 < \delta(F) \leq 1/2$. We focus on $\delta(F)$ and $\delta(G)$ where F is the direct sum of G (of $N-n$ variables) and a monomial m of degree $n > 1$.

Using the Walsh transform:

$$\forall (u,v) \in \mathbb{F}_2^{N-n} \times \mathbb{F}_2^n, \quad \hat{F}_\chi(u,v) = \hat{G}_\chi(u)\hat{m}_\chi(v).$$

As $\max_{v \in \mathbb{F}_2^n}(\hat{m}_\chi(v)) = 2^n - 2$, we deduce:

$$\delta(F)2^N = \delta(G)2^{N-n}(2^n - 2),$$

and accordingly:

$$\delta(G) = \delta(F) \cdot \left(\frac{2^{n-1}}{2^{n-1}-1}\right).$$

The last case to consider is when $n = 1$, then $\mathsf{NL}(F) = 2\mathsf{NL}(G)$ by Lemma 5.1.1 and therefore $\delta(G) = \delta(F)$.

Then, we generalize this study for all ℓ, using the direct sum vector notation. As explained in the proof of Lemma 5.2.7, all functions obtained by fixing ℓ variables of the direct sum of monomials F is a direct sum of monomials. Fixing a variable to 0 cancels a monomial, decreasing the nonlinearity and increasing δ if this monomial is not of degree 1 (in this particular case the nonlinearity does not change). Fixing a variable to 1 increases the nonlinearity if the degree of the monomial is greater than 2, otherwise has the same impact on the nonlinearity as fixing it to 0.

As $\left(\frac{2^{n-1}}{2^{n-1}-1}\right)$ decreases when n increases we obtain the bound:

$$\delta[\ell](F) \leq \delta(F) \max_{\substack{(\ell_1,\cdots,\ell_k) \\ \forall i\ \ell_i \leq m_i \text{ and } \ell = \sum_{i=1}^k \ell_i}} \prod_{i=2}^{k} \left(\frac{2^{i-1}}{2^{i-1}-1}\right)^{\ell_i}.$$

The maximum is taken over all the possible choices of fixing ℓ of the N variables to 0. Indeed, the maximum over all possible choices of fixing the value of ℓ of the N variables can be reached by fixing zeros only as explained above. As the maximum corresponds to at least one of the potential fixes and as the value corresponding to the 2^ℓ times N choose ℓ functions, the recurrent bias equals the product of $\delta(F)$ by this maximum.

So, we optimize the choices for the ℓ_i to determine this maximum. Let j denote the integer such that:

$$\sum_{i=2}^{j} m_i \leq \ell < \sum_{i=2}^{j+1} m_i.$$

If such a j exists (it means that $\ell + m_1 \leq M$) then:

$$\max_{\substack{(\ell_1,\cdots,\ell_k) \\ \forall i\ \ell_i \leq m_i \text{ and } \ell = \sum_{i=1}^k \ell_i}} \prod_{i=1}^{k} \left(\frac{2^{i-1}}{2^{i-1}-1}\right)^{\ell_i} = \left(\prod_{i=2}^{j} \left(\frac{2^{i-1}}{2^{i-1}-1}\right)^{m_i}\right) \left(\frac{2^j}{2^j-1}\right)^{\ell - \sum_{i=2}^{j} m_i},$$

otherwise (the case $\ell + m_1 > M$), the maximum becomes:

$$\prod_{i=2}^{k} \left(\frac{2^{i-1}}{2^{i-1}-1}\right)^{m_i}.$$

Summing the two cases, we get the exact value of $\delta[\ell](F)$ for all ℓ such that $0 \leq \ell < M$ for all direct sums of monomials. We give a not tight bound, as it is easier to compute and can be derived only from $\mathsf{NL}(F)$, without using the direct sum vector representation.

As we always have $\ell_2 \leq \ell$, and that the higher value of $\left(\frac{2^{i-1}}{2^{i-1}-1}\right)$ is obtained for $i = 2$, we consider the worst case: $\ell_2 = \ell$, giving:

$$\prod_{i=2}^{k} \left(\frac{2^{i-1}}{2^{i-1}-1}\right)^{\ell_i} \leq 2^{\ell},$$

and therefore the conservative bound

$$\delta[\ell](F) \leq \delta(F)2^{\ell}.$$

□

Finally, we give a lower bound on $\delta_d[\ell]$, related to the recurrent nonlinearity of higher order (greater than 1); this quantity enables to stand on the attacks related to high order correlation attacks combined with guess-and-determine attacks. Even for direct sums constructions the nonlinearity of higher order is not a criterion well studied (up to our knowledge), way less examined than the other cryptographic criteria. Therefore, the best approximation of degree d of a function obtained by direct sums of monomials F is not exhibited. In order to get bounds for security we assume in the next lemma that the best approximation of F of degree d is reached by its part of degree less than or equal to d. Note that it is not the case for all functions, but we did not find any counter example for functions obtained by direct sums of monomials. In this particular case, it seems that no better approximation of a direct sum could be reached than the one obtained by the direct sum of the best approximations of the two components.

Lemma 5.2.9. *Let F be a Boolean function obtained by direct sums of M monomials with associated direct sum vector $\mathbf{m}_f = [m_1, m_2, \cdots, m_k]$, ℓ an integer such that $0 \leq \ell < M$, and d an integer such that $1 < d \leq k$. If we assume that for all Boolean functions G obtained by direct sums of monomials with associated direct sum vector $\mathbf{m}_G = [m_1, m_2, \cdots, m_k]$, the function G_d with associated direct sum vector $\mathbf{m}_{G_d} = [m_1, m_2, \cdots, m_d]$ corresponds to the best approximation of F of degree d, then the following holds for $\delta[\ell]$:*

$$\delta_d[\ell] \ \leq \ \delta \cdot \left(\frac{2^d}{2^d-1}\right)^{\ell}.$$

Proof. This proof is very similar to the proof of the precedent lemma, the main difference is the need of the assumption to determine the high order nonlinearity of direct sums of monomials. For the first order nonlinearity, Lemma 5.1.1 item 4 enables to determine this quantity, the assumption enables to use the same formula for higher order nonlinearity, in the particular case of direct sums of monomials.

We begin by examining the parameter δ_d on similar functions, noting $F = G + x_I$ where F is an N-variable function, G an $(N-n)$-variable function and x_I a monomial function of degree n. F and G are obtained by direct sums of monomials. We consider two cases: either $n \leq d$, or $n > d$.

- Case $n \leq d$. We make a partition of the truth table of F, in 2^n parts, depending on the value of the n variables of x_I. For all choices not taking the n variables of x_I all equal to 1, we obtain a truth table of size 2^{N-n} where F is equal to G, the distance between $(G_d + x_I)$ and F on this part is therefore $\mathsf{NL_d}(G)$ using the assumption. On the part corresponding to the n variables of x_I being all equal to 1, the function F takes the values of $G+1$, the distance between $(G_d + x_I)$ and F on this part is therefore $\mathsf{NL_d}(G)$ using the assumption. Summing the 2^n parts, it gives:

$$\mathsf{NL_d}(F) = 2^n \mathsf{NL_d}(G).$$

- Case $n > d$. Using the assumption, the best approximation of F is $G_d + 0$. Using the same partition as before, on all parts where the n variables of x_I are not all equal to 1, the distance between G_d and F is equal to $\mathsf{NL_d}(G)$. On the last part, F takes the values of $G+1$, therefore the distance to G_d is equal to $2^{N-n} - \mathsf{NL_d}(G)$. Summing the 2^n parts, it gives:

$$\mathsf{NL_d}(F) = (2^n - 2)\mathsf{NL_d}(G) + 2^{N-n}.$$

In terms of δ_d it gives:

$$\delta_d(G) = \begin{cases} \delta_d(F) & \text{if } n \leq d, \\ \delta_d(F) \cdot \left(\frac{2^{n-1}}{2^{n-1}-1}\right) & \text{otherwise.} \end{cases}$$

The behavior of δ_d for direct sums of monomials is then the same as δ, where the same equation holds for $d = 1$. Then, we can use the proof of the precedent lemma (Lemma 5.2.8), giving the exact value of $\delta_d[\ell](F)$ for all ℓ such that $0 \leq \ell < M$, or the upper bound:

$$\delta_d[\ell] \ \leq \ \delta \cdot \left(\frac{2^d}{2^d - 1}\right)^{\ell}.$$

□

Note that for these criteria, if the recurrent order ℓ reaches or exceeds the number of monomials M of the function F, then $\mathbf{m}^*_F[\ell] = 0$, and $\delta_d[\ell](F) = 1/2$. So, these criteria are useful tools to study the security of direct sums of monomials, when a limited number of variables are fixed. The 3 associated lemmata enable to derive lower bounds easy to compute in order to discard functions for security reasons, and their proof enables to determine the exact value of $\mathsf{AI}[\ell]$ and $\mathsf{NL}[\ell]$.

5.3 Restricted Algebraic Immunity

In a cryptographic framework, Boolean functions are classically studied with an input ranging over the whole vector space $\mathbb{F}_2^n$ of binary vectors of some length n. This is the case when Boolean functions are used as the (main) nonlinear components of a stream cipher, in the so-called combiner and filter models of pseudo-random generators. However, it can happen that the function be in fact restricted to a subset (say E) of $\mathbb{F}_2^n$, an example of such situation is given by the cipher FLIP (see [MJSC16]). Consequently, in [CMR17] we study the main cryptographic criteria of Boolean functions when their input is not $\mathbb{F}_2^n$ but restricted to a subset of $\mathbb{F}_2^n$. This formulation of cryptographic Boolean criteria is quite general: as an

example the standard criteria correspond to the whole set $\mathbb{F}_2^n$, and the recurrent criteria correspond to the subsets of $\mathbb{F}_2^n$ of cardinal $2^{n-\ell}$ obtained by fixing ℓ of the n variables.

This study focuses on the properties of algebraic immunity, nonlinearity and balancedness, restricted to any subset and then more particularly to the subsets of all words of a fixed Hamming weight - that we note $E_{n,k}$ where $E_{n,k} = \{x \in \mathbb{F}_2^n \mid \mathsf{wH}(x) = k\}$ - due to their relation to the Filter Permutator construction, as explained in Section 4.5.3. A study has been made by Yuval Filmus et al. on the restrictions of Boolean functions to sets of inputs of fixed Hamming weight (that he calls "slices") [Fil16b; Fil16a; FKMW16; FM16]. This study is asymptotic and does not really fit with our cryptographic framework; the results from these papers have no overlap with the ones of [CMR17]. For readability, we decide to present the results of [CMR17] relatively to the three main restricted criteria in three different sections. In this section we present the results on restricted algebraic immunity, in the following section (Section 5.4) the study of the restricted nonlinearity and finally we examine the restricted balancedness in Section 5.5 to finish this chapter. We chose this particular order to follow the pattern adopted various times in this manuscript, considering first algebraic immunity, then nonlinearity, and finally balancedness.

We begin our study on the restricted algebraic immunity by defining this concept. It is the natural generalization of the algebraic immunity introduced in [CM03] to any subset of $\mathbb{F}_2^n$.

Definition 5.3.1 (Restricted Algebraic Immunity)**.** *We call Algebraic Immunity of a function f over a set E the number:*

$$\mathsf{AI}_E = \min\{\deg(g) : g \text{ annihilator of } f \text{ or of } f+1 \text{ over } E \text{ and } g \text{ not identically null over } E\}.$$

Note that the work [CFGR12] realizes a theoretical study of the algebraic phase of the so-called algebraic side channel attacks on block ciphers. The authors modify the notion of algebraic immunity to include the information (on Hamming weight or Hamming distance) obtained by exploiting the leakage and are able to obtain enough equations of degree one to solve the algebraic system with Groebner methods. In the following, our modification of the definition of algebraic immunity is also related to Hamming weight when we focus on the sets $E_{n,k}$ but is of a different nature, being related to the fact that the input is restricted. Another major difference is that we focus on functions with one bit of output and not on S-boxes.

In the following, we generalize the algebraic immunity upper bound for all sets E and then we give precise results in the constant Hamming weight case. Finally, we examine the algebraic immunity restricted to $E_{n,k}$ for direct sums of functions and prove a counter-intuitive result showing how the general algebraic immunity can decrease in this case.

5.3.1 Algebraic Immunity Upper Bound for all Restricted Sets

In the case of $E = \mathbb{F}_2^n$, Courtois and Meier have shown that, for every non-negative integers d and e such that $d + e \geq n$, there exists a nonzero Boolean function g of algebraic degree at most e and a Boolean function h of algebraic degree at most d such that $h = gf$. For $e = d = \lceil n/2 \rceil$, this proved that the algebraic immunity of f is at most $\lceil n/2 \rceil$. We revisit here these results for functions defined over a subset of $\mathbb{F}_2^n$.

Proposition 5.3.2. *Let E be any non-empty subset of $\mathbb{F}_2^n$ and f any Boolean function defined over E. Let d and e be two non-negative integers and $M_{d,E}$ be the $(\sum_{i=0}^{d} \binom{n}{i}) \times |E|$*

binary matrix whose term at row indexed by $u \in \mathbb{F}_2^n$, $\mathsf{w_H}(u) \leq d$, *and at column indexed by* $x \in E$ *equals* $x^u := \prod_{i=1}^n x_i^{u_i}$. *Assume that the ranks of matrices* $M_{d,E}$ *and* $M_{e,E}$ *are such that:*

$$\mathsf{rank}(M_{d,E}) + \mathsf{rank}(M_{e,E}) > |E|,$$

then there exists a Boolean function g *of algebraic degree at most* e *over* $\mathbb{F}_2^n$, *whose restriction to* E *is not identically null, and a Boolean function* h *of algebraic degree at most* d *on* $\mathbb{F}_2^n$, *such that functions* gf *and* h *coincide on* E.

Proof. Let $\mathcal{F}_d$ (respectively $\mathcal{F}_e$) be a maximum-size free family of restrictions to E of monomials x^u of algebraic degree $\deg(x^u) = \mathsf{w_H}(u) \leq d$ (respectively at most e). By definition of the rank of a matrix, the size of $\mathcal{F}_d$ equals $\mathsf{rank}(M_{d,E})$ and that of $\mathcal{F}_e$ equals $\mathsf{rank}(M_{e,E})$.

Let us consider now the family, that we shall denote by $\mathcal{F}_e f$, whose elements (with possible repetitions) are the products of the elements of $\mathcal{F}_e$ by the function f. By hypothesis:

$$|\mathcal{F}_d| + |\mathcal{F}_e f| > |E|,$$

where $|E|$ is the dimension of the $\mathbb{F}_2$-vector-space of all Boolean functions over E, (indeed, the number of Boolean functions over E equals $2^{|E|}$).

Then, there exists a linear combination of the elements of $\mathcal{F}_d$ and of those of $\mathcal{F}_e f$, which equals the zero function and whose coefficients are not all null. Gathering the part of this linear combination dealing with the elements of $\mathcal{F}_d$ and those dealing with $\mathcal{F}_e f$, we obtain respectively functions h and g such that h and gf coincide over E, and the restrictions of g and h to E are not both null (since both families $\mathcal{F}_e$ and $\mathcal{F}_d$ are free), that is, the restriction of g to E is nonzero. □

Remark 5.3.3. *The matrices* $M_{d,E}$ *and* $M_{e,E}$ *are generator matrices of punctured binary Reed-Muller codes of order* d *and* e *respectively; more connexions between restricted criteria and punctured binary Reed-Muller code are explained in Section 5.4.*

Taking $e = 0$ in Proposition 5.3.2, we obtain $\mathsf{rank}(M_{e,E}) = 1$, since the constant function 1 does not vanish over E, and:

Corollary 5.3.4. *Let* E *be any non-empty subset of* $\mathbb{F}_2^n$ *and* f *any Boolean function defined over* E. *Let* n *and* d *be such that* $\mathsf{rank}(M_{d,E}) = |E|$, *then there exists a Boolean function over* $\mathbb{F}_2^n$ *of algebraic degree at most* d *which coincides with* f *on* E.

In other words, the algebraic degree of any Boolean function over E is bounded above by the least value of d such that $\mathsf{rank}(M_{d,E}) = |E|$. Indeed, in Proposition 5.3.2, we have $g = 1$ and $gf = h$ on E where h has algebraic degree at most d.

Taking $d = 0$, we have $\mathsf{rank}(M_{d,E}) = 1$ and, calling *annihilator of* f *on* E any Boolean function g over E whose product with f vanishes:

Corollary 5.3.5. *Let* E *be any non-empty subset of* $\mathbb{F}_2^n$ *and* f *any non-constant Boolean function defined over* E. *Let* n *and* e *be such that* $\mathsf{rank}(M_{e,E}) = |E|$, *then there exists a nonzero annihilator of* f *of algebraic degree at most* e *over* E.

Indeed, in Proposition 5.3.2, we have h constant and since $gf = 1$ on E is impossible (as f is non constant), we have then $h = 0$. Note that this shows that the algebraic immunity

of a function, can tumble down when the input is restricted to a set E. Notice also that Corollary 5.3.5 can be viewed as a consequence of Corollary 5.3.4, since we can take $f+1$ for annihilator.

This being observed, we have in fact a stronger result when taking $e = d$. We obtain:

Corollary 5.3.6. *Let E be any non-empty subset of $\mathbb{F}_2^n$ and f any Boolean function defined over E. Let n and e be such that $\mathsf{rank}(M_{e,E}) > \frac{|E|}{2}$, then there exists g of algebraic degree at most e, whose product with f or $f+1$ is null on E, and whose restriction to E is nonzero.*

Indeed, using a classical idea of Courtois and Meier [CM03], either the functions g and h of Proposition 5.3.2 coincide on E, and we have then $gf + h = g(f+1) = 0$ on E, where g has algebraic degree at most e and nonzero restriction to E, or they do not and we have, after multiplication of equality $h = gf$ by f, that $(g+h)f = 0$, where $g+h$ has algebraic degree at most e and nonzero restriction to E.

The situation is then similar to that described by Courtois and Meier and this explains our definition of restricted algebraic immunity and leads to the following property:

Corollary 5.3.7. *The algebraic immunity of any Boolean function F defined over E is bounded above:*

$$\mathsf{AI}_E(F) \leq min\left(e;\ \mathsf{rank}(M_{e,E}) > \frac{|E|}{2}\right).$$

5.3.2 Algebraic Immunity Upper Bound for Fixed Hamming Weight Input

In this section, we focus on the particular case when the input is restricted to the words with fixed Hamming weight: $E_{n,k}$ for some $k \in [1, n-1]$. More formally for $k \leq n$ we denote by $E_{n,k}$ the set of such entries: $E_{n,k} = \{x \in \mathbb{F}_2^n; \mathsf{w_H}(x) = k\}$. To be able to evaluate efficiently in such situation the algebraic immunity by using Proposition 5.3.2 and its corollaries, there remains to calculate the rank of the matrix $M_{d,E_{n,k}}$ for each d and k:

Theorem 5.3.8. *Let n, k, and d be non negative integers, let $M_{d,E_{n,k}}$ be the $(\sum_{i=0}^{d}\binom{n}{i})\times|E_{n,k}|$ binary matrix whose term at row indexed by $u \in \mathbb{F}_2^n$, $\mathsf{w_H}(u) \leq d$, and at column indexed by $x \in E_{n,k}$ equals $x^u := \prod_{i=1}^{n} x_i^{u_i}$. Then the following holds:*

$$\mathsf{rank}(M_{d,E_{n,k}}) = \binom{n}{\min(d, k, n-k)}$$

Proof. For a readability perspective we first denote $M_{d,E_{n,k}}$ as $M_{n,k,d}$ in the following proof. The principle of this proof is to find a recurring relation on the rank of $M_{n,k,d}$. To this aim, we use a construction which looks like the well-known $u \parallel u+v$ construction of Reed-Muller codes where every Boolean function f of algebraic degree at most d can be written in the form :

$$f(x_1, \cdots, x_n) = g(x_1, \cdots, x_{n-1}) + x_n h(x_1, \cdots, x_{n-1}),$$

where g has algebraic degree at most d and h has algebraic degree at most $d-1$. In the sequel of the proof, we shall use the additional notation:

$$N_k = \binom{n}{k}, \quad \text{and} \quad D = \sum_{0 \leq i \leq d} \binom{n}{i}.$$

Let $\psi_{n,k,d}$ be the following linear application, mapping every Boolean function in n variables (defined by its ANF) of algebraic degree at most d to the restriction of its truth table to the elements in $E_{n,k}$:

$$\begin{array}{rccc} \psi_{n,k,d} & : & \mathbb{F}_2^D & \longrightarrow & \mathbb{F}_2^{N_k} \\ & & (a_u)_{u\in\mathbb{F}_2^n,\,\mathsf{w_H}(u)\le d} & \longmapsto & \left(\sum_{u\preceq x} a_u\right)_{x\in\mathbb{F}_2^n,\,\mathsf{w_H}(x)=k} \end{array}$$

where $u \preceq x$ means $u_i \le x_i$ for every i such that $1 \le i \le n$. This application $\psi_{n,k,d}$ is linear and moreover, the rank of $M_{n,k,d}$ is exactly the rank of this linear application $\psi_{n,k,d}$.

Denoting by m_u the monomial x^u, the rank of $\psi_{n,k,d}$ is the rank of the family of the following vectors:

$$(\psi_{n,k,d}(m_u))_{u\in\mathbb{F}_2^n,\,\mathsf{w_H}(u)\le d}.$$

We split the family of vectors $u \in \mathbb{F}_2^n, \mathsf{w_H}(u) \le d$ in two families, depending on the value of the last coordinate:

$$F_1 = \{u \in \mathbb{F}_2^n, \mathsf{w_H}(u) \le d; u_n = 0\}, \quad \text{and} \quad F_2 = \{u \in \mathbb{F}_2^n, \mathsf{w_H}(u) \le d; u_n = 1\}.$$

We also split the vectors x of $\mathbb{F}_2^n$ of Hamming weight k in two sets, depending on the value of the last coordinate:

$$E_1 = \{x \in \mathbb{F}_2^n, \mathsf{w_H}(x) = k; x_n = 0\} \quad \text{and} \quad E_2 = \{x \in \mathbb{F}_2^n, \mathsf{w_H}(x) = k; x_n = 1\}.$$

Notice that for every $u \in F_2$ and every $x \in E_1$, we have $m_u(x) = 0$.

The $\mathbb{F}_2^D \times \mathbb{F}_2^{N_k}$ matrix $M_{n,k,d}$ representing the linear application $\psi_{n,k,d}$ has then the form given in Figure 5.1, we use the represented decomposition to illustrate a recurrent relation on the rank of $M_{n,k,d}$.

The matrix $A_{n,k,d}$ takes its entries on the set of monomials in which x_n does not occur and of degrees at most d. The output of the linear application defined by $A_{n,k,d}$ is the truth table of Boolean functions on those inputs of Hamming weight k where the value of x_n is set to 0. Then, as x_n does not occur in the entries and is fixed to 0 in the output, $A_{n,k,d}$ defines exactly the linear application which gives the truth table on words of Hamming weight k of all Boolean functions with $n-1$ variables (x_n being fixed) and of degree at most d.

The matrix $B_{n,k,d}$ defines the linear application which gives the truth table on words of weight $k-1$ (because x_n is fixed to 1 and not 0 anymore) of all Boolean functions with $n-1$ variables (x_n being fixed) and of degree at most $d-1$. Hence, $A_{n,k,d}$ defines the linear application $\psi_{n-1,k,d}$ and $B_{n,k,d}$ defines $\psi_{n-1,k-1,d-1}$.

Moreover, let us prove that the rank of this matrix is equal to the rank of $A_{n,k,d}$ plus the rank of $B_{n,k,d}$ (*i.e.* $M_{n-1,k-1,d}$ does not play any role in the rank of the whole matrix). Indeed, if we have a vector of length $\binom{n}{k}$ which is a linear combination on the lines such that the last $\binom{n-1}{k}$ coordinates of the resulting vector are null, (*i.e.* we are in the kernel of $A_{n,k,d}$) then this vector is linearly dependent from the vectors defined by $B_{n,k,d}$. In terms of Boolean functions, we prove that if f is a Boolean function in the linear span of F_1 such that $\forall x \in E_1, f(x) = 0$ (*i.e.* in the kernel of $A_{n,k,d}$) then f is in the linear span of F_2; indeed:

$$f(x_1, \cdots, x_n) = x_n g(x_1, \cdots, x_{n-1}) + h(x_1, \cdots, x_{n-1}).$$

The Boolean function f is of degree less than d, implying that h is of degree less than d and g is of degree less than $d-1$. But for all $x \in E_1$, we have $f(x) = 0$, then that means

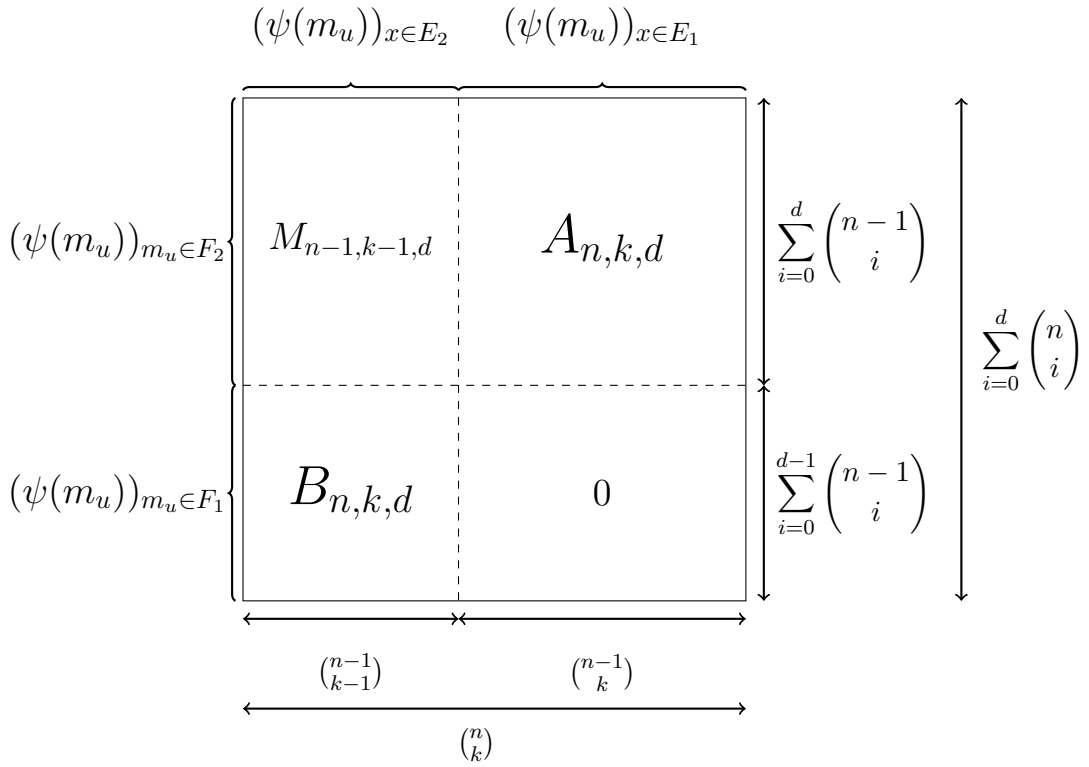

Figure 5.1: Decomposition of the matrix $M_{n,k,d}$.

that $h(x_1, \cdots, x_{n-1}) = 0$, then f is in the linear span of F_2. Then we deduce the following recurring relation:

$$\dim(\Im(\psi_{n,k,d})) = \dim(\Im(\psi_{n-1,k-1,d-1})) + \dim(\Im(\psi_{n-1,k,d})).$$

We need to initialize this relation. This can be done by considering the cases $d = 0$, $d = k$ and $d = n-k$. In the case $d = 0$ the matrix is only a row vector corresponding to the constant function 1, giving a dimension on $1 = \binom{n}{d}$. Moreover, if $d \geq k$ then $\dim(\Im(\psi_{n,k,d})) = \binom{n}{k} = \dim(\Im(\psi_{n,k,k}))$. In fact, the monomials of degree exactly k correspond to the canonical basis of the Boolean functions defined over $E_{n,k}$ (representing within their truth table). For $d \geq n-k$, we can choose the Boolean functions defined by $f(x) = (1+x_{i_1})(1+x_{i_2})\cdots(1+x_{i_{n-k}})$ which are of degree less than or equal to d and form also the canonical basis of the Boolean functions defined over $E_{n,k}$, giving here $\dim(\Im(\psi_{n,k,d})) = \binom{n}{n-k} = \dim(\Im(\psi_{n,k,n-k}))$. Then, as we found a recurring relation between $\dim(\Im(\psi_{n,k,d}))$, $\dim(\Im(\psi_{n-1,k-1,d-1}))$ and $\dim(\Im(\psi_{n-1,k,d}))$, the recursion always ends on one of these cases $d = 0$, $d \geq k$ or $d \geq n-k$, justifying the choice of initialization step.

Finally, using Pascal's identity we deduce by induction that:

$$\dim(\Im(\psi_{n,k,d})) = \binom{n}{\min(d, k, n-k)}.$$

□

From Corollary 5.3.7 and Theorem 5.3.8, we deduce:

Corollary 5.3.9. *Let k be any positive integer such that $k \leq n/2$. The algebraic immunity of the restriction of F to $E_{n,k}$ respects the following bound:*

$$\mathsf{AI}_{E_{n,k}}(F) \leq \min\left\{e;\ 2\binom{n}{e} > \binom{n}{k}\right\}.$$

Remark 5.3.10. *We investigate the behavior of this bound compared to the bound on $\mathbb{F}_2^n$: $\mathsf{AI}(F) \leq \lceil n/2 \rceil$. For $r > 0$, we have:*

$$2\binom{n}{k-r} = 2\binom{n}{k}\frac{k(k-1)\ldots(k-r+1)}{(n-k+r)\ldots(n-k+1)} = 2\binom{n}{k}\frac{k}{(n-k+r)}\cdots\frac{(k-r+1)}{(n-k+1)}.$$

Considering k such that:

$$k > \frac{(n+1)+2^{1/r}(r-1)}{1+2^{1/r}}, \quad \textit{is equivalent to} \quad \frac{(k-r+1)}{(n-k+1)} > \frac{1}{2^{1/r}},$$

then the r fractions are all superior to $2^{-1/r}$, therefore guaranteeing:

$$2\binom{n}{k-r} > \binom{n}{k}.$$

Fixing $k = n/2$, the condition on k can be propagated to n (even):

$$n > \frac{2+2(r-1)2^{1/r}}{2^{1/r}-1} \Longrightarrow 2\binom{n}{n/2-r} > \binom{n}{n/2}.$$

Hence, the best possible algebraic immunity of a function with constrained input Hamming weight is lower than for unconstrained functions.

With Theorem 5.3.8, we have the dimension of the image of $\psi_{n,k,d}$ and then of its kernel. We can exhibit more properties of its kernel: first we give a proposition identifying elements of this kernel, second we give a direct corollary of Theorem 5.3.8 giving a basis of the image of $\psi_{n,k,d}$ in particular cases.

Proposition 5.3.11. *Let k, r and n be non negative integers such that $r \leq k \leq n$, let $1 \leq i_1 < i_2 < \cdots < i_r \leq n$. Then, any n-variable Boolean function defined as:*

$$\begin{cases} x_{i_1}x_{i_2}\cdots x_{i_r}\left(\sum\limits_{j\neq i_1,i_2,\cdots,i_r} x_j\right) & \text{if } k-r \equiv 0 \mod 2, \\ x_{i_1}x_{i_2}\cdots x_{i_r}\left(1+\sum\limits_{j\neq i_1,i_2,\cdots,i_r} x_j\right) & \text{if } k-r \equiv 1 \mod 2, \end{cases}$$

is null on the set $E_{n,k}$.

More generally, let k, n, r and s be non negative integers such that $r \leq k \leq n$, and $1 \leq s \leq n-r$, let $1 \leq i_1 < i_2 < \cdots < i_r \leq n$ and note $I = \{i_1, \cdots, i_r\}$. Then, any n-variable Boolean function defined as:

$$\begin{cases} x_{i_1}x_{i_2}\cdots x_{i_r}\left(\sum\limits_{\{j_1,\ldots,j_s\}\cap I=\emptyset}\prod\limits_{\ell=1}^{s} x_{i_\ell}\right) & \text{if } \binom{k-r}{s} \equiv 0 \mod 2, \\ x_{i_1}x_{i_2}\cdots x_{i_r}\left(1+\sum\limits_{\{j_1,\ldots,j_s\}\cap I=\emptyset}\prod\limits_{\ell=1}^{s} x_{i_\ell}\right) & \text{if } \binom{k-r}{s} \equiv 1 \mod 2, \end{cases}$$

is null on $E_{n,k}$.

Proof. The first part of the proposition is a particular of the second one, corresponding to the case $s = 1$, therefore we prove the second part. Without loss of generality, we take $I = \{1, \cdots, r\}$ and denote f the function we study in this proposition. Let $x \in E_{n,k}$, we consider two cases. If the Hamming weight on the first r coordinates of x is less than r then $x_1x_2\cdots x_r = 0$ giving $f = 0$. In the other case, the Hamming weight of x on the first r coordinates is r, giving:

$$f(x) = f'(x_{r+1}, \cdots, x_n) = \left(\sum_{\{j_1,\ldots,j_s\}\cap I=\emptyset}\prod_{\ell=1}^{s} x_{i_\ell}\right) + \binom{k-r}{s} \mod 2$$

The Hamming weight on the $n-r$ other coordinates of x is $k-r$. The $n-r$ variables Boolean function f' described just above is an elementary symmetric Boolean function of degree s, then it is constant when the Hamming weight of the entry is fixed, and its value is $\binom{k-r}{s} \mod 2$. Therefore $f(x) = 0$ in this second case, in conclusion $f(x) = 0$ for all x of Hamming weight k. □

Corollary 5.3.12. *Let n, k, and d be non negative integers, and $\psi_{n,k,d}$ be the linear application defined as in Theorem 5.3.8 then the following holds:*

- *If $d \geq k$, then $\Im(\psi_{n,k,d}) = \mathbb{F}_2^{\binom{n}{k}}$ and a basis is the set of all the monomials of degree k.*

- *If $d \geq n-k$, then $\Im(\psi_{n,k,d}) = \mathbb{F}_2^{\binom{n}{k}}$ and a basis is the set of all the Boolean functions of the form $(1+x_{i_1})(1+x_{i_2})\cdots(1+x_{i_{n-k}})$ with $i_1 < i_2 < \cdots < i_{n-k}$.*

We noticed afterwards that the result of Theorem 5.3.8 have already been proven, independently, in the context of error-correcting codes. Recently, Dumer and Kapralova published results on *Spherically Punctured Reed-Muller Codes* [DK17], whose generator matrix corresponds to the matrix $M_{n,k,d}$. Our article [CMR17] was posted on Eprint before this publication, however it turns out that the rank of this matrix is already mentioned and proved in Olga Kapralova's PhD thesis, of 2013, together with results equivalent to Corollary 5.3.12. Their proof techniques are similar to ours, using the recursive decomposition, and based on older results [Got66; Wil90].

5.3.3 Algebraic Immunity Restricted To $E_{n,k}$ and Direct Sums

We investigate here the behavior of $\mathsf{AI}_{E_{n,k}}$ relatively to direct sums, and then prove a counter-intuitive relation between $\mathsf{AI}_{E_{n,k}}$ and the standard algebraic immunity for these constructions.

Lemma 5.3.13 (Direct Sum and $\mathsf{AI}_{E_{N,k}}$)**.** *Let F be the direct sum of f and g with n and m variables respectively, an $N = n+m$-variable Boolean function. Then for all $k \leq \min(n,m)$, $\mathsf{AI}_{E_{N,k}}(f)$ follows the bound :*

$$\mathsf{AI}_{E_{N,k}}(F) \geq \min_{0\leq\ell\leq k}(\max[\mathsf{AI}_{E_{n,\ell}}(f), \mathsf{AI}_{E_{m,k-\ell}}(g)]).$$

Proof. Suppose that we have h a non-zero annihilator of F on $E_{N,k}$, then we show that the restriction of h is a non-zero annihilator of f or $1+f$ on $E_{n,\ell}$ and of g or $1+g$ on $E_{m,k-\ell}$ for some ℓ such that $0 \leq \ell \leq k$.

For all $X \in E_{n+m,k}$, $h(X)F(X) = 0$, and h is non-null on $E_{n+m,k}$, so, there exists $\widetilde{X} \in E_{n+m,k}$ such that $h(\widetilde{X}) = 1$. We split $\widetilde{X}$ as $(\widetilde{x},\widetilde{y})$ where $\widetilde{x} \in \mathbb{F}_2^n$ and we define ℓ as $\ell = \mathsf{w_H}(\widetilde{x})$, giving the Hamming weight of $\widetilde{y} \in \mathbb{F}_2^m$ equal to $k-\ell$.

For this particular ℓ, we fix for $X \in E_{n+m,k}$ the x part (the first n coordinates) to the value $\widetilde{x}$ and we consider all possible values for $y \in \mathbb{F}_2^m$ of Hamming weight $k-\ell$. By doing so, it appears that h restricted to $\{(\widetilde{x},y) \mid y \in E_{m,k-\ell}\}$ is an annihilator of g or $1+g$ on $E_{m,k-\ell}$, and is non null by construction. We can also fix y to $\widetilde{y}$ and consider all possible values for x of Hamming weight ℓ and similarly find out that h restricted to $\{(x,\widetilde{y}) \mid x \in E_{n,\ell}\}$ is also a non-zero annihilator of f or $1+f$ on $E_{n,\ell}$. Therefore:

$$\deg(h) \geq \deg(h(X) \mid X \in E_{n+m,k}) \geq \max[\mathsf{AI}_{E_{n,\ell}}(f), \mathsf{AI}_{E_{m,k-\ell}}(g)].$$

Recalling that $\ell = \mathsf{w_H}(\widetilde{x})$, and then $0 \leq \ell \leq k$ finishes the proof. □

Moreover, for direct sums constructions we can link the standard algebraic immunity to the restricted algebraic immunity on $E_{N,k}$ for particular values of k, as explained by the following theorem.

Theorem 5.3.14 (Link between $\mathsf{AI}_{E_{N,k}}$ and AI in direct sum constructions)**.** *Let F be the direct sum of the two Boolean functions f and g with n and m variables respectively, and let k be such that $n \leq k \leq m$. Then the following relation holds:*

$$\mathsf{AI}_{E_{n+m,k}}(F) \geq \mathsf{AI}(f) - \deg(g).$$

Proof. Let $h(x,y)$ be a non-null annihilator of F over $E_{n+m,k}$, let $(a,b) \in \mathbb{F}_2^{n+m}$ have Hamming weight k and be such that $h(a,b)=1$. Since (a,b) has Hamming weight k, and $n \leq k \leq m$, we may, up to changing the order of the coordinates of b (and without loss of generality), assume that for every $j=1,\ldots,n$, we have $b_j = a_j + 1$ and for every $j=n+1,\ldots k$, we have $b_j = 1$ (so that for every $j=k+1,\ldots m$, we have $b_j = 0$). We define the following affine function over $\mathbb{F}_2^n$:

$$L(x) = (x_1+1, x_2+1, \ldots, x_n+1, 1, \ldots, 1, 0, \ldots, 0),$$

where the length of the part $(1,\ldots,1)$ equals $k-n$. We have $L(a)=b$, the n-variable function $h(x,L(x))$ is then non-zero and is an annihilator of $f(x)+g(L(x))$ over $\mathbb{F}_2^n$. If $g(b)=0$, then function $h(x,L(x))\,(g(L(x))+1)$ is a non-zero annihilator of f and has algebraic degree at most $\deg(h)+\deg(g)$; then we have:

$$\deg(h)+\deg(g) \geq AI(f).$$

If $g(b)=1$, then by applying the same reasoning to $f+1$ instead of f and $g+1$ instead of g, we obtain the same relation. If $h(x,y)$ is a non-null annihilator of $F+1$ over $E_{n+m,k}$, we have the same conclusion by replacing f by $f+1$ or g by $g+1$, this completes the proof. □

Remark 5.3.15. *Note that this theorem used jointly with Theorem 5.1.5 enables to derive lower bounds on* $\mathsf{AI}_{E_{N,k}}$ *for FLIP functions.*

The lower bound of this theorem proves in particular that, if $k \geq n$, then adding $m \geq k$ virtual variables to a function (taking $g=0$) does not lower the algebraic immunity with inputs of Hamming weight k with respect to the (global) original algebraic immunity. Note that this is already true (with no condition on n and m) when dealing with functions with no restriction on the input and is straightforward (as a direct consequence of Lemma 5.1.1 item 1 as example), while here it is less obvious. Notice that the bound of Theorem 5.3.14 is tight when $\deg(g)=0$: take for f a function whose algebraic immunity equals its algebraic degree; we have then $\mathsf{AI}_{E_{N,k}}(F) = \mathsf{AI}(f) = \deg(f)$, since it cannot be larger than the algebraic degree of f over $E_{n,k}$ (consequence of Corollary 5.3.4) which is at most equal to $\deg(f)$; the three parameters $\mathsf{AI}_{E_{N,k}}(F)$, the algebraic degree of f over $E_{n,k}$ and $\deg(f)$ are then equal.

Nevertheless, the bound of Theorem 5.3.14 also suggests that making the direct sum with a non-constant Boolean function g may lower the algebraic immunity over inputs of Hamming weight k with respect to the (global) original algebraic immunity. This may seem rather counter-intuitive, but it is true. Let us give an example:

$$f(x_1,x_2,x_3) = x_1 + x_2x_3, \text{ and } g(x_4,\ldots,x_{10}) = \sum_{i=4}^{10} x_i, \quad \text{and } k=5,$$

then $\mathsf{AI}(f) = \deg(f) = 2$, and $\mathsf{AI}(f) - \deg(g) = 1$. On $E_{10,5}$ we have:

$$x_2(f(x_1,x_2,x_3)+g(x_4,\ldots,x_{10})) = x_2(1+\sum_{i=1}^{10} x_i) = 0, \text{ and } x_2 \neq 0,$$

resulting to:

$$\mathsf{AI}_{E_{10,5}}(f+g) = 1 = \mathsf{AI}(f) - \deg(g),$$

the bound is tight here.

Making the direct sum with a non-constant Boolean function g may decrease drastically the algebraic immunity over inputs of Hamming weight k, take n odd:

$$f(x) = 1 + Maj_n(x), \text{ and } g(y) = Maj_n(y), \quad \text{and } k = n.$$

Maj_n has optimal algebraic immunity $\frac{n+1}{2}$, but $F(x,y) = f(x) + g(y)$ is null at fixed input weight n, because if $\mathsf{w_H}(x) + \mathsf{w_H}(y) = n$ then either $\mathsf{w_H}(x) \leq \frac{n-1}{2}$ and $\mathsf{w_H}(y) \geq \frac{n+1}{2}$ or $\mathsf{w_H}(x) \geq \frac{n+1}{2}$ and $\mathsf{w_H}(y) \leq \frac{n-1}{2}$.

It gives $\mathsf{AI}_{E_{2n,n}}(F) = 0$: we fall down to a null algebraic immunity with input Hamming weight n, however, the bound is not tight here because the algebraic degree of Maj_n is in general strictly larger than its algebraic immunity. Note that the same construction with g a constant function gives a different result. These examples of degradation of the standard algebraic immunity of direct sum constructions on constant-Hamming-weight sets finish our study on the restricted algebraic immunity.

5.4 Restricted Nonlinearity

A second criterion to consider on subsets of $\mathbb{F}_2^n$, which plays an important role for quantifying the contribution of the function to the resistance against attacks by affine approximations, like the fast correlation attack [MS88], is the nonlinearity $\mathsf{NL}(f)$. We shall denote it $\mathsf{NL}_E(f)$ when the input to f is taken from a set E rather than the whole set $\mathbb{F}_2^n$. It leads to the following definition:

Definition 5.4.1 (Restricted Nonlinearity)**.** *Let E be any subset of $\mathbb{F}_2^n$ and f any Boolean function defined over E. We call* nonlinearity of f over E *and denote by $\mathsf{NL}_E(f)$ the minimum Hamming distance between f and the restrictions to E of affine functions over $\mathbb{F}_2^n$.*

The nonlinearity of Boolean functions under non-uniform input distribution has been recently studied in [GGPS17], but the chosen distribution is binomial and there is no overlap with our work [CMR17] in this case. Posteriorly to our work, Sihem Mesnager improved the upper bound of the maximal restricted nonlinearity [Mes17] that we introduced; in Section 5.4.2.1 we present our results on this bound and we briefly explain how she improved it.

In this section we study the criterion of nonlinearity on restricted inputs; first we study the bound on the maximal nonlinearity reachable by a function on a restricted set, then we investigate the behavior of this bound for the fixed-Hamming-weight case. We give an error-correcting code perspective on these investigations, enabling to construct functions with a guaranteed amount of nonlinearity when the Hamming weight is fixed, then we show how direct sums can provide some nonlinearity in this setting. Finally, we show how fixing the Hamming weight can deteriorate the standard nonlinearity of a function, giving example of bent functions affine on all sets of fixed Hamming weight.

5.4.1 Nonlinearity Upper Bound for All Restricted Sets

As for the standard nonlinearity definition (Definition 2.4.9), we can give another definition linked to the Walsh transform, as shown by the following proposition.

Proposition 5.4.2. *For every n-variable Boolean function f over $\mathbb{F}_2^n$ and every non empty subset E of $\mathbb{F}_2^n$, we have:*

$$\mathsf{NL}_E(f) = \frac{|E|}{2} - \frac{1}{2}\max_{a\in\mathbb{F}_2^n} |\sum_{x\in E}(-1)^{f(x)+a\cdot x}|.$$

Proof. Let $a \in \mathbb{F}_2^2$ and $\varepsilon \in \mathbb{F}_2$, then any affine function (in n variables) is uniquely defined as $\ell(x) = a \cdot x + \varepsilon$. Denoting by $f_a(x)$ the sum (in $\mathbb{F}_2$) of $f(x)$ and $a \cdot x$, we have:

$$\sum_{x\in E}(-1)^{f(x)+a\cdot x} = \sum_{x\in E}(1-2f_a(x)),$$

and the Hamming distance between f and $a \cdot x$ on inputs ranging over E equals:

$$\sum_{x\in E} f_a(x) = \frac{|E|}{2} - \frac{1}{2}\sum_{x\in E}(-1)^{f(x)+a\cdot x},$$

where the sums are performed in $\mathbb{Z}$. Hence, the Hamming distance between f and ℓ over E equals:

$$\frac{|E|}{2} - \frac{(-1)^\epsilon}{2}\sum_{x\in E}(-1)^{f(x)+a\cdot x}.$$

Finally, NL_E, the minimum distance over E between f and all affine functions is therefore:

$$\mathsf{NL}_E(f) = \frac{|E|}{2} - \frac{1}{2}\max_{a\in\mathbb{F}_2^n} |\sum_{x\in E}(-1)^{f(x)+a\cdot x}|.$$

□

There exist nice properties (see *e.g.* [Car10]) of the Walsh transform that can be used in proofs. In particular, the *Parseval relation* $\sum_{a\in\mathbb{F}_2^n} W_f^2(a) = 2^{2n}$ leads to the so-called covering radius (upper) bound on the standard nonlinearity: $\mathsf{NL}(f) \leq 2^{n-1} - 2^{n/2-1}$. On a restricted set E, the same relation can be used, leading to a upper bound on NL_E:

Proposition 5.4.3. *For every subset E of $\mathbb{F}_2^n$ and every Boolean function f defined over E, we have:*

$$\mathsf{NL}_E(f) \leq \frac{|E|}{2} - \frac{\sqrt{|E|}}{2}.$$

Proof. We have:

$$\begin{aligned}\sum_{a\in\mathbb{F}_2^n}\left(\sum_{x\in E}(-1)^{f(x)+a\cdot x}\right)^2 &= \sum_{x,y\in E}(-1)^{f(x)+f(y)}\sum_{a\in\mathbb{F}_2^n}(-1)^{a\cdot(x+y)}\\ &= 2^n\,|E|.\end{aligned}$$

The first equation is obtained by changing the order of the two summations and applying the classical equality $(\sum_{i\in I} a_i)^2 = \sum_{i,j\in I} a_i a_j$ expressing the square of a summation. The second sum being not null only when $x+y=0$, we get the second equation. As the maximum of a sequence of numbers is always bounded below by the arithmetic mean, we deduce the final bound.

□

This bound when applied with $E = \mathbb{F}_2^n$ is the covering radius bound, reached by the bent functions. We show that for some sets E this bound can be improved.

Proposition 5.4.4. *Let E be any subset of $\mathbb{F}_2^n$, f a Boolean function over E, and F a vector-space where there exists v in $\mathbb{F}_2^n$ such that $v \cdot (x+y) = 1$ for all $(x,y) \in E^2$ satisfying $0 \neq x+y \in F^\perp$. Then we have:*

$$\mathsf{NL}_E(f) \leq \frac{|E|}{2} - \frac{1}{2}\sqrt{|E| + \lambda},$$

where

$$\lambda = |\sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}} (-1)^{f(x)+f(y)}|.$$

Proof. Let F be any vector subspace of $\mathbb{F}_2^n$. Then we have:

$$\begin{aligned}\sum_{a\in F}\left(\sum_{x\in E}(-1)^{f(x)+a\cdot x}\right)^2 &= \sum_{(x,y)\in E^2}(-1)^{f(x)+f(y)}\sum_{a\in F}(-1)^{a\cdot(x+y)} \\ &= |F|\sum_{\substack{(x,y)\in E^2 \\ x+y\in F^\perp}}(-1)^{f(x)+f(y)} \\ &= |F|\left(|E| + \sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}}(-1)^{f(x)+f(y)}\right),\end{aligned}$$

which implies:

$$\max_{a\in\mathbb{F}_2^n}|\sum_{x\in E}(-1)^{f(x)+a\cdot x}| \geq \sqrt{|E| + \sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}}(-1)^{f(x)+f(y)}},$$

and

$$\mathsf{NL}_E(f) \leq \frac{|E|}{2} - \frac{1}{2}\sqrt{|E| + \sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}}(-1)^{f(x)+f(y)}}.$$

Let us assume that there exists v in $\mathbb{F}_2^n$ such that, for all $(x,y) \in E^2$ such that $0 \neq x+y \in F^\perp$, we have $v \cdot (x+y) = 1$. Suppose that:

$$\sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}}(-1)^{f(x)+f(y)} = \lambda \neq 0.$$

Then λ may be without loss of generality assumed to be positive. Indeed, if λ is negative, then let v be as above, and let $f'(x) = f(x)+v\cdot x$, as for every $b \in \mathbb{F}_2^n$, denoting $f'(x) = f(x)+b\cdot x$, we have $\mathsf{NL}_E(f') = \mathsf{NL}_E(f)$; we obtain:

$$\sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}}(-1)^{f'(x)+f'(y)} = \sum_{\substack{(x,y)\in E^2 \\ 0\neq x+y\in F^\perp}}(-1)^{f(x)+f(y)+v\cdot(x+y)} = -\lambda > 0.$$

Therefore we deduce the bound of the proposition. □

Moreover, we can also take a family of vector-spaces $\mathcal{F}$, and the proposition above can then lead to the corollary below.

Corollary 5.4.5. *Let E be any subset of $\mathbb{F}_2^n$, f a Boolean function over E, and $\mathcal{F}$ a family of vector-spaces F for each of which there exists v in $\mathbb{F}_2^n$ such that $v \cdot (x+y) = 1$ for all $(x,y) \in E^2$ such that $0 \neq x+y \in F^\perp$. Then we have:*

$$\mathsf{NL}_E(f) \leq \frac{|E|}{2} - \frac{1}{2}\sqrt{|E| + \lambda},$$

where

$$\lambda = \max_{F \in \mathcal{F}} \Big| \sum_{\substack{(x,y) \in E^2 \\ 0 \neq x+y \in F^\perp}} (-1)^{f(x)+f(y)} \Big|.$$

In particular, taking for $\mathcal{F}$ the family of all linear hyperplanes of $\mathbb{F}_2^n$ (for which such v always exists since $F^\perp$ has dimension 1), we have:

Corollary 5.4.6. *Let E be a subset of $\mathbb{F}_2^n$ and f a Boolean function over E. Then:*

$$\mathsf{NL}_E(f) \leq \frac{|E|}{2} - \frac{1}{2}\sqrt{|E| + \lambda},$$

where

$$\lambda = \max_{a \in \mathbb{F}_2^n; a \neq 0} \Big| \sum_{\substack{(x,y) \in E^2 \\ x+y=a}} (-1)^{f(x)+f(y)} \Big|.$$

Remark 5.4.7. *Note that this result applied for $E = \mathbb{F}_2^n$ proves again that the derivatives of bent functions are all balanced.*

At the "13th International Conference on Finite Fields and their Applications" in 2017 Sihem Mesnager gave a talk called "On the nonlinearity of Boolean functions with restricted input" [Mes17] where she gave a better upper bound for particular sets E, using the power sums of Walsh transforms. For completeness we briefly describe here the core ideas of her presentation leading to this upper bound. To do so, we first define a restricted Walsh transform and the power sums of the restricted Walsh transform.

Definition 5.4.8 (Restricted Walsh Transform)**.** *Let $f \in \mathcal{B}_n$ be a Boolean function and E a subset of $\mathbb{F}_2^n$, the Walsh transform of f on E; $\hat{f}_{\chi,E}$ at $a \in \mathbb{F}_2^n$ is defined as:*

$$\hat{f}_{\chi,E} = \sum_{x \in E} (-1)^{f(x)+a \cdot x}.$$

Definition 5.4.9 (Power Sum of Restricted Walsh Transforms)**.** *Let f be a Boolean function in n variables, let E and F be non empty subsets of $\mathbb{F}_2^n$ and ℓ a nonnegative integer, we define the power sum of Walsh transforms $S_{2\ell}(f,E,F)$ as :*

$$S_{2\ell}(f,E,F) = \sum_{a \in F} \left(\hat{f}_{\chi,E}(a)\right)^{2\ell}.$$

Power sums of Walsh transforms have already been used to give upper bounds on covering radii in [CM07], here it enables to derive the following proposition:

Proposition 5.4.10. *Let f be a Boolean function in n variables, let E be a non empty subset of $\mathbb{F}_2^n$, for all $F \subseteq \mathbb{F}_2^n$ such that $S_2(f,E,F) \neq 0$ and for all nonnegative integers ℓ the following holds:*

$$\mathsf{NL}_E(f) \leq \frac{|E|}{2} - \frac{1}{2}\sqrt{\frac{S_{2\ell+2}(f,E,F)}{S_{2\ell}(f,E,F)}}.$$

Proof. First note that, for every strictly positive integer ℓ, $S_{2\ell+2}(f,E,F) = 0$ if and only if $S_{2\ell}(f,E,F) = 0$, and note that $S_0(f,E,F) \neq 0$ since F is non empty.

Then we bound the quotient of two consecutive power sums, as:

$$\sum_{a\in F}\left(\hat{f}_{\chi,E}(a)\right)^{2\ell+2} \leq \left(\max_{a\in F}\hat{f}_{\chi,E}(a)\right)^2 \sum_{a\in F}\left(\hat{f}_{\chi,E}(a)\right)^{2\ell},$$

we obtain the bound:

$$\frac{S_{2\ell+2}(f,E,F)}{S_{2\ell}(f,E,F)} \leq \left(\max_{a\in F}\hat{f}_{\chi,E}(a)\right)^2$$

Using the definition of the restricted Walsh transform and Proposition 5.4.2 enables to conclude.

□

With these notation, Sihem Mesnager showed that our analysis (Proposition 5.4.3 and Proposition 5.4.4 and their corollaries) focuses on the case $\ell = 0$: the behavior of $S_2(f,E,F)/S_0(f,E,F)$. Remarking that $\frac{S_{2\ell+2}(f,E,F)}{S_{2\ell}(f,E,F)}$ is a non-decreasing sequence, she analyses when:

$$\frac{S_{2\ell+2}(f,E,F)}{S_{2\ell}(f,E,F)} > \frac{S_2(f,E,F)}{S_0(f,E,F)} > 0,$$

where $\ell \geq 1$. For this case the fraction can be decomposed in 2 strictly positive terms, and she gives examples of subsets E where it leads to a better upper bound.

We think that further works in this direction could give an even better upper bound, and could extract cases where an upper bound is reached by a function. It concludes this part on the maximal nonlinearity over all restricted sets. We follow our study by focusing on sets of particular interest: $E_{n,k}$.

5.4.2 Nonlinearity Restricted To Fixed Hamming Weight Input

In this part we investigate more particularly the restricted nonlinearity on the sets $E_{n,k}$. First we examine the upper bound on the nonlinearity for this family of sets. Then, we consider this nonlinearity in the context of error-correcting codes to prove the existence of functions with nontrivial $\mathsf{NL}_{E_{n,k}}$. Finally we explore the behavior of direct sum constructions relatively to this criterion.

5.4.2.1 Nonlinearity Upper Bound for Fixed Hamming Weight Input

Let us now consider the case of $E = E_{n,k}$ for $k = 0,\ldots,n$, where $E_{n,k}$ is the set of vectors of Hamming weight k in $\mathbb{F}_2^n$. From Proposition 5.4.2 we have:

$$\mathsf{NL}_{E_{n,k}}(f) \leq \frac{\binom{n}{k}}{2} - \frac{1}{2}\sqrt{\binom{n}{k}}.$$

Note that this bound could be tight only if $\binom{n}{k}$ is a square, but we shall see that even in that case, it is not. Erdos showed the following theorem.

Theorem 5.4.11. *[AZ04] The equation $\binom{n}{k} = m^\ell$ has no integer solution with $\ell \geq 2$ and $3 \leq k \leq n-3$, except for $n = 3$ and $k = 3$ or $k = 47$.*

We can give the following proposition to stand on the bound of Equation 5.4.2.1.

Proposition 5.4.12. *For all n and $k \in \{1, \ldots, n-1\}$, Bound (5.4.2.1) is never tight, except maybe for two particular pairs (n, k): $(50, 3)$ and $(50, 47)$.*

Proof. Indeed, the bound can only be tight when $E_{n,k}$ is a square.

The only solution for $k = 3$ is $n = 50$, therefore we only consider the cases $k \in \{0, 1, 2, n-2, n-1, n\}$.

- $k = 0$ (or $k = n$): Proposition 5.4.3 gives $\mathsf{NL}_{E_{n,0}}(f) \leq 0$ which is tight because for all n and for all Boolean function f, f_k when $k = 0$ (or $k = n$) is constant.
- $k = 1$ (or $k = n-1$): $|E_{n,1}|$ is a square if and only if n is a square; every function restricted to its entries of Hamming weight 1 (or $n-1$) is linear (more details in Remark 5.4.15) therefore $\mathsf{NL}_{E_{n,1}}(f) = 0$ whereas the bound tells $\mathsf{NL}_{E_{n,1}}(f) \leq \frac{n-\sqrt{n}}{2}$.
- $k = 2$ (or $k = n-2$): We use here the Corollary 5.4.6, finding cases where λ cannot be null, it consists in finding a particular vector a such that:

$$a \in \mathbb{F}_2^n,\ a \neq 0, \quad \text{and} \sum_{\substack{(x,y) \in E_{n,k}^2 \\ x+y=a}} (-1)^{f(x)+f(y)} \neq 0.$$

Let us denote by i the Hamming weight of a, and by S the previous sum. If i is odd then $\{(x, y) \in E_{n,k}^2;\ x + y = a\}$ is empty forcing S to be null; otherwise, i is even, then $|\{(x, y) \in E_{n,k}^2;\ x + y = a\}|$ equals the number of possible choices (for building the support of x) of $i/2$ indexes in the support of a and of $k - i/2$ indexes outside the support of a. Then,

$$|\{(x, y) \in E_{n,k}^2;\ x + y = a\}| = \binom{i}{\frac{i}{2}} \binom{n-i}{k-\frac{i}{2}}.$$

Since the sum S is invariant when swapping x and y:

$$\left(\binom{i}{\frac{i}{2}} \binom{n-i}{k-\frac{i}{2}} \mod 4 \right) \neq 0 \Longrightarrow (S \mod 4) = 2,$$

as S equals twice the sum of an odd number of integers equal to ± 1. Now we consider various cases for i.

For $i = 2$:

$$\binom{i}{\frac{i}{2}} \binom{n-i}{2-\frac{i}{2}} = 2(n-2),$$

then, if n is odd, the sum S cannot be null.

For $i = 4$:

$$\binom{i}{\frac{i}{2}}\binom{n-i}{2-\frac{i}{2}} = 6\binom{n-4}{0},$$

then for $n \geq 4$ the sum S cannot be null.

By summing the two cases we can conclude on the case $k = 2$ (or $k = n-2$) we finish the proof, for all strictly positive n and for all Boolean functions f:

$$\mathsf{NL}_{E_{n,2}}(f) < \frac{|E_{n,2}| - \sqrt{|E_{n,2}|}}{2}.$$

□

This proposition shows that for the family of sets $E_{n,k}$ there are few chances to find functions reaching the upper bound of Proposition 5.4.3. Naturally the bound can be released to:

$$\mathsf{NL}_{E_{n,k}}(f) \leq \left\lfloor \frac{\binom{n}{k}}{2} - \frac{1}{2}\sqrt{\binom{n}{k}} \right\rfloor,$$

but it seems difficult to determine for which values of n this latter bound is tight. Moreover, the particular examples exhibited in [Mes17] where the bound of Proposition 5.4.4 cannot be tight are sets $E_{n,k}$. It makes think that the real maximal nonlinearity over most of the sets $E_{n,k}$ is much lower than the upper bounds proved up to date, and deserves further investigations.

5.4.2.2 Error Correcting Codes Perspective

In the following part we study the nonlinearity restricted to the sets $E_{n,k}$ from the error-correcting code theory. We use this perspective to prove the existence of functions with nontrivial restricted nonlinearity; the strong connection between restricted nonlinearity (of any strictly positive order) and punctured Reed-Muller codes could lead to more results on restricted criteria.

Reed-Muller codes $RM(r,n)$ are binary codes of length 2^n whose codewords are the evaluations of all Boolean functions in n variables of algebraic degree at most r on their 2^n entries. Fixing the Hamming weight of the entries gives particular punctured Reed-Muller codes whose characteristics are directly linked to Boolean functions with fixed weight entries. As Reed-Muller codes have been intensively studied in other contexts we do not describe fundamental new results in this part, we rather use another perspective to give interesting constructions and help to link our problematic to a quite well-known topic.

Definition 5.4.13. *For all $n \in \mathbb{N}^*; r, k \in [0,n]$ we denote by $RM(r,n)_{E_{n,k}}$ the punctured Reed-Muller code of length $\binom{n}{k}$ obtained by puncturing $RM(r,n)$ on all entries of Hamming weight different from k.*

Remark 5.4.14. *$RM(1,n)_{E_{n,k}}$ corresponds to the evaluation of all affine functions in n variables on entries of Hamming weight k; therefore, for every Boolean function f, $\mathsf{NL}_{E_{n,k}}(f)$ is the distance between f's truth table restricted to Hamming weight k entries and $RM(1,n)_{E_{n,k}}$. The maximal value of $\mathsf{NL}_{E_{n,k}}(f)$ when f ranges over the set of all Boolean functions equals the covering radius of $RM(1,n)_{E_{n,k}}$.*

Note also that the previous definition can be generalized to all non empty subsets E of $\mathbb{F}_2^n$; all punctured Reed-Muller code can then be written as $RM(r,n)_E$ and its covering radius is therefore NL_E.

In the next remark we exhibit the parameters of the code $RM(1,n)_{E_{n,k}}$; this provides a lower bound on the maximal value of $\mathsf{NL}_{E_{n,k}}$.

Remark 5.4.15. *$RM(1,n)_{E_{n,k}}$ is a linear code with parameters $[\binom{n}{k}, n, d]$ where*

$$d = \frac{\binom{n}{k} - \max_{(0<\ell\leq n/2)} \left|\sum_{i\in\mathbb{Z}} (-1)^i \binom{\ell}{i}\binom{n-\ell}{k-i}\right|}{2}.$$

Let $s(x) = \sum_{i\in I} x_i$ be any linear Boolean function whose restriction to the entries of Hamming weight k is non constant, and let $|I| = \ell$. We have $\ell \in \{1,\ldots,n-1\}$. The number of entries x of Hamming weight k such that $|supp(x)\cap I| = i$ equals $\binom{\ell}{i}\binom{n-\ell}{k-i}$. We deduce that the minimum distance of $RM(1,n)_{E_{n,k}}$ equals:

$$\min_{(0<\ell<n)} \left(\min\left(\sum_{i\ odd} \binom{\ell}{i}\binom{n-\ell}{k-i}, \sum_{i\ even} \binom{\ell}{i}\binom{n-\ell}{k-i}\right)\right) =$$

$$\frac{\min_{(0<\ell<n)} \left(\min\left(\sum_{i\in\mathbb{Z}} \binom{\ell}{i}\binom{n-\ell}{k-i} - \sum_{i\in\mathbb{Z}} (-1)^i \binom{\ell}{i}\binom{n-\ell}{k-i}, \sum_{i\in\mathbb{Z}} \binom{\ell}{i}\binom{n-\ell}{k-i} + \sum_{i\in\mathbb{Z}} (-1)^i \binom{\ell}{i}\binom{n-\ell}{k-i}\right)\right)}{2} =$$

$$\frac{\min_{(0<\ell<n)} \left(\sum_{i\in\mathbb{Z}} \binom{\ell}{i}\binom{n-\ell}{k-i} - \left|\sum_{i\in\mathbb{Z}} (-1)^i \binom{\ell}{i}\binom{n-\ell}{k-i}\right|\right)}{2}.$$

In other words, writing $P[X^k]$ for the coefficient of X^k in a polynomial $P(X)$, the minimum distance of $RM(1,n)_{E_{n,k}}$ equals:

$$\frac{\min_{(0<\ell<n)} \left((1+X)^\ell (1+X)^{n-\ell}[X^k] - \left|(1-X)^\ell (1+X)^{n-\ell}[X^k]\right|\right)}{2} =$$

$$\frac{\min_{(0<\ell<n)} \left(\binom{n}{k} - \left|(1-X)^\ell (1+X)^{n-\ell}[X^k]\right|\right)}{2} =$$

$$\frac{\binom{n}{k} - \max_{(0<\ell<n)} \left|\sum_{i\in\mathbb{Z}} (-1)^i \binom{\ell}{i}\binom{n-\ell}{k-i}\right|}{2}.$$

Note that $\left|\sum_{i\in\mathbb{Z}} (-1)^i \binom{\ell}{i}\binom{n-\ell}{k-i}\right|$ is invariant when changing ℓ into $n-\ell$ (by changing i into $k-i$); we can then replace $\max_{(0<\ell<n)}$ by $\max_{(0<\ell\leq n/2)}$.

Note that the maximal value of $\mathsf{NL}_{E_{n,k}}(f)$ when f ranges over the set of all Boolean functions (*i.e.* the covering radius of $RM(1,n)_k$) is bounded from below by $\frac{d}{2}$. Therefore, this remark proves that it is nonzero except for particular values of k, consequently it enables to build functions with nonlinearity reaching this lower bound.

We noticed afterwards that Dumer and Kapralova have, independently, already obtained the result of Remark 5.4.15. They studied these particular punctured Reed-Muller codes of order 1 in 2012 in term of *Spherically Punctured Bi-orthogonal Codes* [DK12] and more recently [DK13; DK17] they also studied the general case (order $r \geq 1$) as *Spherically*

Punctured Reed-Muller Codes. More particularly concerning Remark 5.4.15, they obtain the same result using a decomposition of the generator matrix similar to the one of Theorem 5.3.8, and they get a simpler expression of the minimal distance. Their work being in the optic of error correcting codes, they determine the parameters of these codes (length, dimension and minimal distance) and focus on efficient decoding algorithms; they do not present particular results on the covering radius, the major concern of our study on restricted nonlinearity.

5.4.2.3 Direct sum and $\mathsf{NL}_{E_{n,k}}$

In this part we focus on the behavior of direct sum constructions relatively to the nonlinearity restricted to the sets of all input of Hamming weight fixed to k. The first motivation for this focus is to know more on the parameters of FLIP functions. Indeed, knowing the parameters of simple functions and a lower bound for direct sums construction enables to derive security bounds when FLIP functions are use in a Filter Permutator construction.

Lemma 5.4.16 (Direct sum and $\mathsf{NL}_{E_{N,k}}$)**.** *Let F be the direct sum of f and g of n and m variables respectively, we have:*

$$\mathsf{NL}_{E_{N,k}}(F) \geq \sum_{i=0}^{k} \binom{n}{i} \mathsf{NL}_{E_{m,k-i}}(g) + \sum_{i=0}^{k} \mathsf{NL}_{E_{n,i}}(f) \left(\binom{m}{k-i} - 2\mathsf{NL}_{E_{m,k-i}}(g) \right).$$

Proof. We have:

$$\begin{aligned}
\mathsf{NL}_{E_{N,k}}(F) &= \frac{\binom{N}{k}}{2} - \frac{1}{2} \max_{(a,b)\in\mathbb{F}_2^n\times\mathbb{F}_2^m} \left| \sum_{(x,y)\in E_{N,k}} (-1)^{F(x,y)+a\cdot x+b\cdot y} \right| \\
&\geq \frac{\binom{N}{k}}{2} - \frac{1}{2} \max_{a\in\mathbb{F}_2^n, b\in\mathbb{F}_2^m} \sum_{i=0}^{k} \left| \sum_{x\in E_{n,i},\, y\in E_{m,k-i}} (-1)^{f(x)+g(y)+a\cdot x+b\cdot y} \right| \\
&= \frac{\binom{N}{k}}{2} - \frac{1}{2} \max_{a\in\mathbb{F}_2^n, b\in\mathbb{F}_2^m} \sum_{i=0}^{k} \left| \sum_{x\in E_{n,i}} (-1)^{f(x)+a\cdot x} \right| \left| \sum_{y\in E_{m,k-i}} (-1)^{g(y)+b\cdot y} \right| \\
&\geq \frac{\binom{N}{k}}{2} - \frac{1}{2} \sum_{i=0}^{k} \left(\max_{a\in\mathbb{F}_2^n} \left| \sum_{x\in E_{n,i}} (-1)^{f(x)+a\cdot x} \right| \right) \left(\max_{b\in\mathbb{F}_2^m} \left| \sum_{y\in E_{m,k-i}} (-1)^{g(y)+b\cdot y} \right| \right) \\
&= \frac{\binom{N}{k}}{2} - \frac{1}{2} \sum_{i=0}^{k} \left(\binom{n}{i} - 2\mathsf{NL}_{E_{n,i}}(f) \right) \left(\binom{m}{k-i} - 2\mathsf{NL}_{E_{m,k-i}}(g) \right) \\
&= \sum_{i=0}^{k} \binom{n}{i} \mathsf{NL}_{E_{m,k-i}}(g) + \sum_{i=0}^{k} \binom{m}{k-i} \mathsf{NL}_{E_{n,i}}(f) - 2\sum_{i=0}^{k} \mathsf{NL}_{E_{n,i}}(f)\mathsf{NL}_{E_{m,k-i}}(g) \\
&= \sum_{i=0}^{k} \binom{n}{i} \mathsf{NL}_{E_{m,k-i}}(g) + \sum_{i=0}^{k} \mathsf{NL}_{E_{n,i}}(f) \left(\binom{m}{k-i} - 2\mathsf{NL}_{E_{m,k-i}}(g) \right).
\end{aligned}$$

Equation (5.5) is obtained using Vandermonde convolution (see Proposition 2.5.5).

□

Although this inequality does not provide a tight bound, it enables to guarantee some nonlinearity on fixed Hamming weight input of a function from two simpler functions with high nonlinearity in this context.

5.4.3 Deterioration of Functions with Optimal Standard Nonlinearity

Fixing the input Hamming weight may deteriorate in an extreme way the nonlinearity of a Boolean function, we examine in this part some Boolean functions with optimal standard nonlinearity (bent functions) with trivial restricted nonlinearity (defined on the sets $E_{N,k}$).

Proposition 5.4.17. *For every n, there exist n-variable bent functions f such that, for every $k = 0, \ldots, n$, $\mathsf{NL}_{E_{n,k}}(f) = 0$.*

Proof. This is for instance the case of the function $f(x) = \binom{\mathsf{w_H}(x)}{2} = \sum_{1 \leq i < j \leq n} x_i x_j$. This function is, up to the addition of an affine function, the only bent symmetric function (see *e.g.* [Car10]). Since it is symmetric, fixing the Hamming weight of its input makes it constant and therefore with null nonlinearity. □

More generally, it would be interesting to characterize those bent functions whose restrictions to $E_{n,k}$ have null nonlinearity (*i.e.* are affine), for every k. This task seems very difficult but we are able to achieve it in the particular case of quadratic functions. We begin with a proposition characterizing the functions with null $\mathsf{NL}_{E_{n,k}}$ for every k.

Proposition 5.4.18. *Let f be a Boolean function in n variables, the following holds:*

$$\forall k \; \mathsf{NL}_{E_{n,k}}(f) = 0 \iff f(x) = \ell'_0(x) + \sum_{i=1}^{n} \sigma_i(x) \ell'_i(x),$$

where the ℓ'_i's are all affine functions and σ_i are the elementary symmetric functions in n variables.

Proof. First, a Boolean function satisfies $\mathsf{NL}_{E_{n,k}}(f) = 0$ for every k if all its restrictions to $E_{n,k}$ are affine by definition. Such function can be expressed in terms of indicator functions of the $n+1$ sets $E_{n,k}$ and affine functions. As an indicator function of $E_{n,k}$ is a symmetric function, and as a sum of symmetric functions is still a symmetric function, we can use the following equivalence. The $n+1$ (weightwise) restrictions of a Boolean function are affine if and only if there exist symmetric Boolean functions $\varphi_0, \varphi_1, \ldots, \varphi_n$ such that:

$$f(x) = \varphi_0(x) + \sum_{i=1}^{n} \varphi_i(x)\, x_i.$$

Using the fundamental theorem of symmetric polynomials, any symmetric Boolean functions $\varphi(x)$ can be written in the form $\ell \circ \Sigma(x)$ where ℓ is affine and Σ is the vectorial (n,n)-function whose ith coordinate function is the elementary symmetric function $\sigma_i = \sum_{1 \leq j_1 < \cdots < j_i \leq n} \prod_{l=1}^{i} x_{j_l}$.

We deduce that f satisfies $\mathsf{NL}_{E_{n,k}}(f) = 0$ for every k if and only if it has the form $f(x) = \ell_0 \circ \Sigma(x) + \sum_{i=1}^{n} \ell_i \circ \Sigma(x) x_i$, where the ℓ_i's are affine.

Gathering all the terms in this expression which involve each elementary symmetric function σ_i we obtain the result:

$$f(x) = \ell'_0(x) + \sum_{i=1}^{n} \sigma_i(x) \ell'_i(x).$$

□

Now, we can characterize all quadratic bent functions which are affine on all sets of constant input weight in the following proposition.

Proposition 5.4.19 (Characterization of quadratic bent functions with null $\mathsf{NL}_{E_{n,k}}$ for all k)**.** *For every even $n \geq 4$, the quadratic bent functions satisfying $\mathsf{NL}_{E_{n,k}}(f) = 0$ for every k are those functions of the form $f(x) = \sigma_1(x)\ell(x) + \sigma_2(x)$, where $\ell(1, \ldots, 1) = 0$.*

Proof. According to Proposition 5.4.18, a quadratic function satisfies $\mathsf{NL}_{E_{n,k}}(f) = 0$ for every k if and only if, up to the addition of an affine function, it has the form:

$$f(x) = \sigma_1(x)\ell(x) + \varepsilon\sigma_2(x),$$

where ℓ is linear and $\varepsilon \in \mathbb{F}_2$.

The symplectic form associated to $(x, y) \mapsto f(x + y) + f(x) + f(y) + f(0)$ (see *e.g.* [Car10] p.70) equals:

$$\sigma_1(y)\ell(x) + \sigma_1(x)\ell(y) + \varepsilon \sum_{1 \leq j \neq i \leq n} x_j y_i.$$

Denoting $\ell(x) = \sum_{i=1}^n l_i x_i$, the kernel

$$E = \{x \in \mathbb{F}_2^n; \forall y \in \mathbb{F}_2^n, f(x + y) + f(x) + f(y) + f(0) = 0\}$$

of this symplectic form is the vector space of equations:

$$(L_i) : \ell(x) + l_i \sum_{j=1}^n x_j + \varepsilon \sum_{j \neq i} x_j = 0,$$

where i ranges from 1 to n. The sum $L_i + L_{i'}$ of two of these equations equals

$$(L_i + L_{i'}) : (l_i + l_{i'}) \sum_{j=1}^n x_j + \varepsilon(x_i + x_{i'}) = 0.$$

If $l_i = l_{i'}$ we obtain: $\forall x \in E$, $x_i = x_{i'}$ if $\varepsilon = 1$ and no condition on $x \in E$ otherwise. If $l_i \neq l_{i'}$, we obtain: $\forall x \in E$, $\sum_{j=1}^n x_j = x_i + x_{i'}$ if $\varepsilon = 1$ and $\forall x \in E$, $\sum_{j=1}^n x_j = 0$ otherwise. Hence, denoting

$$I = \{i = 1, \ldots, n; l_i = 0\},$$

we have that, if $\varepsilon = 1$, then all the coordinates of indexes $i \in I$ of an element of E are equal to some bit η and all those such that $i \in I^c$ are equal to $\eta + \sum_{j=1}^n x_j$, and if $\varepsilon = 0$, there is no condition on $x \in E$ if $I = \emptyset$ or $I = \{1, \ldots, n\}$ and if $I \neq \emptyset, \{1, \ldots, n\}$, the condition is $\sum_{j=1}^n x_j = 0$. We then have two cases:

- if $x \in E$ is such that $\sum_{j=1}^n x_j = 0$ then:
  - if $\varepsilon = 1$, then either all x_i's are null, in which case (L_i) is satisfied, or all are equal to 1, in which case (L_i) becomes (since n is even) $\ell(1, \ldots, 1) = 1$; hence, if this latter equality is true (i.e. if I has odd cardinality), $E \neq \{0\}$;
  - if $\varepsilon = 0$ then all equations L_i are equal to $\ell(x) = 0$; then $E \neq \{0\}$ unless the hyperplane $\ker \ell$ has a trivial intersection with the hyperplane of equation $\sum_{j=1}^n x_j = 0$, which is possible only if $n = 2$; the case $\varepsilon = 0$ is then compatible with f bent only for $n = 2$; we shall not consider it anymore.

- if $x \in E$ is such that $\sum_{j=1}^{n} x_j = 1$ then if $\varepsilon = 1$, all x_i's such that $i \in I$ are equal to η and those x_i's such that $i \in I^c$ are equal to $\eta + 1$, which implies $\eta|I| + (\eta+1)|I^c| = |I^c| = 1 \pmod 2$; hence I has odd cardinality and we have seen that $E \neq \{0\}$ in such case.

The only case where f is bent, *i.e.* where $E = \{0\}$, for $n \geq 4$, is then $\begin{cases} \varepsilon = 1 \\ \ell(1,\ldots,1) = 0 \end{cases}$, finishing the proof.

□

Note that all the functions above are EA-equivalent to each others (two n-variable Boolean functions f and g are called EA-equivalent if there exist an affine automorphism L over $\mathbb{F}_2^n$ and an affine n-variable function ℓ such that $f = g \circ L + \ell$). EA-equivalence is does not preserve the Hamming weight, so the nonlinearity degradation on constant-Hamming-weight sets cannot be studied equivalence class by equivalence class.

These examples of functions with the best nonlinearity in the general context and worst nonlinearity in the fixed-Hamming-weight context finishes our study on the restricted nonlinearity.

5.5 Restricted Balancedness

A first requirement on cryptographic Boolean functions is to be balanced or at least almost balanced. We shall then be focus on those functions which are balanced on the input set E, which corresponds to a notion of balancedness (for any set of even cardinality), where a function is balanced over E if it takes the value 1 for exactly half of the elements of E. But since E may change in the process (this is not the case in FLIP but it could be in a variant), we are interested in Boolean functions whose restrictions to all sets E in some family $\mathcal{E}$ are balanced. Even if E does not change, we may wish to have a Boolean function which is balanced on a family of sets E, so that it can be used in a variety of situations. Given some family $\mathcal{E}$ of subsets of $\mathbb{F}_2^n$, we shall say that a Boolean function f is perfectly balanced over $\mathcal{E}$ if its restriction to any set $E \in \mathcal{E}$ of even size is balanced, and this is the notion of balancedness we study in this section. We shall be in particular interested in the family of sets of all elements of fixed Hamming weight: $\mathcal{E} = \{E_{n,0}, E_{n,1}, \ldots, E_{n,n-1}, E_{n,n}\}$, where $E_{n,k} = \{x \in \mathbb{F}_2^n; \mathsf{w_H}(x) = k\}$. We shall then call such functions *weightwise perfectly balanced*.

As we center our study of the restricted balancedness on this family, we use the following notation. We denote by $\mathsf{w_H}(f)_k$ the Hamming weight of the evaluation vector of the function f on all the entries of fixed Hamming weight k:

$$\mathsf{w_H}(f)_k = |\{x \in \mathbb{F}_2^n, \mathsf{w_H}(x) = k, f(x) = 1\}|,$$

where $\mathsf{w_H}$ denotes the Hamming weight. We accordingly denote $\overline{\mathsf{w_H}(f)_k} = |\{x, \mathsf{w_H}(x) = k, f(x) = 0\}| = \binom{n}{k} - \mathsf{w_H}(f)_k$.

Definition 5.5.1. *Let f be a Boolean function defined over $\mathbb{F}_2^n$. It will be called* weightwise perfectly balanced *(WPB) if, for every $k \in \{1,\ldots,n-1\}$, the restriction of f to $E_{n,k}$, is balanced, that is, $\forall k \in [1, n-1], \mathsf{w_H}(f)_k = \frac{\binom{n}{k}}{2}$.*

To make the function balanced on its whole domain $\mathbb{F}_2^n$, we additionally impose that $f(0,\ldots,0) \neq f(1,\ldots,1)$ and more precisely that:

$$f(0,\ldots,0) = 0; \quad f(1,\ldots,1) = 1.$$

This last constraint does not reduce the generality (when $f(0,\ldots,0) \neq f(1,\ldots,1)$), up to the addition of constant 1 to f, and it makes some constructions clearer. Note that weightwise perfectly balanced Boolean functions exist only if, for every $k \in [1, n-1]$, $\binom{n}{k}$ is even and this property is satisfied if and only if n is a power of 2. Note that for these n, $\mathsf{w_H}(f)_k = \frac{\binom{2^\ell}{k}}{2}$ is then even for $k \in [1,\ldots,2^{\ell-1}-1] \cup [2^{\ell-1}+1,\ldots,2^\ell-1]$ and odd for $k = 2^{\ell-1} = n/2$, this property will be used in the rest of the section. To be able to address the case where n is not a power of 2, we introduce:

Definition 5.5.2. *Let f be a Boolean function defined over $\mathbb{F}_2^n$. It will be called* weightwise almost perfectly balanced functions *(WAPB) if, for every $k \in [1, n-1]$, $\mathsf{w_H}(f)_k = \frac{\binom{n}{k}}{2}$ when $\binom{n}{k}$ is even and $\mathsf{w_H}(f)_k = \frac{\binom{n}{k}\pm 1}{2}$ when $\binom{n}{k}$ is odd.*

The bias of a stream cipher output in presence of Hamming weight leakage is considered in [JD06]. Precisely, it is shown that knowing the Hamming weight of a register when the updating function is an LFSR in a particular representation enables to distinguish the keystream from a random binary stream, and the authors also describe a correlation attack in this setting. This result is therefore connected to our study on restricted balancedness: they use the balancedness flaw of a function on the sets $\{x \mid \mathsf{w_H}(x) = k\}$. They exploit the fact that this function is not weightwise (almost) perfectly balanced, and combine it with other equations to mount a fast correlation attack on these LFSR (similarly to the attack presented in Section 4.5.3.3).

We focus our study on necessary conditions for Boolean functions to be *WPB* or *WAPB* and on the construction of such functions. Note that contrarily to balanced functions the weightwise (almost) perfect balanced functions represent a negligible proportion of the Boolean functions in n variables. The next remark shows that these functions can be quite different than the functions usually used in cryptography for their balancedness, as highly resilient functions.

Remark 5.5.3. *For all $n \geq 2$, there exists an $(n-1)$-resilient function (i.e. a balanced Boolean function which remains balanced when at most $n-1$ of its variables are arbitrarily fixed) which is unbalanced for all weights $k \in [1, n-1]$.*

Indeed, the first elementary symmetric Boolean function $\sigma_1 = \sum_{i=1}^n x_i = \mathsf{w_H}(x) \mod 2$ is $(n-1)$-resilient and is constant on all fixed-weight inputs, its weightwise restrictions are as much unbalanced as possible.

The goal of this study on the restricted balancedness is therefore to identify and construct *WPB* and *WAPB* functions for all strictly positive n. First we examine some relations between ANF and weightwise balancedness, second we present constructions of *WPB* and *WAPB* functions.

5.5.1 Weightwise Balancedness and ANF

In the following, we give more insights on necessary conditions on the ANF of *WPB* or *WAPB*. We begin by determining the possible number of monomials of degree 1 and 2.

Proposition 5.5.4. *If f is a weightwise (almost) perfectly balanced Boolean function of n variables then the ANF of f contains $\lceil n/2 \rfloor$ monomials of degree 1 and at least $\lfloor n/4 \rfloor$ monomials of degree 2, where $\lceil n/2 \rfloor$ equals $n/2$ if n is even and $(n \pm 1)/2$ if n is odd.*

Proof. In the particular case where f is linear, $\mathsf{w_H}(f)_k$ is exactly the number of entries of weight k for which an odd number of the monomials of f are set to 1. Therefore denoting by d the number of (degree-1) monomials in the ANF of f, we have:

$$\mathsf{w_H}(f)_k = \sum_{i \text{ odd}} \binom{d}{i}\binom{n-d}{k-i}.$$

For any function f, as $\mathsf{w_H}(f)_k$ is only determined by the monomials of f of degree at most k, let us partition f into ℓ_f, q_f and f', respectively made of the monomials of degree 1, 2 and strictly larger than 2 in the ANF of f. For $k = 1$, we have:

$$\mathsf{w_H}(f)_1 = \mathsf{w_H}(\ell_f)_1 = \binom{|\ell_f|}{1},$$

where $|\ell_f|$ is the number of monomials of ℓ_f.

Therefore, if f is (almost) balanced for fixed weight 1, then $|\ell_f| = \frac{n}{2}$ for n even and $|\ell_f| = \frac{n\pm1}{2}$ for n odd. We have:

$$\mathsf{w_H}(f)_2 = \mathsf{w_H}(\ell_f + q_f)_2 = \mathsf{w_H}(\ell_f)_2 + \mathsf{w_H}(q_f)_2 - 2\mathsf{w_H}(\ell_f \cdot q_f)_2.$$

As $\mathsf{w_H}(q_f)_2 \geq 2\mathsf{w_H}(\ell_f \cdot q_f)_2 - \mathsf{w_H}(q_f)_2$ it implies $\mathsf{w_H}(q_f)_2 \geq \mathsf{w_H}(\ell_f)_2 - \mathsf{w_H}(f)_2$.
Therefore, if f is (almost) balanced for fixed weights 1 and 2, then:

- for n even:

$$\mathsf{w_H}(\ell_f)_2 = \binom{\frac{n}{2}}{1}\binom{\frac{n}{2}}{1} = \frac{n^2}{4}, \text{ and } \mathsf{w_H}(\ell_f)_2 - \frac{\binom{n}{2}}{2} = \frac{n}{4}, \text{ so } \mathsf{w_H}(q_f)_2 \geq \left\lfloor \frac{n}{4} \right\rfloor,$$

- for n odd:

$$\mathsf{w_H}(\ell_f)_2 = \binom{\frac{n+1}{2}}{1}\binom{\frac{n-1}{2}}{1} = \frac{n^2-1}{4}, \text{ and } \mathsf{w_H}(\ell_f)_2 - \frac{\binom{n}{2}}{2} = \frac{n-1}{4}, \text{ so } \mathsf{w_H}(q_f)_2 \geq \left\lfloor \frac{n}{4} \right\rfloor.$$

□

We can prove that all weightwise perfectly balanced functions ha an algebraic degree of at least $n/2$ (giving an upper bound on their resiliency).

Proposition 5.5.5. *If f is a weightwise perfectly balanced Boolean function of n variables, then the ANF of f contains at least one monomial of degree $n/2$.*

Proof. Let m_d be a monomial of degree d, we focus on the parity of $\mathsf{w_H}(m_d)_k$; for all $1 \leq k \leq n-1$ and $1 \leq d \leq k$:

$$\mathsf{w_H}(m_d)_k = \binom{n-d}{k-d}$$

More particularly when $k = d$, $\mathsf{w_H}(m_k)_k = \binom{n-k}{0} = 1$. We have seen that f being weightwise perfectly balanced implies that $n = 2^\ell$ and therefore we can determine the parity of $\mathsf{w_H}(f)_k = \frac{\binom{2^\ell}{k}}{2}$:

$$\frac{\binom{2^\ell}{k}}{2} \mod 2 = \begin{cases} 0 \text{ if } k \in [1, \ldots, 2^{\ell-1}-1,] \cup [2^{\ell-1}+1, \ldots, 2^\ell - 1] \\ 1 \text{ if } k = 2^{\ell-1} = n/2. \end{cases}$$

This enables to determine the parity of the number of monomials of each degree of f smaller than or equal to $2^{\ell-1} = n/2$. As $\mathsf{w_H}(m_k)_k = 1$, and $\mathsf{w_H}(f)_k$ depends only on the monomials of degree less than or equal to k, we first show by induction that the number of monomials of degree k is even for k such that $1 \leq k < n/2$. The principle of this proof is to decompose a function as a sum, and then to study the parity of its weight using the weight of its summands.

For $k = 1$, only the degree-1 monomials determine $\mathsf{w_H}(f)_1$, as $\mathsf{w_H}(m_1)_1 = 1$ (and as the supports of two degree-1 monomials on $E_{n,1}$ are disjoint), only a sum of an even number of degree-1 monomials gives an even $\mathsf{w_H}(f)_1$. From k to $k+1$ (with $k < n/2+1$), we denote g the part of the function containing all these monomials and h the part containing the monomials of degree $k+1$. The following relation holds:

$$\mathsf{w_H}(f)_{k+1} = \mathsf{w_H}(g)_{k+1} + \mathsf{w_H}(h)_{k+1} - 2\mathsf{w_H}(gh)_{k+1}. \tag{5.5}$$

First, we show than $\mathsf{w_H}(g)_{k+1}$ has even parity. We can express $\mathsf{w_H}(g)_{k+1}$ using the decomposition of g in k terms, each term g_i corresponding to all the monomials of g of degree i with $1 \leq i \leq k$. Indeed, $\mathsf{w_H}(g)_{k+1}$ is equal to the sum of all the $\mathsf{w_H}(g_i)_{k+1}$ minus twice the cross-terms as in Equation (5.5). Each term g_i is the sum of an even number of monomials of degree i by hypothesis, then $\mathsf{w_H}(g_i)_{k+1}$ is equal to the sum of an even number of times $\mathsf{w_H}(m_i)_{k+1}$ minus twice the cross-terms. Then each $\mathsf{w_H}(g_i)_{k+1}$ is even, and $\mathsf{w_H}(g)_{k+1}$ is even.

As $\mathsf{w_H}(g)_{k+1}$ is even, $\mathsf{w_H}(f)_{k+1}$ has the same parity as $\mathsf{w_H}(h)_{k+1}$. As h is the part containing the monomials of degree $k+1$, and $w(m_{k+1})_{k+1} = 1$, using the same decomposition technique, the parity of $\mathsf{w_H}(h)_{k+1}$ is equal to the parity of the number of monomials of degree $k+1$. The parity of the number of monomials of degree $k+1$ is then equal to the parity of $\mathsf{w_H}(f)_{k+1}$ which is even (as $k+1 < n/2$), finishing the induction.

Finally, for $k = n/2$ we can apply the same strategy of decomposition, $\mathsf{w_H}(g)_{n/2}$ is still even in this case. Consequently the number of monomials of degree $k+1$ is then odd, equal to the parity of $\mathsf{w_H}(f)_{n/2}$. It implies that the ANF of f contains at least one monomial of degree $n/2$. Note that as $\mathsf{w_H}(f)_{n/2}$ is odd, we cannot use the same strategy to determine the parity of the number of monomials of higher degree.

□

Remark 5.5.6. *The precedent results show that the ANF of every weightwise perfectly balanced function of 2^ℓ variables contains monomials of degrees 2^0, 2^1 and $2^{\ell-1}$. This raises the question if having monomials of degree all powers of 2 is a necessary condition for weightwise perfectly balanced function. It turns out that it is not necessary, for example the function of 16 variables and algebraic degree 3:*

$$f = \sum_{i=1}^{8} x_i + \sum_{i=1}^{4} x_i x_{i+8} + x_1 x_2 x_5 + x_1 x_4 x_{16}$$

is weightwise balanced for $k \in [0, \cdots, 4]$ and can be completed in a weightwise perfectly balanced function only by adding monomials of degree > 4.

All the necessary conditions for weightwise (almost) perfect balancedness presented above are not sufficient; thus we investigate the constructions of such functions in the next part.

5.5.2 Constructions of Weightwise (Almost) Perfectly Balanced Functions

The direct sum construction (see Definition 2.4.13) can be a starting point to build weightwise perfectly balanced function. This secondary construction does not build a weightwise perfectly balanced function from two weightwise perfectly balanced functions as we can see from the next lemma and corollary.

Lemma 5.5.7. *Let f be the direct sum of g_1 and g_2 each one in $2^{\ell-1}$ variables such that $\ell \in \mathbb{N}^*$, and such that:*

$$g_1(0\dots0) + g_1(1\dots1) + g_2(0\dots0) + g_2(1\dots1) \equiv 0 \mod 2,$$

then f cannot be weightwise perfectly balanced.

Proof. As f is a direct sum of g_1 and g_2, for every $k \in [1, n-1]$ we can link the value of $\mathsf{w_H}(f)_k$ to $\mathsf{w_H}(g_1)_i$ and $\mathsf{w_H}(g_2)_{k-1}$ with $i \leq k$.

First, we do a partition of the entries of f of Hamming weight k depending on the Hamming weight of the entries of g_1 and g_2, this gives a partition in $k+1$ sets where g_1 is evaluated on $E_{n_1,i}$ and g_2 is evaluated on $E_{n_2,k-i}$.

Then, $f(x_1, \dots, x_n) = 1$ is equivalent to $g_1(x_1, \dots, x_{n_1}) \neq g_2(x_{n_1+1}, \dots, x_n)$, so we can link $\mathsf{w_H}(f)_k$ to the number of entries where g_1 gives 1 and g_2 gives 0 plus the number of entries where g_1 gives 0 and g_2 gives 1. Finally we obtain:

$$\mathsf{w_H}(f)_k = \sum_{i=0}^{k} \mathsf{w_H}(g_1)_i \left(\binom{n_2}{k-i} - \mathsf{w_H}(g_2)_{k-i} \right) + \mathsf{w_H}(g_2)_{k-i} \left(\binom{n_1}{i} - \mathsf{w_H}(g_1)_i \right)$$

Now we suppose that f is weightwise perfectly balanced and we use that $n_1 = n_2 = \frac{n}{2}$; in particular, $\mathsf{w_H}(f)_{\frac{n}{2}} = \frac{1}{2}\binom{n}{\frac{n}{2}} \equiv 1 \mod 2$ and developing:

$$\mathsf{w_H}(f)_{\frac{n}{2}} = \sum_{i=0}^{n/2} \binom{\frac{n}{2}}{\frac{n}{2}-i} \mathsf{w_H}(g_1)_i + \binom{\frac{n}{2}}{i} \mathsf{w_H}(g_2)_{\frac{n}{2}-i} - 2\mathsf{w_H}(g_1)_i \mathsf{w_H}(g_2)_{\frac{n}{2}-i}$$

Moreover, we know that, as $\frac{n}{2}$ is also a power of 2, then for each $i \in [1, \frac{n}{2}-1]$, $\binom{n/2}{i}$ is even. To conclude, if f is weightwise perfectly balanced, then we have the following relation:

$$1 \equiv \mathsf{w_H}(g_1)_0 + \mathsf{w_H}(g_1)_{\frac{n}{2}} + \mathsf{w_H}(g_2)_0 + \mathsf{w_H}(g_2)_{\frac{n}{2}} \mod 2$$

Then we need that $g_1(0\dots0) + g_1(1\dots1) + g_2(0\dots0) + g_2(1\dots1) \equiv 1 \mod 2$ □

The corollary below is a direct consequence:

Corollary 5.5.8. *If $g_1(x_1, \dots, x_{\frac{n}{2}})$ and $g_2(x_{\frac{n}{2}+1}, \dots, x_n)$ are two weightwise perfectly balanced functions, then the Boolean function defined by the direct sum of g_1 and g_2 cannot be weightwise perfectly balanced.*

Hence, the direct sum, when applied to perfectly balanced functions, does not lead to a weightwise perfectly balanced function. It comes from the constraint $\mathsf{w_H}(f)_0 \neq \mathsf{w_H}(f)_n$ guarantying the classical balancedness (*i.e.* on $\mathbb{F}_2^n$) of a WPB function. Nevertheless we can derive such construction from weightwise perfectly balanced functions by applying the direct

sum after modifying one of the functions: if f and g are two n-variable weightwise perfectly balanced functions, then:

$$h(x,y) = f(x) + \prod_{i=1}^{n} x_i + g(y),$$

is a $2n$-variable weightwise perfectly balanced function. In fact, this result is a particular case of a more general construction, inspired by the indirect sum construction, which builds a Boolean function from four Boolean functions as follows:

$$h(x,y) = f(x) + g(y) + (f(x) + f'(x))(g(y) + g'(y)),$$

and which allowed to construct bent and correlation immune functions.

Theorem 5.5.9. *Let f, f' and g be three weightwise perfectly balanced n-variable functions and let g' be any n-variable Boolean function, then:*

$$h(x,y) = f(x) + \prod_{i=1}^{n} x_i + g(y) + (f(x) + f'(x))g'(y),$$

where $x, y \in \mathbb{F}_2^n$, is a weightwise perfectly balanced $2n$-variable function.

Proof.

- If $k = 0$, then $\mathsf{w_H}(x,y) = k$ is equivalent to $x = y = (0,\ldots,0)$ and we have $h(x,y) = f(0,\ldots,0) + g(0,\ldots,0) = 0$.
- If $k \in \{1,\ldots,n-1\}$, then, the set $\{(x,y) \in \mathbb{F}_2^{2n}; \mathsf{w_H}(x,y) = k\}$ equals the disjoint union of the following sets:
  - $\{(0,\ldots,0)\} \times \{y \in \mathbb{F}_2^n; \mathsf{w_H}(y) = k\}$, on which $h(x,y)$ equals $f(0,\ldots,0) + g(y)$ (since $f(0,\ldots,0) + f'(0,\ldots,0) = 0$) and is then balanced;
  - $\{x \in \mathbb{F}_2^n; \mathsf{w_H}(x) = i\} \times \{y\}$, where $1 \le i \le k$ and $\mathsf{w_H}(y) = k - i$, on each of which $h(x,y)$ equals $f(x) + g(y)$ if $g'(y) = 0$ and $f'(x) + g(y)$ if $g'(y) = 1$; in both cases, it is balanced;
- If $k = n$, then the set $\{(x,y) \in \mathbb{F}_2^{2n}; \mathsf{w_H}(x,y) = k\}$ equals the disjoint union of the following sets:
  - $\{((0,\ldots,0),(1,\ldots,1))\} \cup \{((1,\ldots,1),(0,\ldots,0))\}$, on which $h(x,y)$ equals respectively $f(0,\ldots,0) + g(1,\ldots,1) = 1$ (since $f(0,\ldots,0) + f'(0,\ldots,0) = 0$) and $f(1,\ldots,1) + g(0,\ldots,0) + 1 = 0$ (since $f(1,\ldots,1) + f'(1,\ldots,1) = 0$) and is then globally balanced;
  - $\{x \in \mathbb{F}_2^n; \mathsf{w_H}(x) = i\} \times \{y\}$, where $1 \le i \le n-1$ and $\mathsf{w_H}(y) = n - i$, on each of which $h(x,y)$ equals $f(x) + g(y)$ if $g'(y) = 0$ and $f'(x) + g(y)$ if $g'(y) = 1$; in both cases, it is balanced;
- If $k \in \{n+1,\ldots,2n-1\}$, then the set $\{(x,y) \in \mathbb{F}_2^{2n}; \mathsf{w_H}(x,y) = k\}$ equals the disjoint union of the following sets:
  - $\{(1,\ldots,1)\} \times \{y \in \mathbb{F}_2^n; \mathsf{w_H}(y) = k-n\}$, on which $h(x,y)$ equals $f(1,\ldots,1) + g(y) + 1$ and is then balanced;
  - $\{x \in \mathbb{F}_2^n; \mathsf{w_H}(x) = i\} \times \{y\}$, where $k - n + 1 \le i \le n-1$ and $\mathsf{w_H}(y) = k - i$, on each of which $h(x,y)$ equals $f(x) + g(y)$ if $g'(y) = 0$ and $f'(x) + g(y)$ if $g'(y) = 1$; in both cases, it is balanced;

- If $k = 2n$, then $\mathsf{w_H}(x, y) = k$ is equivalent to $x = y = (1, \ldots, 1)$ and we have $h(x, y) = 1 + 1 + 1 = 1$.

□

Note that for $f = f'$ or $g' = 0$, we obtain the construction related to the direct sum mentioned above. Noting that $f(x_1, x_2) = x_1$ is weightwise perfectly balanced, we can recursively build weightwise perfectly balanced Boolean functions of 2^ℓ variables, for all ℓ in $\mathbb{N}^*$. For instance, applying the construction with $f = f'$, we get:

$$f(x_1, x_2, \ldots, x_{2^\ell}) = \sum_{a=1}^{\ell} \sum_{i=1}^{2^{\ell-a}} \prod_{j=0}^{2^{a-1}-1} x_{i+j2^{\ell-a+1}}.$$

And since g' can be freely chosen and f' can be a version of f in which the coordinates of x are permuted, we have a large number of weightwise perfectly balanced functions by applying Theorem 5.5.9.

We can extend the previous example to get weightwise almost perfectly balanced Boolean function on n variables for all n.

Proposition 5.5.10. *The function f_n in $n \geq 2$ variables, recursively defined by $f_2(x_1, x_2) = x_1$ and for $n \geq 3$:*

$$f_n(x_1, \ldots, x_n) = \begin{cases} f_{n-1}(x_1, \ldots, x_{n-1}) & \text{if } n \text{ odd,} \\ f_{n-1}(x_1, \ldots, x_{n-1}) + x_{n-2} + \prod_{i=1}^{2^{d-1}} x_{n-i} & \text{if } n = 2^d; d > 1, \\ f_{n-1}(x_1, \ldots, x_{n-1}) + x_{n-2} + \prod_{i=1}^{2^d} x_{n-i} & \text{otherwise } n = p \cdot 2^d; p \text{ odd.} \end{cases}$$

is a weightwise almost perfectly balanced Boolean function of degree 2^{d-1}, where $2^d \leq n < 2^{d+1}$, and with $n - 1$ monomials in its ANF if n is even and $n - 2$ monomials if n is odd. Note that this function can be written as a direct sum for all $n \geq 2$.

Proof. The degree and number of monomials of f_n are easily checked by induction on n for $n \geq 2$. We prove the weightwise almost perfect balance property by induction on n as well:

The initialization step is $n = 2$, in this case $f_2 = x_1$ is *WPB*. We now assume that $n \geq 3$ and that, for every $2 \leq i \leq n - 1$, f_i is *WAPB*. We prove under this induction hypothesis that f_n is *WAPB*, using a disjunction of cases depending when n is odd, a power of 2 or an even number not power of 2.

- for n odd:
  - if $k = 0$, then $\mathsf{w_H}(f_n)_0 = \mathsf{w_H}(f_{n-1})_0 = 0$;
  - if $k \in [1, n - 1]$, then:

    $$\mathsf{w_H}(f_n)_k = \mathsf{w_H}(f_{n-1})_k + \mathsf{w_H}(f_{n-1})_{k-1}.$$

    As $n - 1$ is even, at least one of the coefficients $\binom{n-1}{k}, \binom{n-1}{k-1}$ is even. As $n - 1$ is even and k or $k-1$ is odd therefore one of those written in binary has a digit equal to 1 where the corresponding one of n is 0 which characterize the even parity of this binomial coefficient, more precisely $\binom{n}{k}$ has the same parity as $\binom{n-1}{k-(k \bmod 2)}$. Therefore, it gives two cases. If both are even:

    $$\mathsf{w_H}(f_{n-1})_k + \mathsf{w_H}(f_{n-1})_{k-1} = \frac{\binom{n-1}{k} + \binom{n-1}{k-1}}{2} = \frac{\binom{n}{k}}{2},$$

otherwise:

$$\mathsf{w_H}(f_{n-1})_k + \mathsf{w_H}(f_{n-1})_{k-1} = \frac{\binom{n-1}{k} + \binom{n-1}{k-1} \pm 1}{2} = \frac{\binom{n}{k} \pm 1}{2}.$$

– if $k = n$, then $\mathsf{w_H}(f_n)_n = \mathsf{w_H}(f_{n-1})_{n-1} = 1$

Hence, f_n is *WAPB*.

- for $n = 2^d; d > 1$, we can view f_n as the following direct sum:

$$f_n(x_1, \ldots, x_n) =$$

$$f_{2^{d-1}}(x_1, \cdots, x_{2^{d-1}-1}, x_n) + f_{2^{d-1}}(x_{2^{d-1}}, \cdots, x_{n-1}) + \prod_{i=1}^{2^{d-1}} x_{n-i}.$$

As $f_{2^{d-1}}$ is *WPB* by hypothesis, we can apply Theorem 5.5.9 with $g' = 0$, giving that f_n is *WPB*.

- $n = p \cdot 2^d; 1 < p$ odd ; we decompose f_n in a direct sum and use techniques of Theorem 5.5.9's proof:

$$f_n(x_1, \ldots, x_n) =$$

$$f(x_1, \cdots, x_{n-2^d-1}, x_n) + g(x_{n-2^d}, \cdots, x_{n-1}) + \prod_{i=1}^{2^d} x_{n-i}$$

reordering the variables we get $f = f_{n-2^d}$ and $g = f_{2^d}$, with f_{2^d} *WPB* and f_{n-2^d} *WAPB* by the induction hypothesis. f_n being a direct sum of f and $g + \prod_{i=1}^{2^d} x_{n-i}$ we get:

– if $k = 0$: $\mathsf{w_H}(f_n)_0 = \mathsf{w_H}(f)_0\overline{\mathsf{w_H}(g)_0} + \overline{\mathsf{w_H}(f)_0}\mathsf{w_H}(g)_0 = 0$

– if $k \in [1, 2^d - 1]$:

$$\mathsf{w_H}(f_n)_k = \sum_{i=0}^{k} \mathsf{w_H}(g)_i\overline{\mathsf{w_H}(f)_{k-i}} + \overline{\mathsf{w_H}(g)_i}\mathsf{w_H}(f)_{k-i}, \tag{5.6}$$

$$= \mathsf{w_H}(f)_k + \sum_{i=1}^{k} \mathsf{w_H}(g)_i(\overline{\mathsf{w_H}(f)_{k-i}} + \mathsf{w_H}(f)_{k-i}), \tag{5.7}$$

$$= \mathsf{w_H}(f)_k + \frac{1}{2}\sum_{i=1}^{k} \binom{2^d}{i}\binom{n-2^d}{k-i}, \tag{5.8}$$

$$= \mathsf{w_H}(f)_k + \frac{1}{2}\left(\binom{n}{k} - \binom{n-2^d}{k}\right). \tag{5.9}$$

Equation (5.7) comes from g being *WPB* of 2^d variables, therefore for $i \in [1, 2^d - 1]$: $\overline{\mathsf{w_H}(g)_i} = \mathsf{w_H}(g)_i$. Equation (5.8) is obtained using that $\mathsf{w_H}(f)_{k-i} + \overline{\mathsf{w_H}(f)_{k-i}} = \binom{n-2^d}{k-i}$ by definition and $\mathsf{w_H}(g)_i = \frac{1}{2}\binom{2^d}{i}$ because f is a *WPB* function. Equation (5.9) is obtained using Vandermonde convolution: $\sum_{i=0}^{k}\binom{n}{i}\binom{m}{k-i} = \binom{n+m}{k}$.

Therefore $\mathsf{w_H}(f_n)_k = \frac{1}{2}\binom{n}{k}$ if $\binom{n-2^d}{k}$ is even and $\mathsf{w_H}(f_n)_k = \frac{1}{2}\left(\binom{n}{k} \pm 1\right)$ otherwise.

– if $k \in [2^d, n-1]$:

$$\begin{aligned}
\mathsf{w_H}(f_n)_k &= \sum_{i=1}^{2^d-1} \mathsf{w_H}(g)_i\overline{\mathsf{w_H}(f)_{k-i}} + \overline{\mathsf{w_H}(g)_i}\mathsf{w_H}(f)_{k-i} \\
&\quad + \mathsf{w_H}(g)_0\overline{\mathsf{w_H}(f)_k} + \overline{\mathsf{w_H}(g)_0}\mathsf{w_H}(f)_k \\
&\quad + \overline{\mathsf{w_H}(g)_{2^d}\mathsf{w_H}(f)_{k-2^d}} + \mathsf{w_H}(g)_{2^d}\mathsf{w_H}(f)_{k-2^d} \\
&= \sum_{i=1}^{2^d-1} \frac{1}{2}\binom{2^d}{i}\binom{n-2^d}{k-i} + \mathsf{w_H}(f)_k + \mathsf{w_H}(f)_{k-2^d} \\
&= \frac{1}{2}\left(\binom{n}{k} - \binom{n-2^d}{k} - \binom{n-2^d}{k-2^d}\right) + \mathsf{w_H}(f)_k + \mathsf{w_H}(f)_{k-2^d}
\end{aligned}$$

As $n - 2^d \equiv 0[2^{d+1}]$ at least one of $\binom{n-2^d}{k}, \binom{n-2^d}{k-2^d}$ is even therefore $\mathsf{w_H}(f_n)_k = \frac{1}{2}\binom{n}{k}$ if both are even and $\mathsf{w_H}(f_n)_k = \frac{1}{2}\left(\binom{n}{k} \pm 1\right)$ otherwise.

– if $k = n$: $\mathsf{w_H}(f_n)_n = \mathsf{w_H}(f)_{n-2^d}\mathsf{w_H}(g)_{2^d} + \overline{\mathsf{w_H}(f)_{n-2^d}\mathsf{w_H}(g)_{2^d}} = 1$ Giving that f_n is *WAPB*.

To conclude for all $n \geq 2$, f_n is weightwise (almost) perfectly balanced.

□

These highlighted weightwise (almost) perfectly balanced functions are very structured, obtainable from direct sum constructions and with an ANF containing a low number of monomials. It finishes our study on the restricted balancedness, and also on the new criteria on Boolean functions presented in this thesis.

Chapter 6

Conclusion and Perspectives

In this part we conclude on the main topic addressed in this thesis, working towards practical fully homomorphic frameworks for outsourcing computations. We summarize the contributions of the previous chapters in this direction, and finally we give more perspectives on the presented material, emphasizing on the connections with future works and open questions.

Contents

6.1 Conclusion

In this thesis, we presented a solution towards efficient fully homomorphic encryption frameworks for outsourcing computations. Beyond the mathematical beauty of fully homomorphic encryption, a deeper study is necessary to materialize the applications it was dreamed for. The existence of fully homomorphic encryption is due to lattice-based cryptography, in this thesis we showed that the realization of one of these applications, outsourcing computation, becomes realistic with the joint use of constructions from other domains of cryptography.

In Chapter 3, we presented a hybrid framework designed for outsourcing computations, enabling to transfer the cost in time and data of the homomorphic executions from the client to the Cloud. The technical study of the homomorphic properties in practice permitted to identify adapted functions in order to minimize the delay which is independent of the outsourced computations.

Then, in Chapter 4, we introduced the Filter Permutator, a family of symmetric encryption schemes which is designed to be combined with homomorphic encryption schemes in hybrid frameworks. We examined its behavior relatively to a particular error-growth metric, with a close to optimal achievement in theory, and good results in practice.

Finally, in Chapter 5, we investigated the relevant Boolean criteria to assess the security of the symmetric primitive, and therefore of the whole framework. It required to examine the parameters of low-cost functions, and to identify and study new Boolean cryptographic criteria.

This material provides a blueprint for efficient homomorphic hybrid frameworks for outsourcing computations almost optimal in theory, and giving room for practical improvements. It also contributes to a better understanding of some cryptographic constructions, as the study of the error-growth of homomorphic operations, the consideration of transciphering between symmetric and homomorphic encryptions, and the investigation of Boolean functions in a specific context.

6.2 Perspectives

In the following part we introduce a few selected topics related to the material developed in this thesis, that could lead to further studies. First, we describe Goldreich's PRG [Gol00], and show how this primitive can be related to the Filter Permutator. Then, we investigate modifications of the Filter Permutator which could lead to more efficient instantiations than FLIP. We also consider how the cryptanalysis method based on fixed Hamming weight input applying on the Filter Permutator can be extended to other primitives. Finally, we emphasis some open questions on Boolean functions arising from our study.

6.2.1 Goldreich's PRG

More known in the complexity theory community, the work of Oded Goldreich on pseudorandom generators with low complexity has been recently (Spring 2017) pointed out in the cryptographic community, as part of a candidate construction for indistinguishability obfuscation. In 2000, Oded Goldreich introduced a candidate one-way function based on expander graphs [Gol00], which is very easy to evaluate, as designed for practical concerns. More particularly, with appropriate choices of the parameters this one-way function can be in the complexity class NC_1 and even NC_0, giving birth to the pseudorandom local generators,

PRG for which any output depends on a number of inputs upper bounded by a constant. For a more general view on pseudorandom local generators and more generally on pseudorandom local functions (PRF for which each output bit depends on a limited number of inputs) we refer to the survey of Benny Applebaum [App13]. Local pseudorandom generators can be used to construct indistinguishability obfuscation, where the locality of the generator determines the degree of the multi-linear map (studied as a cryptographic primitive in [BS02]) necessary to the construction. Indistinguishability obfuscation is a primitive which received a lot of attention in the last years, as it has been proved that this primitive implies most of the cryptographic primitives, as depicted in [Bar16]. For the constructions of indistinguishability obfuscation from d-linear maps and the frenetic publications on Eprint on this topic we refer to [LT17].

In the following we begin by describing Goldreich's PRG, then we show the connection with the Filter Permutator, finally we consider how this connection could be interesting relatively to these two primitives.

6.2.1.1 Generalities on Goldreich's PRG

Definition 6.2.1 (Goldreich's One Way Function (adapted from [Gol00]))**.** *Let*

- $\{f_n : \{0,1\}^n \to \{0,1\}^m\}_{n\in\mathbb{N}}$ *be a (uniform) collection of functions where* $m = n^s$,
- $S_1, \cdots, S_m \subset [n]$ *be a collection of subsets of size* d,
- $P : \{0,1\}^d \to \{0,1\}$ *be a predicate.*

For all $x \in \{0,1\}^n$ *and* $S \subset [n]$*, where* $S = \{i_1, \cdots, i_d\}$ *and* $i_j < i_{j+1}$ *we denote by* x_S *the projection of* x *on* S *that is* $x_{i_1} \cdots x_{i_d}$*. Fixing* P *and* $S_1, \cdots S_m$ *we define Goldreich's One Way Function as:*

$$f_n(x) = (P(x_{S_1}), P(x_{S_2}), \cdots, P(x_{S_m})) \in \mathbb{F}_2^m.$$

Note that in [Gol00] the value s is fixed to 1 (*i.e.* $m = n$) and the function is conjectured non invertible within time $2^{n/\mathcal{O}(1)}$ (with more concerns on P and the collection of subsets). Benny Applebaum [App12] proved that these random local functions are weak PRG where the distinguishing advantage is upper bounded by an arbitrary fixed inverse polynomial in n (called PPRG) if they are one-way; justifying the denomination of Goldreich's PRG in this section. More precisely, from [AL16], for any polynomial stretch ($s > 1$) the family f_n is conjectured to be a PPRG if:

- the hypergraph obtained by considering as nodes the n coefficients of x and as hyperedges of arity d each subset S_i is a good expander,
- the predicate P has resiliency at least $2s$ and algebraic immunity at least s.

This statement concerns in fact $1 - o(1)$ of such predicates and hypergraphs, for the exact result we refer to [AL16]. For our perspective, it is interesting to know that such kind of pseudorandom generator exists and reductions to other cryptographic primitives are known for it, together with studied conjectures. For example the precedent statement has been proven for particular classes of adversaries: $\mathbb{F}_2$ linear tests and the Lasserre/Parrilo sum-of-squares hierarchy, and conjectured for any polynomial time adversary. These results can be of particular interest for the Filter Permutator model.

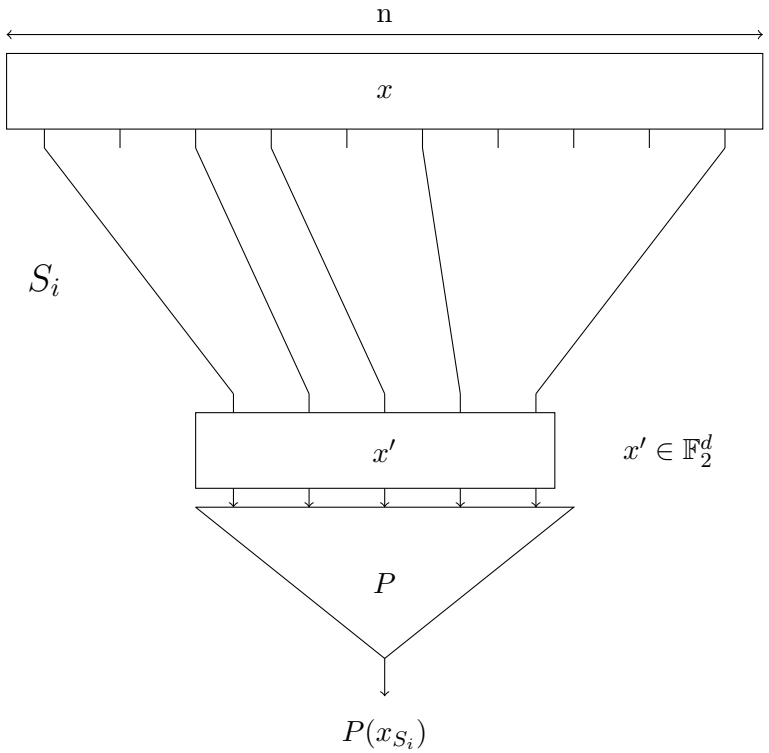

Figure 6.1: Goldreich's PRG output generation.

6.2.1.2 Goldreich's PRG and Filter Permutator

To show the connection between Goldreich's PRG and the Filter Permutator, let us first depict in Figure 6.1 how a bit of output of Goldreich's PRG is computed.

The similitudes between Figure 6.1 and Figure 4.1 permit to show a connection between the two primitives they represent. For a fixed n and a fixed seed $x \in \mathbb{F}_2^n$ for each of the m outputs this instantiation of Goldreich's PRG takes the d coefficients of x indexed by the randomly chosen subset S_i and apply the d variables predicate P. First, note that the predicate P is a Boolean function defined over d variables, that can also be defined as the direct sum of a Boolean function in d variables and the null function in $n - d$ variables. Then, note that XORing the output of a PRG to a plaintext (of smaller or equal length) is equivalent to a stream cipher where the secret key is the PRG seed and whose security relies on the one of the PRG. Finally, note that choosing at random m subsets of $[n]$ of d elements in the natural order ($i_j < i_{j+1}$) is equivalent to choosing at random m permutations of $[n]$ where the first d elements are in the natural order.

Summing up these observations, it gives only two differences between Goldreich's PRG and the Filter Permutator construction (without considering the security requirements of these different primitives). The first one is the size of the input of the predicate (respectively the filtering function), which is smaller than the seed length (respectively key length) for Goldreich's PRG, giving the locality of the PRG and more particularly the membership in NC_0 for d independent of n. The second difference is the order in the subsets, for all S_i the elements are in natural order whereas the permutations of the Filter Permutator do not

guarantee any order. Thus, at the cost of minor modifications, instantiations of Goldreich PRG and instantiations of the Filter Permutator are equivalent, motivating further researches on how the knowledge of one of these primitives can benefit to the other.

A potential research direction is to reduce the security of the Filter Permutator to more investigated hypotheses as the one developed to analyze Goldreich PRG and more generally random local functions. These functions have been studied extensively in the complexity theory community and can be used to construct a variety of cryptographic primitives (as surveyed in [App13]) as public key encryption [ABW10]. All these results are asymptotic, then they could ensure notion of security for families of instantiations of the Filter Permutator rather than concrete security as required for deployed symmetric encryption schemes. Nevertheless, it could also have applications on the symmetric encryption scheme for a fixed security parameter as the security of Goldreich's PRG is conjectured to depend on only two Boolean criteria, the resiliency and the algebraic immunity. Such conjecture could lead to consider instantiations of the filter Permutator similar to FLIP without the quadratic part and exempted of considerations of nonlinearity.

Another interesting research direction is the cryptanalysis of Goldreich's PRG or of the Filter Permutator. The resiliency and algebraic immunity of the 4 proposed candidates of FLIP have parameters adequate with conjectured secure instances of Goldreich's PRG, but as they are not part of a family indexed by $n \in \mathbb{N}$ it does not lead to more conclusions. Simpler candidates can be broken for a fixed security (as shown in [DLR16b]), with parameter still adequate with conjectured secure families of functions, then it could be interesting to examine if these attacks can be extended to the asymptotic constructions. Then, in the complexity theory community, the algebraic attacks have been rediscovered in 2016 in [AL16] where before the Boolean function:

$$\sum_{i=1}^{2s} x_i + \prod_{i=2s+1}^{3s} x_i,$$

where s is the polynomial stretch, was conjectured to be a good predicate for Goldreich's PRG.

Note that this function has algebraic immunity 2 independently of s, as defined in 2003 ([CM03]), reducing the security to solving an algebraic system of degree at most 2 of n^s equations in n variables, totally breaking the pseudorandomness for all stretch $s \geq 2$. As the only criterion considered before were the algebraic degree and the resiliency, it seems interesting to study which standard symmetric cryptanalysis could apply to Goldreich's PRG, and why some cannot apply for constructions defined for all n. This study could also determine minimal values of n for which concrete instantiations of Goldreich's PRG could not be distinguished with good probability with particular attacks bounded by 2^λ computations. Indeed, up to our knowledge, it seems difficult to find parameters (values of n) for a concrete instantiation of Goldreich's PRG for polynomial stretch relatively to a fixed security level λ, whereas it could be needed for instantiating indistinguishability obfuscation candidates. Finally, one important particularity of the Filter Permutator beyond the symmetric encryption schemes is the invariance of the Hamming weight of its input during the encryption. For Goldreich's PRG the seed does not variate neither but each output bit depends on a few bits of the seed, then using the notation used in Figure 6.1, we have $\mathsf{w_H}(x) = k$ and $0 \leq \mathsf{w_H}(x') \leq \min(d, k)$. Then the distribution of $\mathsf{w_H}(x')$ is not uniform, as it depends on k and d, and it could be interesting to investigate if this particularity implies a usable weakness or not. Due to the particular relations between n and d the distribution

of $\mathsf{w_H}(x')$ could be more or less statistically close to the uniform distribution over binary string of length d. Then the choice of the predicate could also play a role in the security, depending on how it amplifies or reduces the impact of this particularity. As this concept seems orthogonal to the standard criteria of resiliency and algebraic immunity, it could lead to further studies. This concludes our perspective on Goldreich's PRG from the contributions of this thesis.

6.2.2 Improved Filter Permutator

In this thesis we presented the version of the Filter Permutator as presented in [MJSC16], since then the symmetric security of this primitive is more understood, enabling to contemplate new instantiations for particular applications. We briefly describe here 3 potential orientations of the initial design deserving further investigations.

In the presented instantiation FLIP, most of the security analysis concerns guess-and-determine attacks and fixed-Hamming-weight cryptanalysis, requiring to consider a filtering function with good parameters relatively to many criteria. In particular for FLIP filtering functions it leads to consider functions built from various triangular functions, increasing the number of variables, so decreasing the efficiency of the scheme. Two modifications of the Filter Permutator permit to reduce the impact of these attacks: fixing the input size of F smaller than the key size (as in Goldreich's PRG) and adding a public whitening on F input at each clock cycle. Both of these techniques break the invariant of the Hamming weight on F input, the first modification gives more variation on the Hamming weight of the input of F, depending on the Hamming weight of the key and the input size of F. The second modification makes the input of F uniform, making more complex to consider an attack based on the parameters of F on restricted subsets as we did for FLIP in Section 4.5.3. Then, the first modification diminishes the impact of guess-and-determine attacks as the guessed variables are not always taken in F inputs, so with the same function as in the instances of Section 4.3.2.2 and a larger key register the data cost of the attacks with guess-and-determine increases. The second modification makes more difficult to obtain samples where a particular function F' in fewer variables is obtained, as the guesses on the key bits have to match with the whitening, also increasing the data complexity of such attacks.

Note that both of these modifications do not have a negative impact when the Filter Permutator is used in a hybrid homomorphic framework: the increasing of the key register do not imply to compute more complex functions on the homomorphic ciphertexts and the whitening can be performed by adding zero-noise ciphertext or using hybrid ciphertexts as explained in Section 3.4. Then using these modifications together with the same analysis of error-growth for the third generation FHE and a similar security analysis centered on the standard Boolean criteria could lead to use simpler FLIP functions and therefore more efficient FHE frameworks for outsourcing computation.

The third generation of FHE has been shown to be compatible with branching programs, as illustrated by the works [BV14; CGGI16], it could be interesting to design an instantiation of the Filter Permutator in this direction. The instantiation FLIP is oriented to produce low-noise ciphertexts when they are computed with AND and XOR gates, the Filter Permutator could be oriented to produce low-noise ciphertexts when its decryption circuit is evaluated as a branching program, or a close variation. For the third generation, the use of MUX gates to evaluate a function can influence the low-noise quality of a function, as proved in Section 3.3.3.2. Then it could lead to study other Boolean functions than direct sums of

monomials in order to obtain secure instantiations as symmetric encryption scheme and low-noise branching programs. The particular shape of functions corresponding to low-noise branching programs could be quite different from the well-studied families of Boolean functions or from the functions with bounded number of monomials potentially requiring a new study as Section 5.1.

Finally, an orientation of the Filter Permutator for bigger fields could be an attractive research direction. As shown in [AGR+16], symmetric cryptographic primitives with low multiplicative complexity can be useful in context various as MPC, FHE and zero-knowledge proofs, not only for primitives working on $\mathbb{F}_2$ but also on bigger fields. As a first approach, we could consider the relations between current instantiations of FLIP (on $\mathbb{F}_2$) and similar instantiations on bigger rings or fields R. Considering a secret key of elements in R and FLIP functions defined from R^N to R, we can consider the system of equations given by the keystream function of this version of the Filter Permutator. The FLIP functions being Boolean functions, their sums should be interpreted as XOR for other moduli using that $x_1 \oplus x_2 = x_1 + x_2 - 2x_1x_2$ for these cases, and the considered system should be complemented with the ring or fields equations. Considering such systems some reductions of security of FLIP versions from a structure to another could be proven. As a trivial example, a decryption oracle for FLIP defined on $\mathbb{Z}/2\ell\mathbb{Z}$ would be sufficient to solve the system of FLIP defined over the Boolean field. In this case, it could lead to a non-binary symmetric encryption scheme with security at least the one of the current instantiations. A better (but way more intricate) approach would consist in determining which are the important cryptographic criteria for a filtering function defined over a particular ring or field, and investigate which functions have good parameters for these criteria and are sufficiently low cost for the considered applications.

These 3 potential orientations show that is still large room for improvements on the Filter Permutator, both on a practical side: find more efficient instantiations, and on a theoretical side: find the minimal functions providing security.

6.2.3 Weightwise Cryptanalysis

In Section 4.5.3 we presented a security analysis of the FLIP ciphers based on the behavior of a function on restricted sets, more particularly on the sets of all vectors of a fixed Hamming weight. Up to our knowledge, this *Weightwise Cryptanalysis* has not been considered before [CMR17] and could be applied to other ciphers. Note that on the preprint version of this work, this cryptanalysis is explained in a context of side channel (a strong model where at multiple times the exact Hamming weight of a register is leaked), and as a proof of concept described on the stream-cipher Grain. The restriction to a constant and known Hamming weight is not common in cryptographic constructions, the leakage of the exact Hamming weight of critical values of a scheme is not common neither, nevertheless these considerations can lead to two interesting topics.

First, in a constrained context where functions are forced to be evaluated on inputs with bounded-Hamming-weight, which security could be expected? And consequently, could we obtain constructions with better guaranties of security dealing with functions restricted to bounded Hamming weight inputs rather than functions considered on their whole input space? Note that the Hamming weight is a common measure in the side-channel area, and that restricting the Hamming weight has a non negligible influence on an electronic device perspective, potentially motivating the study of good cryptographic functions on inputs of restricted Hamming weight.

Second, for some constructions the Hamming weight could be determined, exactly or approximatively and then a weaker weightwise cryptanalysis could be applied. More precisely for ciphers using LFSR or NFSR with the Fibonacci representation the Hamming weight is not variating a lot when few updates are performed, therefore in a chosen plaintext attack some information on the Hamming weight of the registers at particular times could be extracted. This weight would be in a particular range of possible Hamming weights, if this range defines a subset where one of the applied functions has a cryptographic flaw it could be used to attack the whole scheme. Regarding LFSR, various results are known on the sequence they generate, as its period. In the context we consider, results on the evolution of the Hamming weight of consecutive elements of this sequence could be sufficient to apply a weightwise cryptanalysis. Indeed, it could be sufficient to determine particular times where the input of a function has Hamming weight restricted to a particular range of values and then use the analysis of the Boolean function on these sets only. Therefore weightwise cryptanalysis could lead to interesting studies relatively to LFSR or NFSR sequences, finishing our thoughts on this perspective.

6.2.4 New Considerations on Boolean Functions

The study of Chapter 5 presents various results on Boolean functions whose motivation comes from the particularities of the Filter Permutator. Beyond this motivation, extending this study could benefit to other works. First, the investigation on the simplest functions (relatively to a specific metric) providing security is suitable for the design of efficient cryptographic primitives. Then, the generalization of Boolean cryptographic criteria to particular restrictions of the function's input (recurrent and restricted criteria) could have various applications in cryptanalysis. Finally, the better understanding of mathematical objects as Boolean functions, Codes, Lattices, *etc* has many applications, and above all, it is a goal in itself. Consequently we lay emphasis in the following on some of the open problems arising from our work on Boolean functions.

6.2.4.1 Open Questions on Low Cost Boolean Functions

In Section 5.1 we presented results on functions with an extremely low number of monomials and good parameters relatively to standard cryptographic criteria, nevertheless some questions are still open for these simple constructions.

The first one arises from the recurrent high-order nonlinearity of direct sums of monomials (Section 5.2.2):

Question 6.1. *For all Boolean functions g and h, and all integers d such that $d > 1$ what is the exact expression of the nonlinearity of order d of their direct sum $f = g + h$ in terms of $\mathsf{NL_d}(g)$ and $\mathsf{NL_d}(h)$?*

Many questions remain open on the high-order nonlinearity criterion, for the applications we considered to provide nonlinearity we used a Dickson function in $2n$ variables as it is a bent function with only n monomials in its ANF. Optimizing the nonlinearity relatively to the number of monomials leads to this question:

Question 6.2. *For all strictly positive integers n and d, what is the minimal number of monomials of an n-variable Boolean function to reach the optimal nonlinearity of order d?*

For the algebraic immunity criterion, a similar question is still open:

Question 6.3. *What is the minimal number of monomials of an n-variable Boolean function to reach the optimal algebraic immunity?*

Remark 5.1.6 emphases on the equality of the algebraic immunity and the number of monomials of the triangular functions, and on the optimality of this ratio, but it does not answer to the precedent question for $n > 2$. More investigations could be conducted relatively to this ratio, denoting $M(f)$ the number of monomials in the ANF of the function f:

Question 6.4. *For a strictly positive integer c, what are the families of functions (indexed by n) such that the algebraic immunity of f is (less than or) equal to $M(f)/c$?*

6.2.4.2 Open Questions on Restricted Criteria

In Sections 5.3, 5.4 and 5.5, we considered Boolean functions on restricted input, which is quite a new cryptographic point of view. Our study focusing on fixed Hamming weight input tries to address most of the natural questions in this context, we highlight here some other questions of various interest and presumed difficulty which are not answered yet.

Following on constructions with few monomials, in Section 5.5.2 we exhibited a *WPB* function with ANF containing $n-1$ monomials whereas we proved a lower bound of $3n/4+1$ (for $n > 4$), raising the following interrogation:

Question 6.5. *For any integer ℓ, $\ell > 3$, what is the minimal number of monomials of a WPB function in 2^ℓ variables?*

The optimal restricted nonlinearity is examined in Section 5.4.1 where various upper bounds are considered in the general case and latter for the sets $E_{n,k}$. It leads to the wide question:

Question 6.6. *For a particular integer n and a particular set $E \in \mathbb{F}_2^n$, what is the maximal possible value of NL_E? And, generalizing to a family of sets $\mathcal{E} = \{E_n\}_{n \in I \subseteq \mathbb{N}}$ where for each $n \in I$ a subset of $\mathbb{F}_2^n$ is associated, what are the families of functions (indexed by n) reaching this optimal restricted nonlinearity for all sets of $\mathcal{E}$?*

Note that it is equivalent to determine the covering radius of all punctured Reed-Muller codes of order 1 and exhibit families of vectors reaching this value. This code analogy also brings back the simpler question of constructing functions with optimal weightwise nonlinearity for a particular Hamming weight using the recursive structure of Reed-Muller codes.

Some questions naturally arise from our study of Section 5.3.2 on the restricted algebraic immunity, more specifically regarding the sets $E_{n,k}$. We proved that on these sets the restricted algebraic immunity e is bounded (Corollary 5.3.9) by the relation:

$$2\binom{n}{e} > \binom{n}{k}.$$

Leading to the following issue:

Question 6.7. *For all integers n and k, what is the smallest integer e satisfying this relation? And what is the asymptotic behavior of e relatively to the standard algebraic immunity upper bound $\lceil n/2 \rceil$?*

We linked the algebraic immunity upper bound to the rank of the generator matrix of a punctured Reed-Muller code $RM(d,n)_{E_{n,k}}$. This matrix can also be used to compute the exact $AI_{E_{n,k}}$ of a given function, by partitioning the columns depending on the value of f on the column entry and determining when the rank of one of these two matrices is strictly inferior to the rank of the global one. For matrices with rank r at least twice the number of columns, proving the existence of a partition of the columns in two matrices with rank r will prove the tightness of the $AI_{E_{n,k}}$ upper bound e, leading to the following question:

Question 6.8. *For all sets $E_{n,k}$, is the upper bound of Corollary 5.3.9 tight?*

If this bound is tight, it would be interesting to find families of functions reaching it. It cannot be the case of the majority functions, with optimal standard algebraic immunity, as being symmetric functions they are constant when the input Hamming weight is fixed. Therefore optimal functions for restricted weight algebraic immunity could lead to very different constructions. The same reasoning applies for the nonlinearity for which we proved that bent functions could have trivial weightwise nonlinearity (Section 5.4.3); and for the balancedness as the lower bound on the degree of a *WPB* function (see Section 5.5.1) implies an upper bound on its resiliency. These examples of degradation of the optimality between a standard criterion and its weightwise variant leads to the following question:

Question 6.9. *Can we find families of functions (indexed by n) which are both optimal for a cryptographic criterion on $\mathbb{F}_2^n$ and for its restricted variant on the family of sets $E_{n,k}$?*

All these questions witness the potential variety of research direction concerning the restricted Boolean criteria. They also enable to envisage other families of sets of $\mathbb{F}_2^n$ neither vectorspaces neither $E_{n,k}$ with particular cryptanalysis applications and new constructions of Boolean functions, potentially leading to plenty of further studies.

Bibliography

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. "Public-key cryptography from different assumptions". In: *42nd Annual ACM Symposium on Theory of Computing*. Ed. by Leonard J. Schulman. Cambridge, MA, USA: ACM Press, June 2010, pp. 171–180 (cit. on p. 153).

[ACG+06] Frederik Armknecht, Claude Carlet, Philippe Gaborit, Simon Künzli, Willi Meier, and Olivier Ruatta. "Efficient Computation of Algebraic Immunity for Algebraic and Fast Algebraic Attacks". In: *Advances in Cryptology – EUROCRYPT 2006*. Ed. by Serge Vaudenay. Vol. 4004. Lecture Notes in Computer Science. St. Petersburg, Russia: Springer, Heidelberg, Germany, May 2006, pp. 147–164 (cit. on p. 79).

[AG11] Sanjeev Arora and Rong Ge. "New Algorithms for Learning in Presence of Errors". In: *ICALP 2011: 38th International Colloquium on Automata, Languages and Programming, Part I*. Ed. by Luca Aceto, Monika Henzinger, and Jiri Sgall. Vol. 6755. Lecture Notes in Computer Science. Zurich, Switzerland: Springer, Heidelberg, Germany, July 2011, pp. 403–415 (cit. on p. 66).

[AGKS05] Masayuki Abe, Rosario Gennaro, Kaoru Kurosawa, and Victor Shoup. "Tag-KEM/DEM: A New Framework for Hybrid Encryption and A New Analysis of Kurosawa-Desmedt KEM". In: *Advances in Cryptology – EUROCRYPT 2005*. Ed. by Ronald Cramer. Vol. 3494. Lecture Notes in Computer Science. Aarhus, Denmark: Springer, Heidelberg, Germany, May 2005, pp. 128–146 (cit. on p. 60).

[AGR+16] Martin R. Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. "MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity". In: *Advances in Cryptology – ASIACRYPT 2016, Part I*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10031. Lecture Notes in Computer Science. Hanoi, Vietnam: Springer, Heidelberg, Germany, Dec. 2016, pp. 191–219. DOI: 10.1007/978-3-662-53887-6_7 (cit. on p. 155).

[AL16] Benny Applebaum and Shachar Lovett. "Algebraic attacks against random local functions and their countermeasures". In: *48th Annual ACM Symposium on Theory of Computing*. Ed. by Daniel Wichs and Yishay Mansour. Cambridge, MA, USA: ACM Press, June 2016, pp. 1087–1100 (cit. on pp. 151, 153).

[And95] Ross J. Anderson. "Searching for the Optimum Correlation Attack". In: *Fast Software Encryption – FSE'94*. Ed. by Bart Preneel. Vol. 1008. Lecture Notes in Computer Science. Leuven, Belgium: Springer, Heidelberg, Germany, Dec. 1995, pp. 137–143 (cit. on p. 82).

[AP13] Jacob Alperin-Sheriff and Chris Peikert. "Practical Bootstrapping in Quasilinear Time". In: *Advances in Cryptology – CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Santa

Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2013, pp. 1–20. DOI: 10.1007/978-3-642-40041-4_1 (cit. on p. 57).

[AP14] Jacob Alperin-Sheriff and Chris Peikert. "Faster Bootstrapping with Polynomial Error". In: *Advances in Cryptology – CRYPTO 2014, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2014, pp. 297–314. DOI: 10.1007/978-3-662-44371-2_17 (cit. on pp. 23, 41, 42, 57, 67).

[App12] Benny Applebaum. "Pseudorandom generators with long stretch and low locality from random local one-way functions". In: *44th Annual ACM Symposium on Theory of Computing*. Ed. by Howard J. Karloff and Toniann Pitassi. New York, NY, USA: ACM Press, May 2012, pp. 805–816 (cit. on p. 151).

[App13] Benny Applebaum. "Cryptographic Hardness of Random Local Functions-Survey". In: *TCC 2013: 10th Theory of Cryptography Conference*. Ed. by Amit Sahai. Vol. 7785. Lecture Notes in Computer Science. Tokyo, Japan: Springer, Heidelberg, Germany, Mar. 2013, p. 599. DOI: 10.1007/978-3-642-36594-2_33 (cit. on pp. 151, 153).

[APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. *On The Concrete Hardness Of Learning With Errors*. Cryptology ePrint Archive, Report 2015/046. http://eprint.iacr.org/2015/046. 2015 (cit. on p. 57).

[ARS+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. "Ciphers for MPC and FHE". In: *Advances in Cryptology – EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Sofia, Bulgaria: Springer, Heidelberg, Germany, Apr. 2015, pp. 430–454. DOI: 10.1007/978-3-662-46800-5_17 (cit. on pp. 63, 74, 75, 77).

[AZ04] Martin Aigner and Gunter M. Ziegler. *Proofs from THE BOOK (3. ed.)* Springer, 2004. ISBN: 978-3-540-40460-6 (cit. on p. 133).

[Bar16] Boaz Barak. *Hopes, Fears and Software Obfuscation: A Survey*. Cryptology ePrint Archive, Report 2016/210. http://eprint.iacr.org/2016/210. 2016 (cit. on p. 151).

[BC11] Christina Boura and Anne Canteaut. "Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak-f and Hamsi-256". In: *SAC 2010: 17th Annual International Workshop on Selected Areas in Cryptography*. Ed. by Alex Biryukov, Guang Gong, and Douglas R. Stinson. Vol. 6544. Lecture Notes in Computer Science. Waterloo, Ontario, Canada: Springer, Heidelberg, Germany, Aug. 2011, pp. 1–17 (cit. on p. 83).

[BDJR97] Mihir Bellare, Anand Desai, Eric Jokipii, and Phillip Rogaway. "A Concrete Security Treatment of Symmetric Encryption". In: *38th Annual Symposium on Foundations of Computer Science*. Miami Beach, Florida: IEEE Computer Society Press, Oct. 1997, pp. 394–403 (cit. on pp. 16, 60, 78).

[Ben16] Fabrice Benhamouda. "Diverse modules and zero-knowledge". PhD thesis. École Normale Supérieure, Paris, France, 2016. URL: https://tel.archives-ouvertes.fr/tel-01399476 (cit. on p. 10).

[BG07] Côme Berbain and Henri Gilbert. "On the Security of IV Dependent Stream Ciphers". In: *Fast Software Encryption – FSE 2007*. Ed. by Alex Biryukov. Vol. 4593. Lecture Notes in Computer Science. Luxembourg, Luxembourg: Springer, Heidelberg, Germany, Mar. 2007, pp. 254–273 (cit. on p. 78).

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) fully homomorphic encryption without bootstrapping". In: *ITCS 2012: 3rd Innovations in Theoretical Computer Science*. Ed. by Shafi Goldwasser. Cambridge, MA, USA: Association for Computing Machinery, Jan. 2012, pp. 309–325 (cit. on pp. 34–36, 41).

[BKW00] Avrim Blum, Adam Kalai, and Hal Wasserman. "Noise-tolerant learning, the parity problem, and the statistical query model". In: *32nd Annual ACM Symposium on Theory of Computing*. Portland, OR, USA: ACM Press, May 2000, pp. 435–440 (cit. on pp. 81, 88).

[BLMZ16] Fabrice Benhamouda, Tancrède Lepoint, Claire Mathieu, and Hang Zhou. *Optimization of Bootstrapping in Circuits*. Cryptology ePrint Archive, Report 2016/785. http://eprint.iacr.org/2016/785. 2016 (cit. on p. 58).

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. "Classical hardness of learning with errors". In: *45th Annual ACM Symposium on Theory of Computing*. Ed. by Dan Boneh, Tim Roughgarden, and Joan Feigenbaum. Palo Alto, CA, USA: ACM Press, June 2013, pp. 575–584 (cit. on p. 57).

[BM84] Manuel Blum and Silvio Micali. "How to Generate Cryptographically Strong Sequences of Pseudorandom Bits". In: *SIAM Journal on Computing* 13.4 (1984), pp. 850–864 (cit. on p. 70).

[BMT78] Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. "On the inherent intractability of certain coding problems (Corresp.)" In: *IEEE Trans. Information Theory* 24.3 (1978), pp. 384–386. DOI: 10.1109/TIT.1978.1055873. URL: https://doi.org/10.1109/TIT.1978.1055873 (cit. on p. 33).

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. "Pseudorandom Functions and Lattices". In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. Lecture Notes in Computer Science. Cambridge, UK: Springer, Heidelberg, Germany, Apr. 2012, pp. 719–737 (cit. on p. 22).

[Bra12] Zvika Brakerski. "Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP". In: *Advances in Cryptology – CRYPTO 2012*. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2012, pp. 868–886 (cit. on p. 34).

[BS02] Dan Boneh and Alice Silverberg. *Applications of Multilinear Forms to Cryptography*. Cryptology ePrint Archive, Report 2002/080. http://eprint.iacr.org/2002/080. 2002 (cit. on p. 151).

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. "Efficient Fully Homomorphic Encryption from (Standard) LWE". In: *52nd Annual Symposium on Foundations of Computer Science*. Ed. by Rafail Ostrovsky. Palm Springs, CA, USA: IEEE Computer Society Press, Oct. 2011, pp. 97–106 (cit. on pp. 34, 35, 57, 66).

[BV14] Zvika Brakerski and Vinod Vaikuntanathan. "Lattice-based FHE as secure as PKE". In: *ITCS 2014: 5th Innovations in Theoretical Computer Science*. Ed. by Moni Naor. Princeton, NJ, USA: Association for Computing Machinery, Jan. 2014, pp. 1–12 (cit. on pp. 46, 49, 154).

[BY03] Mihir Bellare and Bennet S. Yee. "Forward-Security in Private-Key Cryptography". In: *Topics in Cryptology – CT-RSA 2003*. Ed. by Marc Joye. Vol. 2612. Lecture Notes in Computer Science. San Francisco, CA, USA: Springer, Heidelberg, Germany, Apr. 2003, pp. 1–18 (cit. on p. 71).

[Car10] Claude Carlet. "Boolean Functions for Cryptography and Error-Correcting Codes". In: *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*. Ed. by Yves Crama and Peter L.Editors Hammer. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2010, pp. 257–397. DOI: 10.1017/CBO9780511780448.011 (cit. on pp. 24, 96, 98, 129, 137, 138).

[CCF+16] Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. "Stream Ciphers: A Practical Solution for Efficient Homomorphic-Ciphertext Compression". In: *Fast Software Encryption – FSE 2016*. Ed. by Thomas Peyrin. Vol. 9783. Lecture Notes in Computer Science. Bochum, Germany: Springer, Heidelberg, Germany, Mar. 2016, pp. 313–333. DOI: 10.1007/978-3-662-52993-5_16 (cit. on pp. 59, 64, 65, 68, 74, 75, 77).

[CCK+13] Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrède Lepoint, Mehdi Tibouchi, and Aaram Yun. "Batch Fully Homomorphic Encryption over the Integers". In: *Advances in Cryptology – EUROCRYPT 2013*. Ed. by Thomas Johansson and Phong Q. Nguyen. Vol. 7881. Lecture Notes in Computer Science. Athens, Greece: Springer, Heidelberg, Germany, May 2013, pp. 315–335. DOI: 10.1007/978-3-642-38348-9_20 (cit. on p. 34).

[CF08] Claude Carlet and Keqin Feng. "An Infinite Class of Balanced Functions with Optimal Algebraic Immunity, Good Immunity to Fast Algebraic Attacks and Good Nonlinearity". In: *Advances in Cryptology – ASIACRYPT 2008*. Ed. by Josef Pieprzyk. Vol. 5350. Lecture Notes in Computer Science. Melbourne, Australia: Springer, Heidelberg, Germany, Dec. 2008, pp. 425–440 (cit. on p. 100).

[CFGR12] Claude Carlet, Jean-Charles Faugère, Christopher Goyet, and Guénaël Renault. "Analysis of the algebraic side channel attack". In: *J. Cryptographic Engineering* 2.1 (2012), pp. 45–62. DOI: 10.1007/s13389-012-0028-0. URL: https://doi.org/10.1007/s13389-012-0028-0 (cit. on p. 119).

[CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. "Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds". In: *Advances in Cryptology – ASIACRYPT 2016, Part I*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10031. Lecture Notes in Computer Science. Hanoi, Vietnam: Springer, Heidelberg, Germany, Dec. 2016, pp. 3–33. DOI: 10.1007/978-3-662-53887-6_1 (cit. on pp. 46, 49, 57, 67, 76, 154).

[CGGI17] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. "Improving TFHE: faster packed homomorphic operations and efficient circuit bootstrapping". In: *IACR Cryptology ePrint Archive* 2017 (2017), p. 430. URL: http://eprint.iacr.org/2017/430 (cit. on p. 60).

[CLT14] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. "Scale-Invariant Fully Homomorphic Encryption over the Integers". In: *PKC 2014: 17th International Conference on Theory and Practice of Public Key Cryptography*. Ed. by Hugo Krawczyk. Vol. 8383. Lecture Notes in Computer Science. Buenos Aires, Argentina: Springer, Heidelberg, Germany, Mar. 2014, pp. 311–328. DOI: 10.1007/978-3-642-54631-0_18 (cit. on pp. 34, 63, 74).

[CM03] Nicolas Courtois and Willi Meier. "Algebraic Attacks on Stream Ciphers with Linear Feedback". In: *Advances in Cryptology – EUROCRYPT 2003*. Ed. by Eli Biham. Vol. 2656. Lecture Notes in Computer Science. Warsaw, Poland: Springer, Heidelberg, Germany, May 2003, pp. 345–359 (cit. on pp. 79, 119, 121, 153).

[CM07] Claude Carlet and Sihem Mesnager. "Improving the Upper Bounds on the Covering Radii of Binary Reed-Muller Codes". In: *IEEE Trans. Information Theory* 53.1 (2007), pp. 162–173. DOI: 10.1109/TIT.2006.887494. URL: https://doi.org/10.1109/TIT.2006.887494 (cit. on p. 131).

[CMR17] Claude Carlet, Pierrick Méaux, and Yann Rotella. *Boolean functions with restricted input and their robustness; application to the FLIP cipher*. Cryptology ePrint Archive, Report 2017/097. http://eprint.iacr.org/2017/097. 2017 (cit. on pp. 6, 87, 89, 96, 105, 118, 119, 126, 128, 155).

[CN11] Yuanmi Chen and Phong Q. Nguyen. "BKZ 2.0: Better Lattice Security Estimates". In: *Advances in Cryptology – ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. Lecture Notes in Computer Science. Seoul, South Korea: Springer, Heidelberg, Germany, Dec. 2011, pp. 1–20 (cit. on pp. 57, 75).

[CNT12] Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. "Public Key Compression and Modulus Switching for Fully Homomorphic Encryption over the Integers". In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. Lecture Notes in Computer Science. Cambridge, UK: Springer, Heidelberg, Germany, Apr. 2012, pp. 446–464 (cit. on p. 34).

[Cou03a] Nicolas Courtois. "Fast Algebraic Attacks on Stream Ciphers with Linear Feedback". In: *Advances in Cryptology – CRYPTO 2003*. Ed. by Dan Boneh. Vol. 2729. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2003, pp. 176–194 (cit. on pp. 26, 79).

[Cou03b] Nicolas Courtois. "Higher Order Correlation Attacks, XL Algorithm and Cryptanalysis of Toyocrypt". In: *ICISC 02: 5th International Conference on Information Security and Cryptology*. Ed. by Pil Joong Lee and Chae Hoon Lim. Vol. 2587. Lecture Notes in Computer Science. Seoul, Korea: Springer, Heidelberg, Germany, Nov. 2003, pp. 182–199 (cit. on pp. 82, 86).

[CP08] Christophe De Cannière and Bart Preneel. "Trivium". In: *LNCS, New Stream Cipher Designs - The eSTREAM Finalists* 4986 (2008), pp. 244–266 (cit. on p. 64).

[CS15] Jung Hee Cheon and Damien Stehlé. "Fully Homomophic Encryption over the Integers Revisited". In: *Advances in Cryptology – EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Sofia, Bulgaria: Springer, Heidelberg, Germany, Apr. 2015, pp. 513–536. DOI: 10.1007/978-3-662-46800-5_20 (cit. on p. 34).

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. "Lattice Signatures and Bimodal Gaussians". In: *Advances in Cryptology – CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2013, pp. 40–56. DOI: 10.1007/978-3-642-40041-4_3 (cit. on p. 76).

[DEM16] Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. "Higher-Order Cryptanalysis of LowMC". In: *ICISC 15: 18th International Conference on Information Security and Cryptology*. Ed. by Soonhak Kwon and Aaram Yun. Vol. 9558. Lecture Notes in Computer Science. Seoul, Korea: Springer, Heidelberg, Germany, Nov. 2016, pp. 87–101. DOI: 10.1007/978-3-319-30840-1_6 (cit. on pp. 63, 74, 78).

[DGHV09] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. *Fully Homomorphic Encryption over the Integers*. Cryptology ePrint Archive, Report 2009/616. http://eprint.iacr.org/2009/616. 2009 (cit. on p. 34).

[DGM04] Deepak Kumar Dalai, Kishan Chand Gupta, and Subhamoy Maitra. "Results on Algebraic Immunity for Cryptographically Significant Boolean Functions". In: *Progress in Cryptology - INDOCRYPT 2004: 5th International Conference in Cryptology in India*. Ed. by Anne Canteaut and Kapalee Viswanathan. Vol. 3348. Lecture Notes in Computer Science. Chennai, India: Springer, Heidelberg, Germany, Dec. 2004, pp. 92–106 (cit. on p. 113).

[DK12] Ilya Dumer and Olga Kapralova. "Spherically punctured biorthogonal codes". In: *Proceedings of the 2012 IEEE International Symposium on Information Theory, ISIT 2012, Cambridge, MA, USA, July 1-6, 2012*. 2012, pp. 259–263. DOI: 10.1109/ISIT.2012.6283988. URL: https://doi.org/10.1109/ISIT.2012.6283988 (cit. on p. 135).

[DK13] Ilya Dumer and Olga Kapralova. "Spherically Punctured Biorthogonal Codes". In: *IEEE Trans. Information Theory* 59.9 (2013), pp. 6010–6017. DOI: 10.1109/TIT.2013.2250579. URL: https://doi.org/10.1109/TIT.2013.2250579 (cit. on p. 135).

[DK17] Ilya Dumer and Olga Kapralova. “Spherically Punctured Reed-Muller Codes”. In: *IEEE Trans. Information Theory* 63.5 (2017), pp. 2773–2780. DOI: 10.1109/TIT.2017.2673827. URL: https://doi.org/10.1109/TIT.2017.2673827 (cit. on pp. 126, 135).

[DLMW15] Itai Dinur, Yunwen Liu, Willi Meier, and Qingju Wang. “Optimized Interpolation Attacks on LowMC”. In: *Advances in Cryptology – ASIACRYPT 2015, Part II*. Ed. by Tetsu Iwata and Jung Hee Cheon. Vol. 9453. Lecture Notes in Computer Science. Auckland, New Zealand: Springer, Heidelberg, Germany, Nov. 2015, pp. 535–560. DOI: 10.1007/978-3-662-48800-3_22 (cit. on pp. 63, 74, 78).

[DLR16a] Sébastien Duval, Virginie Lallemand, and Yann Rotella. *Cryptanalysis of the FLIP Family of Stream Ciphers*. Cryptology ePrint Archive, Report 2016/271. http://eprint.iacr.org/2016/271. 2016 (cit. on p. 83).

[DLR16b] Sébastien Duval, Virginie Lallemand, and Yann Rotella. “Cryptanalysis of the FLIP Family of Stream Ciphers”. In: *Advances in Cryptology – CRYPTO 2016, Part I*. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9814. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2016, pp. 457–475. DOI: 10.1007/978-3-662-53018-4_17 (cit. on pp. 83, 110, 153).

[DM15] Léo Ducas and Daniele Micciancio. “FHEW: Bootstrapping Homomorphic Encryption in Less Than a Second”. In: *Advances in Cryptology – EUROCRYPT 2015, Part I*. Ed. by Elisabeth Oswald and Marc Fischlin. Vol. 9056. Lecture Notes in Computer Science. Sofia, Bulgaria: Springer, Heidelberg, Germany, Apr. 2015, pp. 617–640. DOI: 10.1007/978-3-662-46800-5_24 (cit. on pp. 57, 67).

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Berlin, Heidelberg, New York: Springer Verlag, 2002. ISBN: 3-540-42580-2 (cit. on p. 62).

[DS09] Itai Dinur and Adi Shamir. “Cube Attacks on Tweakable Black Box Polynomials”. In: *Advances in Cryptology – EUROCRYPT 2009*. Ed. by Antoine Joux. Vol. 5479. Lecture Notes in Computer Science. Cologne, Germany: Springer, Heidelberg, Germany, Apr. 2009, pp. 278–299 (cit. on p. 82).

[DSES14] Yarkin Doröz, Aria Shahverdi, Thomas Eisenbarth, and Berk Sunar. “Toward Practical Homomorphic Evaluation of Block Ciphers Using Prince”. In: *FC 2014 Workshops*. Ed. by Rainer Böhme, Michael Brenner, Tyler Moore, and Matthew Smith. Vol. 8438. Lecture Notes in Computer Science. Christ Church, Barbados: Springer, Heidelberg, Germany, Mar. 2014, pp. 208–220. DOI: 10.1007/978-3-662-44774-1_17 (cit. on pp. 63, 74).

[Fau99] Jean-Charles Faugère. “A new efficient algorithm for computing Groebner bases”. In: *Journal of Pure and Applied Algebra* 139 (1999) (cit. on p. 79).

[FHK16] Pierre-Alain Fouque, Benjamin Hadjibeyli, and Paul Kirchner. "Homomorphic Evaluation of Lattice-Based Symmetric Encryption Schemes". In: *Computing and Combinatorics : 22nd International Conference, COCOON 2016, Ho Chi Minh City, Vietnam, August 2-4, 2016, Proceedings*. Ed. by Thang N. Dinh and My T. Thai. Cham: Springer International Publishing, 2016, pp. 269–280. DOI: 10.1007/978-3-319-42634-1_22. URL: https://doi.org/10.1007/978-3-319-42634-1_22 (cit. on pp. 66, 74).

[Fil16a] Yuval Filmus. "An Orthogonal Basis for Functions over a Slice of the Boolean Hypercube". In: *Electr. J. Comb.* 23.1 (2016), P1.23. URL: http://www.combinatorics.org/ojs/index.php/eljc/article/view/v23i1p23 (cit. on p. 119).

[Fil16b] Yuval Filmus. "Friedgut-Kalai-Naor Theorem for Slices of the Boolean Cube". In: *Chicago J. Theor. Comput. Sci.* 2016 (2016). URL: http://cjtcs.cs.uchicago.edu/articles/2016/14/contents.html (cit. on p. 119).

[FKMW16] Yuval Filmus, Guy Kindler, Elchanan Mossel, and Karl Wimmer. "Invariance Principle on the Slice". In: *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*. 2016, 15:1–15:10. DOI: 10.4230/LIPIcs.CCC.2016.15. URL: https://doi.org/10.4230/LIPIcs.CCC.2016.15 (cit. on p. 119).

[FM07] Simon Fischer and Willi Meier. "Algebraic Immunity of S-Boxes and Augmented Functions". In: *Fast Software Encryption – FSE 2007*. Ed. by Alex Biryukov. Vol. 4593. Lecture Notes in Computer Science. Luxembourg, Luxembourg: Springer, Heidelberg, Germany, Mar. 2007, pp. 366–381 (cit. on p. 82).

[FM16] Yuval Filmus and Elchanan Mossel. "Harmonicity and Invariance on Slices of the Boolean Cube". In: *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*. 2016, 16:1–16:13. DOI: 10.4230/LIPIcs.CCC.2016.16. URL: https://doi.org/10.4230/LIPIcs.CCC.2016.16 (cit. on p. 119).

[Gen09] Craig Gentry. "Fully homomorphic encryption using ideal lattices". In: *41st Annual ACM Symposium on Theory of Computing*. Ed. by Michael Mitzenmacher. Bethesda, MD, USA: ACM Press, May 2009, pp. 169–178 (cit. on pp. 3, 33).

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. "Public-Key Cryptosystems from Lattice Reduction Problems". In: *Advances in Cryptology – CRYPTO'97*. Ed. by Burton S. Kaliski Jr. Vol. 1294. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 1997, pp. 112–131 (cit. on p. 33).

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. "How to Construct Random Functions (Extended Abstract)". In: *25th Annual Symposium on Foundations of Computer Science*. Singer Island, Florida: IEEE Computer Society Press, Oct. 1984, pp. 464–479 (cit. on p. 15).

[GGPS17] Sugata Gangopadhyay, Aditi Kar Gangopadhyay, Spyridon Pollatos, and Pantelimon Stanica. "Cryptographic Boolean functions with biased inputs". In: *Cryptography and Communications* 9.2 (2017), pp. 301–314. DOI: 10.1007/s12095-015-0174-1. URL: https://doi.org/10.1007/s12095-015-0174-1 (cit. on p. 128).

[GH10] Craig Gentry and Shai Halevi. *Implementing Gentry's Fully-Homomorphic Encryption Scheme*. Cryptology ePrint Archive, Report 2010/520. http://eprint.iacr.org/2010/520. 2010 (cit. on p. 33).

[GH11] Craig Gentry and Shai Halevi. "Fully Homomorphic Encryption without Squashing Using Depth-3 Arithmetic Circuits". In: *52nd Annual Symposium on Foundations of Computer Science*. Ed. by Rafail Ostrovsky. Palm Springs, CA, USA: IEEE Computer Society Press, Oct. 2011, pp. 107–109 (cit. on p. 34).

[GHS12] Craig Gentry, Shai Halevi, and Nigel P. Smart. "Homomorphic Evaluation of the AES Circuit". In: *Advances in Cryptology – CRYPTO 2012*. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2012, pp. 850–867 (cit. on pp. 57, 63, 67, 74, 75).

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for all One-Way Functions". In: *21st Annual ACM Symposium on Theory of Computing*. Seattle, WA, USA: ACM Press, May 1989, pp. 25–32 (cit. on p. 15).

[GMW15] Romain Gay, Pierrick Méaux, and Hoeteck Wee. "Predicate Encryption for Multi-dimensional Range Queries from Lattices". In: *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*. Ed. by Jonathan Katz. Vol. 9020. Lecture Notes in Computer Science. Gaithersburg, MD, USA: Springer, Heidelberg, Germany, Mar. 2015, pp. 752–776. DOI: 10.1007/978-3-662-46447-2_34 (cit. on p. 8).

[GN08] Nicolas Gama and Phong Q. Nguyen. "Predicting Lattice Reduction". In: *Advances in Cryptology – EUROCRYPT 2008*. Ed. by Nigel P. Smart. Vol. 4965. Lecture Notes in Computer Science. Istanbul, Turkey: Springer, Heidelberg, Germany, Apr. 2008, pp. 31–51 (cit. on p. 75).

[Gol00] Oded Goldreich. "Candidate One-Way Functions Based on Expander Graphs". In: *Electronic Colloquium on Computational Complexity (ECCC)* 7.90 (2000). URL: http://eccc.hpi-web.de/eccc-reports/2000/TR00-090/index.html (cit. on pp. 150, 151).

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Vol. 1. Cambridge, UK: Cambridge University Press, 2001, pp. xix + 372. ISBN: 0-521-79172-3 (hardback) (cit. on pp. 12, 14, 15).

[Got66] D. H. Gottlieb. "A Certain Class of Incidence Matrices". In: *Proceedings of the American Mathematical Society* 17.6 (1966), pp. 1233–1237. ISSN: 00029939, 10886826. URL: http://www.jstor.org/stable/2035716 (cit. on p. 126).

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. "Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based". In: *Advances in Cryptology – CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2013, pp. 75–92. DOI: 10.1007/978-3-642-40041-4_5 (cit. on pp. 36, 39).

[HAO15] Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. "Packing Messages and Optimizing Bootstrapping in GSW-FHE". In: *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*. Ed. by Jonathan Katz. Vol. 9020. Lecture Notes in Computer Science. Gaithersburg, MD, USA: Springer, Heidelberg, Germany, Mar. 2015, pp. 699–715. DOI: 10.1007/978-3-662-46447-2_31 (cit. on pp. 37, 39, 57).

[HS14] Shai Halevi and Victor Shoup. "Algorithms in HElib". In: *Advances in Cryptology – CRYPTO 2014, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2014, pp. 554–571. DOI: 10.1007/978-3-662-44371-2_31 (cit. on p. 64).

[JD06] Antoine Joux and Pascal Delaunay. "Galois LFSR, Embedded Devices and Side Channel Weaknesses". In: *Progress in Cryptology - INDOCRYPT 2006: 7th International Conference in Cryptology in India*. Ed. by Rana Barua and Tanja Lange. Vol. 4329. Lecture Notes in Computer Science. Kolkata, India: Springer, Heidelberg, Germany, Dec. 2006, pp. 436–451 (cit. on p. 140).

[KGV14] Alhassan Khedr, Glenn Gulak, and Vinod Vaikuntanathan. *SHIELD: Scalable Homomorphic Implementation of Encrypted Data-Classifiers*. Cryptology ePrint Archive, Report 2014/838. http://eprint.iacr.org/2014/838. 2014 (cit. on pp. 39, 75).

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman, Hall/Crc Cryptography, and Network Security Series, 2007. ISBN: ISBN:1584885513 (cit. on p. 70).

[KMN10] Simon Knellwolf, Willi Meier, and María Naya-Plasencia. "Conditional Differential Cryptanalysis of NLFSR-Based Cryptosystems". In: *Advances in Cryptology – ASIACRYPT 2010*. Ed. by Masayuki Abe. Vol. 6477. Lecture Notes in Computer Science. Singapore: Springer, Heidelberg, Germany, Dec. 2010, pp. 130–145 (cit. on p. 83).

[Knu97] Donald E. Knuth. *Seminumerical Algorithms*. Third. Vol. 2. The Art of Computer Programming. Addison-Wesley Professional, Nov. 1997 (cit. on pp. 71, 78).

[KW02] Lars R. Knudsen and David Wagner. "Integral Cryptanalysis". In: *Fast Software Encryption – FSE 2002*. Ed. by Joan Daemen and Vincent Rijmen. Vol. 2365. Lecture Notes in Computer Science. Leuven, Belgium: Springer, Heidelberg, Germany, Feb. 2002, pp. 112–127 (cit. on p. 83).

[Lal16] Virginie Lallemand. “Cryptanalyse de chiffrements symétriques. (Cryptanalysis of symmetric ciphers)”. PhD thesis. Pierre and Marie Curie University, Paris, France, 2016. URL: https://tel.archives-ouvertes.fr/tel-01405436 (cit. on p. 10).

[LF06] Éric Levieil and Pierre-Alain Fouque. “An Improved LPN Algorithm”. In: *SCN 06: 5th International Conference on Security in Communication Networks*. Ed. by Roberto De Prisco and Moti Yung. Vol. 4116. Lecture Notes in Computer Science. Maiori, Italy: Springer, Heidelberg, Germany, Sept. 2006, pp. 348–359 (cit. on pp. 81, 88).

[LLL82] Arjen Lenstra, Hendrik Lenstra, and Lászlò Lovász. “Factoring polynomials with rational coefficients”. In: *Math. Ann.* 261 (1982), pp. 515–534 (cit. on p. 57).

[LN14] Tancrède Lepoint and Michael Naehrig. “A Comparison of the Homomorphic Encryption Schemes FV and YASHE”. In: *AFRICACRYPT 14: 7th International Conference on Cryptology in Africa*. Ed. by David Pointcheval and Damien Vergnaud. Vol. 8469. Lecture Notes in Computer Science. Marrakesh, Morocco: Springer, Heidelberg, Germany, May 2014, pp. 318–335. DOI: 10.1007/978-3-319-06734-6_20 (cit. on pp. 63, 74).

[LNV11] Kristin Lauter, Michael Naehrig, and Vinod Vaikuntanathan. *Can Homomorphic Encryption be Practical?* Cryptology ePrint Archive, Report 2011/405. http://eprint.iacr.org/2011/405. 2011 (cit. on pp. 4, 58).

[LP11] Richard Lindner and Chris Peikert. “Better Key Sizes (and Attacks) for LWE-Based Encryption”. In: *Topics in Cryptology – CT-RSA 2011*. Ed. by Aggelos Kiayias. Vol. 6558. Lecture Notes in Computer Science. San Francisco, CA, USA: Springer, Heidelberg, Germany, Feb. 2011, pp. 319–339 (cit. on p. 75).

[LP13] Tancrède Lepoint and Pascal Paillier. “On the Minimal Number of Bootstrappings in Homomorphic Circuits”. In: *FC 2013 Workshops*. Ed. by Andrew A. Adams, Michael Brenner, and Matthew Smith. Lecture Notes in Computer Science. Okinawa, Japan: Springer, Heidelberg, Germany, Apr. 2013, pp. 189–200. DOI: 10.1007/978-3-642-41320-9_13 (cit. on p. 58).

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. “On Ideal Lattices and Learning with Errors over Rings”. In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by Henri Gilbert. Vol. 6110. Lecture Notes in Computer Science. French Riviera: Springer, Heidelberg, Germany, May 2010, pp. 1–23 (cit. on p. 22).

[LR88] Michael Luby and Charles Rackoff. “How to construct pseudorandom permutations from pseudorandom functions”. In: *SIAM Journal on Computing* 17.2 (1988) (cit. on pp. 15, 64, 78).

[LRW11] Moses Liskov, Ronald L. Rivest, and David Wagner. “Tweakable Block Ciphers”. In: *Journal of Cryptology* 24.3 (July 2011), pp. 588–613 (cit. on p. 79).

[LS12] Adeline Langlois and Damien Stehlé. *Worst-Case to Average-Case Reductions for Module Lattices*. Cryptology ePrint Archive, Report 2012/090. http://eprint.iacr.org/2012/090. 2012 (cit. on p. 66).

[LT17] Huijia Lin and Stefano Tessaro. *Indistinguishability Obfuscation from Trilinear Maps and Block-Wise Local PRGs*. Cryptology ePrint Archive, Report 2017/250. http://eprint.iacr.org/2017/250. 2017 (cit. on p. 151).

[McF73] Robert L McFarland. "A family of difference sets in non-cyclic groups". In: *Journal of Combinatorial Theory, Series A* 15.1 (1973), pp. 1–10. ISSN: 0097-3165. DOI: https://doi.org/10.1016/0097-3165(73)90031-9. URL: http://www.sciencedirect.com/science/article/pii/0097316573900319 (cit. on p. 25).

[Mei11] Willi Meier. "Fast Correlation Attacks: Methods and Countermeasures (Invited Talk)". In: *Fast Software Encryption – FSE 2011*. Ed. by Antoine Joux. Vol. 6733. Lecture Notes in Computer Science. Lyngby, Denmark: Springer, Heidelberg, Germany, Feb. 2011, pp. 55–67 (cit. on p. 82).

[Mes17] Sihem Mesnager. *On the nonlinearity of Boolean functions with restricted input*. Talk at The 13th International Conference on Finite Fields and their Applications. 2017 (cit. on pp. 128, 131, 134).

[Mic10] Daniele Micciancio. "A first glimpse of cryptography′s Holy Grail". In: *Communications of the ACM* 53 Issue 3 (2010), pp. 96–96 (cit. on p. 33).

[MJSC16] Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. "Towards Stream Ciphers for Efficient FHE with Low-Noise Ciphertexts". In: *Advances in Cryptology – EUROCRYPT 2016, Part I*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9665. Lecture Notes in Computer Science. Vienna, Austria: Springer, Heidelberg, Germany, May 2016, pp. 311–343. DOI: 10.1007/978-3-662-49890-3_13 (cit. on pp. 6, 32, 62, 64, 71, 74, 83, 84, 96, 100, 101, 110, 118, 154).

[MP12] Daniele Micciancio and Chris Peikert. "Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller". In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Vol. 7237. Lecture Notes in Computer Science. Cambridge, UK: Springer, Heidelberg, Germany, Apr. 2012, pp. 700–718 (cit. on p. 38).

[MS88] Willi Meier and Othmar Staffelbach. "Fast Correltaion Attacks on Stream Ciphers (Extended Abstract)". In: *Advances in Cryptology – EUROCRYPT'88*. Ed. by C. G. Günther. Vol. 330. Lecture Notes in Computer Science. Davos, Switzerland: Springer, Heidelberg, Germany, May 1988, pp. 301–314 (cit. on pp. 26, 81, 82, 88, 128).

[MvV97] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. The CRC Press series on discrete mathematics and its applications. 2000 N.W. Corporate Blvd., Boca Raton, FL 33431-9868, USA: CRC Press, 1997, pp. xxviii + 780. ISBN: 0-8493-8523-7 (cit. on p. 16).

[OPS15] Emmanuela Orsini, Joop van de Pol, and Nigel P. Smart. "Bootstrapping BGV Ciphertexts with a Wider Choice of p and q". In: *PKC 2015: 18th International Conference on Theory and Practice of Public Key Cryptography*. Ed. by Jonathan Katz. Vol. 9020. Lecture Notes in Computer Science. Gaithersburg, MD, USA: Springer, Heidelberg, Germany, Mar. 2015, pp. 673–698. DOI: 10.1007/978-3-662-46447-2_30 (cit. on p. 57).

[Pas16] Alain Passelègue. “Algebraic Frameworks for Pseudorandom Functions”. PhD thesis. PSL Research University, Paris, France, 2016. URL: https://tel.archives-ouvertes.fr/tel-01422093 (cit. on p. 10).

[Pei15] Chris Peikert. *A Decade of Lattice Cryptography*. Cryptology ePrint Archive, Report 2015/939. http://eprint.iacr.org/2015/939. 2015 (cit. on p. 21).

[Pre15] Thomas Prest. “Gaussian Sampling in Lattice-Based Cryptography”. PhD thesis. École Normale Supérieure, Paris, France, 2015. URL: https://tel.archives-ouvertes.fr/tel-01245066 (cit. on p. 10).

[PV16] Marie Paindavoine and Bastien Vialla. “Minimizing the Number of Bootstrappings in Fully Homomorphic Encryption”. In: *SAC 2015: 22nd Annual International Workshop on Selected Areas in Cryptography*. Ed. by Orr Dunkelman and Liam Keliher. Vol. 9566. Lecture Notes in Computer Science. Sackville, NB, Canada: Springer, Heidelberg, Germany, Aug. 2016, pp. 25–43. DOI: 10.1007/978-3-319-31301-6_2 (cit. on p. 58).

[RAD78] Ron Rivest, Len Adleman, and Michael Dertouzos. “On Data Banks and Privacy Homomorphisms”. In: *Foundations of Secure Computation, Academia Press* (1978), pp. 169–179 (cit. on p. 33).

[Reg05] Oded Regev. “On lattices, learning with errors, random linear codes, and cryptography”. In: *37th Annual ACM Symposium on Theory of Computing*. Ed. by Harold N. Gabow and Ronald Fagin. Baltimore, MA, USA: ACM Press, May 2005, pp. 84–93 (cit. on pp. 22, 75).

[RS10] Markus Rückert and Michael Schneider. *Estimating the Security of Lattice-based Cryptosystems*. Cryptology ePrint Archive, Report 2010/137. http://eprint.iacr.org/2010/137. 2010 (cit. on p. 75).

[SE94] Claus-Peter Schnorr and M. Euchner. “Lattice basis reduction: improved practical algorithms and solving subset sum problems”. In: *Math. Programming* 66 (1994), pp. 181–199 (cit. on pp. 57, 75).

[Sie84] Thomas Siegenthaler. “Correlation-immunity of nonlinear combining functions for cryptographic applications”. In: *IEEE* IT-30.5 (1984), pp. 776–780 (cit. on pp. 80, 88).

[Sie85] Thomas Siegenthaler. “Decrypting a Class of Stream Ciphers Using Ciphertext Only”. In: *IEEE Trans. Computers* 34.1 (1985), pp. 81–85. DOI: 10.1109/TC.1985.1676518. URL: http://doi.ieeecomputersociety.org/10.1109/TC.1985.1676518 (cit. on p. 26).

[SPY13] François-Xavier Standaert, Olivier Pereira, and Yu Yu. “Leakage-Resilient Symmetric Cryptography under Empirically Verifiable Assumptions”. In: *Advances in Cryptology – CRYPTO 2013, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Heidelberg, Germany, Aug. 2013, pp. 335–352. DOI: 10.1007/978-3-642-40041-4_19 (cit. on p. 71).

[SSTX09] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. *Efficient Public Key Encryption Based on Ideal Lattices*. Cryptology ePrint Archive, Report 2009/285. http://eprint.iacr.org/2009/285. 2009 (cit. on p. 22).

[SV10] Nigel P. Smart and Frederik Vercauteren. "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes". In: *PKC 2010: 13th International Conference on Theory and Practice of Public Key Cryptography*. Ed. by Phong Q. Nguyen and David Pointcheval. Vol. 6056. Lecture Notes in Computer Science. Paris, France: Springer, Heidelberg, Germany, May 2010, pp. 420–443 (cit. on p. 34).

[SV11] N.P. Smart and F. Vercauteren. *Fully Homomorphic SIMD Operations*. Cryptology ePrint Archive, Report 2011/133. http://eprint.iacr.org/2011/133. 2011 (cit. on p. 36).

[Wie86] D H Wiedemann. "Solving Sparse Linear Equations over Finite Fields". In: *IEEE Trans. Inf. Theor.* 32.1 (Jan. 1986), pp. 54–62. ISSN: 0018-9448. DOI: 10.1109/TIT.1986.1057137. URL: http://dx.doi.org/10.1109/TIT.1986.1057137 (cit. on p. 79).

[Wil90] Richard M. Wilson. "A Diagonal Form for the Incidence Matrices of t-Subsets vs.k-Subsets". In: *European Journal of Combinatorics* 11.6 (1990), pp. 609–615. ISSN: 0195-6698. DOI: https://doi.org/10.1016/S0195-6698(13)80046-7. URL: http://www.sciencedirect.com/science/article/pii/S0195669813800467 (cit. on p. 126).

[XM88] Guo-Zhen Xiao and James L. Massey. "A spectral characterization of correlation-immune combining functions". In: *IEEE Trans. Information Theory* 34.3 (1988), pp. 569–571. DOI: 10.1109/18.6037. URL: https://doi.org/10.1109/18.6037 (cit. on p. 25).

Résumé

Le chiffrement complètement homomorphe est une classe de chiffrement permettant de calculer n'importe quelle fonction sur des données chiffrées et de produire une version chiffrée du résultat. Il permet de déléguer des données à un cloud de façon sécurisée, faire effectuer des calculs, tout en gardant le caractère privé de ces données. Cependant, l'innéficacité actuelle des schémas de chiffrement complètement homomorphes, et leur inadéquation au contexte de délégation de calculs, rend son usage seul insuffisant pour cette application. Ces deux problèmes peuvent être résolus, en utilisant ce chiffrement dans un cadre plus large, en le combinant avec un schéma de chiffrement symétrique. Cette combinaison donne naissance au chiffrement complètement homomorphe hybride, conçu dans le but d'une délégation de calculs efficace, garantissant des notions de sécurité et de vie privée.

Dans cette thèse, nous étudions le chiffrement complètement homomorphe hybride et ses composantes, à travers la conception de primitives cryptographiques symétriques rendant efficace cette construction hybride. En examinant les schémas de chiffrement complètement homomorphes, nous developpons des outils pour utiliser efficacement leurs propriétés homomorphiques dans un cadre plus complexe. En analysant différents schémas symétriques, et leurs composantes, nous déterminons de bons candidats pour le contexte hybride. En étudiant la sécurité des constructions optimisant l'évaluation homomorphique, nous contribuons au domaine des fonctions booléennes utilisées en cryptologie.

Plus particulièrement, nous introduisons une nouvelle famille de schémas de chiffrement symétriques, avec une nouvelle construction, adaptée au contexte hybride. Ensuite, nous nous intéressons à son comportement homomorphique, et nous étudions la sécurité de cette construction. Finalement, les particularités de cette famille de schémas de chiffrement motivant des cryptanalyses spécifiques, nous développons et analysons de nouveaux critères cryptographiques booléens.

Mots Clés

cryptographie, chiffrement complètement homomorphe, fonctions booléennes

Abstract

Fully homomorphic encryption, firstly built in 2009, is a very powerful kind of encryption, allowing to compute any function on encrypted data, and to get an encrypted version of the result. Such encryption enables to securely delegate data to a cloud, ask for computations, recover the result, while keeping private the data during the whole process. However, today's inefficiency of fully homomorphic encryption, and its inadequateness to the outsourcing computation context, makes its use alone insufficient for this application. Both of these issues can be circumvented, using fully homomorphic encryption in a larger framework, by combining it with a symmetric encryption scheme. This combination gives a hybrid fully homomorphic framework, designed towards efficient outsourcing computation, providing both security and privacy.

In this thesis, we contribute to the study of hybrid fully homomorphic framework, through the analysis, and the design of symmetric primitives making efficient this hybrid construction. Through the examination of fully homomorphic encryption schemes, we develop tools to efficiently use the homomorphic properties in a more complex framework. By investigating various symmetric encryption schemes, and building blocks up to the circuit level, we determine good candidates for a hybrid context. Through evaluating the security of constructions optimizing the homomorphic evaluation, we contribute to a wide study within the cryptographic Boolean functions area.

More particularly, we introduce a new family of symmetric encryption schemes, with a new design, adapted to the hybrid fully homomorphic framework. We then investigate its behavior relatively to homomorphic evaluation, and we address the security of such design. Finally, particularities of this family of ciphers motivate specific cryptanalyses, therefore we develop and analyze new cryptographic Boolean criteria.

Keywords

cryptography, fully homomorphic encryption, Boolean functions