UNIVERSITY OF TRENTO

DEPARTMENT OF MATHEMATICS

Code-Based Digital Signature Schemes:

Construction, Cryptanalysis and Theoretical Foundations

Giovanni Tognolini

PhD thesis in Mathematics

Doctoral programme — XXXVII cycle

Supervised by Nadir Murru

and Alessio Meneghetti

“ἀμὴν ἀμὴν λέγω ὑμῖν, ἐὰν μὴ ὁ κόκκος τοῦ σίτου πεσὼν εἰς τὴν γῆν ἀποθάνῃ, αὐτὸς μόνος μένει· ἐὰν δὲ ἀποθάνῃ, πολὺν καρπὸν φέρει.”

Κατά Ιωάννην, 12:24.

Abstract

In recent decades, numerous code-based digital signature schemes following the hash&sign paradigm have been proposed, often turning out to be insecure and demonstrating that the cryptographic community still seems far from a satisfactory solution in this area.

The first major question we pose in this thesis is whether it is possible to construct, in this regard, a signature scheme that enjoys a solid security reduction and is also efficient. We discuss a proposal that combines some new ideas, but we anticipate that it suffered a severe attack two years ago, making it unusable.

The second question we pose is the possibility to say something about the security of other schemes. According to this, we leave the constructive side and explore the cryptanalytic one, analyzing HWQCS. We break its security assumptions and violate the EUF-CMA security. We also attempt to break Wave, and although the cryptanalysis we propose will not be successful, the intermediate results we obtained will have value in themselves and will provide nice material to address other research questions.

Digital signatures can also be obtained by transforming an interactive protocol into a non-interactive one, typically using the Fiat-Shamir transform. Recently, numerous optimizations to the basic paradigm have been introduced, including the well-known fixed-weight optimization. Although this technique is widely used, its underlying security assumptions are still not well understood, and the formal security of these schemes has not yet been proven. With the intention of laying a first brick in this direction, we prove that the underlying interactive protocol still enjoys knowledge soundness. Finally, we asked ourselves if it is possible to follow an alternative path, building a transform that is fixed-weight by design and easily produces secure signatures.

These are the main questions targeted by this thesis, forming the *fil rouge* that guides the narrative. In the process of addressing them, several secondary questions will naturally arise, which will be described and explored throughout the discussion.

Acknowledgements

Credo sia una tendenza ormai consolidata quella di sovrapporre, semanticamente parlando, alcune nostre esperienze a dei viaggi in mare aperto. Tre anni fa ho incominciato questo viaggio, avventurandomi con una mappa incompleta in un mare che non conoscevo e lanciandomi spesso in mezzo a correnti ignote. In entrambi i casi urge la presenza di qualcuno che ci sappia indicare una direzione, una stella da puntare; qualcuno che quelle correnti le abbia già percorse.

Desidero esprimere la mia più profonda gratitudine ad Alessio Meneghetti e Nadir Murru per la guida costante e la fiducia che hanno riposto in me durante questi anni, dandomi la forza di navigare con perseveranza attraverso le acque più agitate. Vorrei ringraziare anche Marco Baldi e Paolo Santini, con cui ho avuto il piacere e il privilegio di collaborare. Grazie per avermi accompagnato fin qua, per il supporto, per i consigli, e più in generale essere stati un aiuto prezioso nelle giornate di calma apparente, quando domande insolute si sollevavano dal nulla come onde gigantesche. Vorrei ringraziarvi in particolare per il piacere provato in quelle giornate limpide, trascorse insieme a aggiungere nuovi dettagli a quelle mappe con cui tutti siamo partiti, con la speranza che chi ci seguirà possa trarre piacere e utilità dai nostri sforzi, così come abbiamo fatto noi quando ci siamo imbarcati per questo viaggio.

Il mondo della ricerca si affianca inevitabilmente a un contesto decisamente più grande. Di seguito vorrei ringraziare esplicitamente alcune persone con le quali ho condiviso molto negli anni e che hanno contribuito in modo significativo a definire la persona che sono.

Alle persone con cui sono cresciuto fin da piccolo, la piccola compagnia di Castegnato, in particolare a Balli, il nostro diamante pazzo, a Verze, il nosto meccanico di fiducia, a Trap, lo spirito più allegro del gruppo, a Mangi, il nostro pittore senza tempo, a Break, per averci mostrato la forza del ritorno, e a Raffo, che accoglieremo sempre a braccia aperte nonostante lo spazio che ci separa. Più in generale vorrei ringraziare tutta la compagnia di Brescia. Grazie di tutto, le parole non bastano davvero per descrivervi.

Alle persone con cui ho condiviso gli studi. A Pietro, per tutto il tempo che abbiamo passato insieme: la persona che sono la devo in buona parte anche a te. Ad Andrè, un faro di umanità e speranza, un compagno di studio nelle giornate più intense, e il compagno di bici più affiatato nelle sere più limpide.

Alle persone che ho incontrato durante il dottorato. A Marzio, che si è sorbito la quasi totalità dei miei malumori recenti. Grazie per la pazienza, e per tutto quello che abbiamo costruito insieme in questi anni, dentro e fuori dall'università.

A Маша e Наташа, con le quali ho condiviso la vita in casa per tutto questo tempo: mi avete regalato un'umanità senza eguali. Спасибо за все.

A tutti voi: per quanto ognuno segua la sua stella, e per quanto le nostre rotte siano diverse, mi auguro che si intersechino ancora molte volte.

L'ultimo pensiero, senza dubbio il più importante, va ai miei genitori, e più in generale a tutta la mia famiglia, unico punto fermo della mia vita. Un porto sicuro in cui ritornare, soffermarsi, e ritrovare se stessi.

"Caelum, non animum mutant, qui trans mare currunt."

Epistulae I, 11, 27, Orazio

Contents

List of Symbols

$A(\mathbb{F}_q, n, d)$	Maximal size of a code in $\mathbb{F}_q^n$ with min. distance d.
$\mathcal{A}$	Adversary
$\textsc{ApproxWEP}^{\varepsilon}$	Approximate Weight Enumerator Problem
$B(\mathbb{F}_q, x, n, t)$	(closed) Hamminh ball of center x and radius t in $\mathbb{F}_q^n$
$B_t(x)$	Compact notation for $B(\mathbb{F}_q, x, n, t)$
C	Linear Code
C_w	Codewords of C with weight w
CFP	Codeword Finding Problem
circ	Circulant matrix function
Ch	Challenge space
DOOM	Decoding One Out of Many (Problem)
ϵ_c^{P}	Completeness error of P
$\epsilon_s^{\mathsf{P},\mathsf{Ext}}$	Soundness error of P w.r.t. an online extractor Ext
Ext	Extractor
$\mathbb{F}_q$	Finite field with q elements
ϕ_n	n-th cyclitomic polynomial
$\mathbf{G}$	Generator matrix of a code
$\mathbf{H}$	Parity-check matrix of a code
H	Hash function
h_q	q-ary entropy function
H_w	Weighted hash function (with weight w)
HVZK	Honest-Verifier Zero-Knowledge (protocol)
$\mathbf{I}_n$	Identity matrix of dimension n
IP	Interactive Polynomial (complexity class)
ISD	Information Set Decoding algorithm(s)
κ	Knowledge error
KeyGen	Key generation algorithm
LDPC	Low-Density Parity-Check (code)
M	Turing machine
$N_C(w)$	Number of codewords of C with weight w
NP	Nondeterministic Polynomial (complexity class)
P	Polynomial (complexity class)
P	Prover
pk	Public key
PoK	Proof of Knowledge
PPT	Probabilistic Polynomial Time (algorithm)
QC	Quasi-Cyclic codes
Σ	Sigma protocol

$\Sigma_\star$	Sigma protocol from a cryptographic group action
$\mathcal{S}$	Digital signature algorithm
Sim	Simulator
SDP	Syndrome Decoding Problem
Sign	Signing algorithm
sk	Secret key
Supp	Support function
$V_{n,w}$	Set of all vectors of length n and Hamming weight w
V	Verifier
Vf	Verification algorithm
w	Hamming weight function
WEP	Weight Enumerator Problem
$\leq_p$	Polynomially reducible

Introduction

"Wer dies Wasser und seine Geheimnisse verstünde, so schien ihm, der würde auch viel anderes verstehen, viele Geheimnisse, alle Geheimnisse."

Siddhartha, H. Hesse

Intuitively, we would like every cryptographic scheme to ultimately base its security on some computationally intractable problem. A formalization of this intuition is provided by NP-complete problems, introduced by Karp in 1972 as the hardest problems within the NP class [Kar72].

The first public-key cryptographic schemes, in particular the key exchange protocol proposed by Diffie and Hellman in 1976 [DH22] and the RSA scheme developed one year later by Rivest, Shamir, and Adleman [RSA83], are based on the reasonable assumption that the discrete logarithm and integer factorization problems are computationally intractable. Due to their structural simplicity, small key sizes, and good performance, these algorithms have garnered increasing attention since their birth, even though no reduction proving these two problems to belong to the NP-complete class has ever been established. A few years later, in 1978, Berlekamp, McEliece, and van Tilborg demonstrated that two well-known problems related to linear codes, namely the Syndrome Decoding Problem and the Codeword Finding Problem, belong to the NP-complete class, paving the way for their potential use in cryptography [BMVT78]. Indeed, in the same year, McEliece came up with a public-key cryptosystem based on these problems [McE78], and a dual version of this cryptosystem was later proposed by Niederreiter in 1986 [Nie86]. Despite the underlying problems being NP-complete, these schemes faced significant challenges due to their large key sizes and limited practical feasibility compared to competing schemes such as RSA.

The course of events might have been different if Peter Shor had not developed, in 1994, a quantum algorithm capable of solving both factorization and discrete logarithm problems in polynomial time [Sho94]. Although there are currently no quantum computers capable of satisfactorily implementing this algorithm, the theoretical description of the attack alone has motivated the cryptography community to begin a transition to communication methods, in particular encryption and digital signature schemes, which can be considered reasonably safe even in front of a possible quantum computer. In this sense, NP-complete problems represent natural starting points for this research.

While McEliece's algorithm offers, almost half a century after its introduction, a solid scheme that demonstrates the validity of linear codes in the creation of encryption schemes, many efforts have been made to try to use linear codes to also build digital signature schemes. In this regard, two constructive frameworks can be distinguished. On the one

hand, there are schemes following the hash&sign paradigm, while on the other hand, there are schemes based on proofs of knowledge. The former approach requires the presence of a structured code for which efficient decoding is possible, and it also demands masking this code so that it appears as a random one to an adversary. The security of the scheme is thus primarily based on the assumption that the adversary cannot obtain any information about the structured code and, secondarily, on the fact that decoding a random code is an intractable problem. Despite we know this second problem to be NP-complete, the indistinguishability of the structured codes underlying these schemes from random ones is a much more delicate assumption, which has sometimes proven to be simply false [Fau+13; GOT12]. Among the folds of these assumptions lies another issue: information leakage, a phenomenon that afflicts many of these schemes, for which some bits of the private key (involved in the signing process) could be reconstructed by an adversary. In recent decades, numerous code-based digital signature schemes following the hash&sign paradigm have been proposed [CFS01; Dal07; KKS97; RZW+17; Bal+13; Per18; Gab+14; Kim+22; Rit+23; FRX+17; Son+20; LXY20; PMT22; TP24; DAST19], often turning out to be insecure [OT11; DMP21; PT16; SBC19; Xag18; DG20; HW24; Ara+21; Bal+21b; PT23; PT24] and demonstrating that the cryptographic community still seems far from a satisfactory solution in this area. Digital signatures can also be obtained by transforming an interactive protocol into a non-interactive one, typically using the Fiat-Shamir transform. Loosely speaking, if the interactive protocol is complete, knowledge-sound and zero-knowledge, it is possible to apply the Fiat-Shamir transform and obtain an EUF-CMA secure signature scheme. In particular, the zero-knowledge property prevents the possibility of information leakage and solves numerous problems of the hash&sign framework. Recently, numerous optimizations to the basic paradigm have been introduced, including the well-known seed tree, fixed-weight and multiple public key optimization, which modify the interactive protocol, and consequently the non-interactive one. As a result of these ongoing efforts, as of now the NIST second call for digital signature schemes still has two code-based schemes in the running, namely CROSS[Bal+23a] and LESS [Bia+20], both of which follow the Fiat-Shamir framework, with the optimizations mentioned above.

This thesis moves in the generic framework outlined above and represents part of the author's work as a PhD student in Trento. About two-thirds of the materials collected here have already appeared in preprints [PMT22; Bat+24] and publications [PT24]: only minor revisions were made to fit them into a more coherent and up-to-date dissertation. Other sections are instead unpublished, as they have not (yet) reached the maturity necessary to be considered articles in their own right. The following table summarizes the main sources for each part of this thesis.

Sec 1.1, 1.2, 1.3	Known results
Sec 2.1	[PMT22]
Sec 3.1	[PT24]
Sec 3.2, 3.3	Unpublished
Sec 4.1	Known results
Sec 4.2	[Bat+24]
Sec 4.3	Unpublished

We detail below the content of each chapter:

- Chapter 1 serves as a mathematical introduction and gathers some classic results on how complexity theory relates to cryptography, as well as some basic notion of digital

signatures and linear codes.

- Chapter 2 explores the possibility of building a code-based hash&sign signature scheme using QC-LDPC codes. We will describe the ideas behind this construction, as well as an attack presented in 2023 [PT23] which undermined the security of this scheme.

- Chapter 3 shifts the attention from the constructive to the cryptanalytic level. Section 3.1 is completely devoted to the cryptanalysis of HWQCS [TP24], a signature scheme born from the ashes of the one proposed in the previous chapter. In particular, we will show how the scheme is affected by an information leakage, and how to exploit it to mount a universal forgery attack [PT24]. Section 3.2 is dedicated to Wave [DAST19] and to an attempt to break this scheme. In particular, we will try to break the formal security building an efficient distinguisher that can recognize with good probability whether a given input code is a random code or whether it is one of the structured codes involved in this scheme. Although the cryptanalysis will turn out not to be successful, the ideas behind this attempt will provide sufficient material for another work, exposed in Section 3.3. In this section we will modify the standard ISD paradigm and use these algorithms to determine the weight distribution of an given code.

- Chapter 4 moves attention from cryptanalysis to the theoretical foundations of cryptography. The motivations on which this chapter is based are to be found in the code-based schemes recently proposed at the NIST PQC, and in particular in the fixed-weight optimization which have been introduced in these schemes. Although this optimization is widely used, the security assumptions underlying it are still not well understood. In particular, one question that has been overlooked is whether the modified interactive protocol still satisfies the knowledge-soundness property. With the intention of recovering the formal security of these schemes, in Section 4.2 we address and provide a positive answer to this question, laying a first brick in this direction [Bat+24]. Finally, in Section 4.3 we focus on non-interactive zero-knowledge proofs, trying to construct a new transform, which is fixed weight by design and produces provably secure proofs without asking to modify (and so proving properties on) the original interactive protocol.

The work presented in this thesis is designed for transversal reading, so that the reader interested in a specific topic can reach it in the easiest way possible. Despite this, we suggest a chronological reading order, which, in our opinion, also allows for a pleasant climax from the survey-level of Chapter 1 to the more technical analysis of Chapter 4.

Chapter 1

Preliminaries

In the following, we introduce some fundamental concepts from complexity theory that will be useful throughout our discussion. The description we provide in the following is mainly taken from [AB09] and [BS20], but also [GJ79b; Kar75; Kar10; GMR88; BMVT78]. We refer the interested reader to these works for further details. The last part of the chapter is devoted to linear codes. In this regard, results are mostly taken from [WGR22].

1.1 Complexity Theory: a Cryptographic Approach

> *"The great importance ... [of] Turing's computability is largely due to the fact that with this concept one has for the first time succeeded in giving an absolute definition of an interesting epistemological notion, i.e., one not depending on the formalism chosen."*
>
> K. Gödel, 1946

In this thesis, we will only deal with computable functions (for which an algorithm can be described to compute the result of the function in a finite number of steps) and our main concern will be about their complexity, namely the relationship between the input size and the execution time of an algorithm that computes this function. Turing machines formalize this concept clearly [AB09]. Informally, a Turing machine is a mathematical model of computation describing an abstract machine that manipulates symbols on a strip of tape according to a table of rules. Despite the model's simplicity, it is capable of implementing any algorithm. We still refer to Turing machines because both the input size and the execution time can be defined unambiguously in this model.

1.1.1 Languages: P, NP and IP

Decision problems are one of the central topics in computational complexity theory. A decision problem is a computational problem with a binary outcome: either *yes* or *no*. In the first case, we say that the algorithm accepts the input string; otherwise, we say that the algorithm rejects it. Such problems can be represented as formal languages, that is, subsets of $\{0,1\}^*$, where instances producing a *yes* result are members of the language, and those producing a *no* result are non-members. Given a formal language (i.e., a decision problem), the goal is to design an algorithm capable of determining whether a given input string belongs to that language. Complexity theory investigates, among other aspects, the complexity of such algorithms, with the tacit aim of dividing languages in fundamental

complexity classes. We give a description of just some of them, which are very important for the cryptography side.

The complexity class P, namely, the polynomial-time class, contains all decision problems that can be solved by a deterministic Turing machine in polynomial time.

Definition 1.1.1 (P)**.** A language $L \subseteq \{0,1\}^*$ is *polynomial* if there is a polynomial-time Turing machine M such that for every input string $x \in \{0,1\}^*$

$$x \in L \iff \mathrm{M}(x) = 1.$$

The set of all polynomial languages is denoted with P.

Over time, and still today, research has come up against languages for which it is not clear whether they are decidable in polynomial time or not. From here the necessity to formalize a slightly more refined class of languages, i.e. those languages (which perhaps we do not know how to solve in a reasonable time, but) for which someone (normally referred to as "the prover") who knows how to solve them in an affirmative way can convince us too of this fact. Intuitively, our interaction with this prover should satisfy three main properties. First, we want to be convinced of a true statement. Second, we do not want to be convinced of a false statement. Third, we would like to make a decision in polynomial time. This is the intuitive concept of *theorem-proving procedures* [Coo23], and the NP class constitutes a nice formalization of this intuition. Among all possible equivalent formulations, we report the one proposed in [AB09].

Definition 1.1.2 (NP)**.** A language $L \subseteq \{0,1\}^*$ is *nondeterministic polynomial* if there exist a polynomial $p : \mathbb{N} \longrightarrow \mathbb{N}$ and a polynomial-time Turing machine M such that for every input string $x \in \{0,1\}^*$

$$x \in L \iff \exists c \in \{0,1\}^{p(|x|)} : \mathrm{M}(x,c) = 1.$$

If $x \in L$ and $c \in \{0,1\}^{p(|x|)}$ satisfy $\mathrm{M}(x,c) = 1$, then we call c a *certificate* for x (with respect to the language L and machine M). The set of all nondeterministic polynomial languages is denoted with NP.

What the definition mimics is the presence of a second (possibly unbounded) Turing machine which can solve the decision problem, find a proof (i.e., a certificate) and send it in order to convince us that a given string is indeed in the language. Observe that the definition only requires a (polynomially verifiable) certificate to exist only for *yes* answers, while *no* instances (i.e., strings x such that $x \notin L$) might not be verifiable.

Remark 1.1.3. An alternative definition of NP highlights the meaning of the class name. NP can be seen as the set of problems that can be solved in polynomial time by a non-deterministic Turing machine (NTM). Informally, a non-deterministic Turing Machine is a Turing machine with two different, independent transition functions. At each step, it makes an arbitrary choice as to which function to apply, and every sequence of choices defines a possible computation of the machine. We say that a nondeterministic Turing machine accepts an input x if at least one computation (i.e., one of the possible arbitrary sequences of choices) terminates in an accepting state. This description is the basis for the abbreviation NP for *Nondeterministic Polynomial* languages. We refer the interested reader to [Sip96] for more details on NTMs.

Notice that P $\subseteq$ NP, as Definition 1.1.2 reduces to Definition 1.1.1 if we set $q(n) = 0$, i.e., we accept an empty certificate; languages in P are polynomially verifiable without need of certificates. On the other hand, the natural temptation is to assert that NP is strictly larger than P. Despite this, no formal proof of this has been found to date. Today, the question of whether P $\neq$ NP remains one of the most fundamental open problems in mathematics [Coo00]. In order to better assess the problem, we set up a hierarchy within NP in order to identify, if possible, languages that are harder than others [AB09].

Definition 1.1.4. Given two languages $L, L_0 \in$ NP, we say that L is polynomially reducible to L_0, and we write $L \leq_p L_0$, if there is a function $R : \{0,1\}^* \longrightarrow \{0,1\}^*$ such that

$$x \in L \iff R(x) \in L_0$$

and R halts in polynomial time w.r.t. $|x|$.

In other words, R maps strings in L to strings in L_0 and strings that are not in L to strings that are not in L_0. Note that we require R to be computable in polynomial time, i.e., there must be a polynomial p such that $R(x)$ is computed in at most $p(|x|)$ steps. If $L \leq_p L_0$, we say that L_0 is at least as hard as L. In fact, if we have a procedure to decide L_0, we can apply it to decide also L with *just* a polynomial overhead due to the reduction.

In [AHU74] it has been shown that the *satisfiability* problem, SAT for short, which belongs to NP, exhibits a remarkable property. Specifically, any decision problem in NP can be reduced to SAT. Consequently, while SAT might theoretically have a polynomial-time algorithm, such a discovery would imply that other well-known NP problems admit polynomial-time solutions. However, despite decades of effort, researchers have been unable to find polynomial-time algorithms for any NP problems. This strongly suggests that SAT does not have a polynomial-time algorithm, and thus that P $\neq$ NP. Furthermore, some years later Karp [Kar10] discovered some results that seemed to confirm this intuition, which is made explicit in his work:

> *"In this paper we give theorems that suggest, but do not imply, that these problems, as well as many others, will remain intractable perpetually."*
>
> R. Karp, 1972

More in detail, in [Kar10] it has been shown that SAT can itself be reduced in the sense described above to many other NP-problems. Thus if any of these NP-problems possesses a polynomial time algorithm, then so does every NP-problem, and hence NP=P. NP-problems with this property are called NP-complete problem, and are formalized as follows [AB09].

Definition 1.1.5 (NP-Complete Languages)**.** We say that a language $L \in$ NP is NP-complete if for every language $L' \in$ NP we have that $L' \leq_p L$.

Over the years, the list of NP-complete problems has gradually expanded: [GJ79a] contains hundreds of problems known to be NP-complete; and if NP$\neq$P is true then no NP-complete problem can have a polynomial-time algorithm.

The intuitive notion of theorem-proving procedure, formalized in a nice way with the NP class, can be further developed.

> *"The notion of proof, like the notion of computation, is an intuitive one. Intuition, however, may and must be formalized. [...] Each formalization, however, cannot entirely capture our original and intuitive notions, exactly because they are intuitive. Following our intuition, probabilistic algorithms are means of computing, though they are not in the previous formal model. Similarly, NP is an elegant formalization of the intuitive notion of a theorem proving procedure. However, NP only captures a particular way of communicating a proof."*
>
> S.Goldwasser, S.Micali, C.Rackoff, 1985

[GMR19] points out some limits of the NP class in grasping the intuitive concept of proof-system. In order to refine this concept, they introduce the IP class (which stands for Interactive Polynomial) as the class of decision problems efficiently solvable by an *interactive proof system* (*interactive proof* for short). An interactive proof system consists of two machines, a prover P, which presents a proof that a given string x is a member of some language, and a verifier V, which checks that the presented proof is correct. These two machines exchange a polynomial number of messages (w.r.t. $|x|$) and once the interaction is completed, the verifier must decide whether or not x is in the language. The formalization of this intuition, as given in [GMR19], is as follows.

Definition 1.1.6 (Binary Relation)**.** A *binary relation* is a finite set $R \subseteq X \times Y$, where $X, Y \subseteq \{0,1\}^*$. Given $(x, y) \in R$, we say that y is a *witness* for the *statement* x. The set $L_R = \{x \in X \mid \exists y \in Y \text{ s.t. } (x, y) \in R\}$ is called the set of *true statements* for R, or its *language*.

Definition 1.1.7 (Interactive Proof)**.** An *interactive proof* (P, V) for a binary relation $R \subseteq X \times Y$ is an interactive protocol between two probabilistic polynomial-time machines P and V. The *prover* P takes as input a pair $(x, y) \in R$ while the *verifier* V takes as input x. As the output of the protocol - denoted by $(\mathsf{P}(y), \mathsf{V})(x)$ - V either accepts (outputs 1) or rejects (outputs 0). We say that a *transcript*, i.e. the set of all messages exchanged in a protocol execution, is *accepting* (*rejecting*) if V accepts (rejects, respectively).

An interactive proof, as defined in [GMR19], is required to satisfy two properties.

Definition 1.1.8 (Completeness)**.** An interactive proof (P, V) for a binary relation $R \subseteq X \times Y$ is *complete* if, for every $(x, y) \in R$, we have

$$\Pr[(\mathsf{P}(y), \mathsf{V})(x) = 0] \leq \rho(x)$$

where the value $\rho(x)$ - called *completeness error* - is negligible (in $|x|$). If $\rho(x) = 0$ for all $x \in L_R$, the protocol is said to be *perfectly complete*.

Definition 1.1.9 (Soundness)**.** An interactive proof (P, V) for a binary relation $R \subseteq X \times Y$ is *sound* if, for every $x \notin L_R$ and a (potentially-dishonest) prover P^*, we have

$$\Pr[(\mathsf{P}^*, \mathsf{V})(x) = 1] \leq \sigma(x)$$

where the value $\sigma(x)$ - called *soundness error* - is negligible (in $|x|$).

Note that an interactive proof which satisfies both the previous properties allows a prover P to convince the verifier V that a statement x is true. Furthermore, it is immediate to verify that NP $\subseteq$ IP, as any language in NP can be decided by a very simple type of interactive proof system, where the prover comes up with the proof certificate and the verifier is a deterministic polynomial-time machine that checks it. It is complete because the certificate will make it accept if there is one, and it is sound because the verifier cannot accept if there is no certificate.

1.1.2 Interactive Proofs Systems and Cryptography

In the following, we limit the attention to a particular class of interactive proofs which are useful in cryptography. The first constraint that we have is that, within an execution of an interactive proof (P, V), the prover P always sends the first and the last message. Hence, the number of communication rounds is odd, i.e. of the form $2\mu+1$ with $\mu \in \mathbb{N}^*$. We refer to an interactive proof having $2\mu+1$ communication rounds with the name $(2\mu+1)$-*round interactive proof*. When $\mu = 1$, and thus the rounds are only 3, we call it *Sigma protocol*, and usually denote it with Σ. Another common constraint for an interactive proof is to be *public coin* [AF22b], in the sense of the following definition.

Definition 1.1.10 (Public-Coin)**.** An interactive proof (P, V) is *public-coin* if all V random choices are made public.

If an interactive proof is public-coin, the verifier needs to send to the prover only their random choices. For this reason, we call *challenges* the messages sent by the verifier and *challenge set* the set from which verifier's messages are sampled. In the case of a $(2\mu+1)-$round interactive proof, we define the challenge set Ch as the Cartesian product of μ *round challenge sets* $\mathsf{Ch}^{[i]}$, with $i \in \{1, \ldots, \mu\}$, meaning that the challenge for the i-th round is sampled from $\mathsf{Ch}^{[i]}$. We refer to Figure 1.1 for a graphical representation of a Sigma protocol workflow.

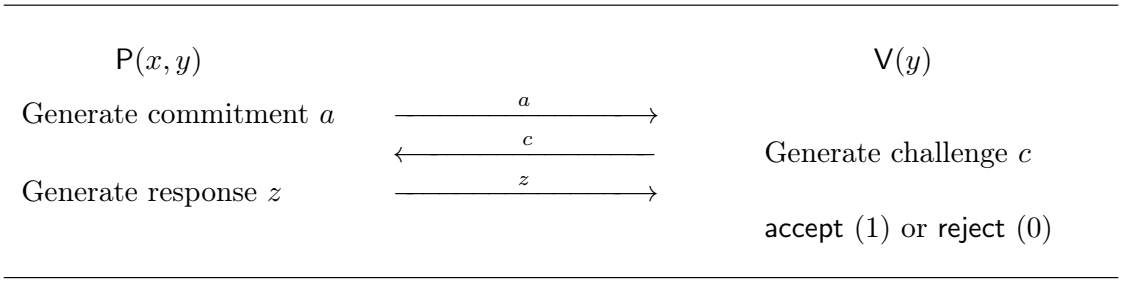

Figure 1.1: Sigma protocol workflow.

Sometimes in cryptography, given a language L and a statement $x \in L$, we would like a honest prover to be able to convince a verifier of the validity of the given statement without ever giving the verifier any other information except truthiness of the statement itself. Namely, $x \in L$. This is formalized by the so called zero-knowledge requirement.

The (Honest Verifier) Zero Knowledge Requirement

This property formalizes the fact that transcripts generated by the interaction between a prover and a honest verifier must be indistinguishable from those that could be generated by someone without the witness. The precise definition of this fact is given in the following [BS20].

Definition 1.1.11 (Honest Verifier Zero Knowledge)**.** Let (P, V) be an interactive protocol for $R \subseteq X \times Y$. We say that (P, V) is *honest verifier zero knowledge*, or HVZK, if there exists an efficient probabilistic algorithm Sim (called a simulator) such that for all possible pairs $(x, y) \in R$, the output distribution of Sim on input x is identical to the distribution of a transcript of a conversation between P (on input (x, y)) and V (on input x).

The term "zero knowledge" is meant to suggest that an adversary learns nothing from P , because an adversary can simulate conversations on his own (using the algorithm Sim),

without knowing y. The term "honest verifier" conveys the fact this simulation only works for conversations between P and the actual, "honest" verifier V , and not some arbitrary algorithm.

The Knowledge Soundness Requirement

An interactive proof which satisfies the previous properties allows a prover P to convince the verifier V that a statement x is true, without disclosing any other information except the validity of the statement. However, it does not guarantee anything about P's knowledge of a witness y such that $(x, y) \in R$. When an interactive proof is meant to allow a prover (not only that a statement is true, but also) to convince a verifier they know a witness, it is further required to be knowledge sound. Informally, this means that any dishonest prover who does not know a witness can only convince a verifier with some *small* probability κ, which is called the knowledge error. This property is formalized by requiring that there exists an efficient algorithm - the extractor - that, given oracle access to a dishonest prover who succeeds with probability $\varepsilon > \kappa$, outputs a witness with probability at least $\varepsilon - \kappa$ up to a multiplicative polynomial loss in the security parameter [AF22b].

Definition 1.1.12 (Knowledge Soundness)**.** An interactive proof (P, V) for a binary relation $R \subseteq X \times Y$ is *knowledge sound*, with *knowledge error* κ, if there exists an algorithm Ext that, given as input any $x \in X$ and rewindable oracle access to a (potentially-dishonest) prover P^*, runs in an expected polynomial time (in $|x|$) and outputs a witness $y \in Y$ for x with probability

$$\Pr[(x, \mathsf{Ext}^{\mathsf{P}^*}(x)) \in R] \geq \frac{\varepsilon(x, \mathsf{P}^*) - \kappa(x)}{\mathsf{poly}(|x|)},$$

where $\varepsilon(x, \mathsf{P}^*) = \Pr[(\mathsf{P}^*, \mathsf{V})(x) = 1]$. The algorithm Ext is called *knowledge extractor*.

Remark 1.1.13. To simplify the subsequent analysis, where not otherwise specified, we will assume that the prover P^* is deterministic. In fact, it is possible to show that the extractor is well-defined even when restricted to deterministic provers [AF22a]. More precisely, suppose that P^* is a probabilistic prover, and denote by $\mathsf{P}^*[r]$ the deterministic prover obtained by setting the randomness of P^* to r. Then, it is easy to show that $\varepsilon(x, \mathsf{P}^*) = \mathbb{E}[\varepsilon(x, \mathsf{P}^*[r])]$ and $\Pr[(x, \mathsf{Ext}^{\mathsf{P}^*}(x)) \in R] = \mathbb{E}\Big[\Pr[(x, \mathsf{Ext}^{\mathsf{P}^*[r]}(x)) \in R]\Big]$, where the expected value is taken over the random choice of r.

Definition 1.1.14 (Proof of Knowledge)**.** An interactive proof (P, V) for a binary relation $R \subseteq X \times Y$ which satisfies both completeness with completeness error ρ and knowledge soundness with knowledge error κ is a *proof of knowledge* if there exists a positive-definite polynomial p over the integers such that $1 - \rho(x) \geq \kappa(x) + \frac{1}{p(|x|)}$ for all $x \in X$.

A common and easier strategy to prove the knowledge soundness of a public-coin interactive proof is showing a stronger property, called special soundness [HL10; ACK21b; AF22b], which informally means that there exists an extracting algorithm able to compute a witness given enough accepting transcripts relative to a true statement x. In particular, a 3-round interactive proof is 2-special-sound if there exists an efficient algorithm that, given two valid transcripts (a, c, z) and (a, c', z') relative to the same statement x and with distinct second messages (challenges) $c \neq c'$, outputs a witness for x. This property can be generalized both in the necessary transcripts and in the number of rounds, leading to the notion of $(k_1, \ldots, k_\mu)$-special soundness for $(2\mu + 1)$-round interactive proofs. Notice that this coincides with the standard 2-special-soundness notion when $\mu = 1$ and $k_1 = 2$. In order to formalize this concept, we need to firstly introduce the notion of tree of transcripts.

Definition 1.1.15 (Tree of Transcripts)**.** Let $k_1, \ldots, k_\mu, N_1, \ldots, N_\mu \in \mathbb{N}^*$, $R \subseteq X \times Y$ be a binary relation and (P, V) a $(2\mu+1)-$round public-coin interactive proof for R, where V samples $i-$th challenges ($i \in \{1, \ldots, \mu\}$) from a set $\mathsf{Ch}^{[i]}$ of cardinality $N_i \geq k_i$. A $(k_1, \ldots, k_\mu)$-*tree of transcripts* for (P, V) is a set of $K = \prod_{i=1}^{\mu} k_i$ transcripts relative to a given statement $x \in X$, arranged in the following tree structure, where nodes correspond to prover's messages while edges to verifier's challenges. From every node at level i, with $i \in \{1, \ldots, \mu\}$, exactly k_i edges originate, corresponding to k_i pairwise-distinct challenges belonging to $\mathsf{Ch}^{[i]}$. Then, each of the K transcripts corresponds to exactly one path from the root node to a leaf node.

A graphical representation of a tree of transcripts is provided in Figure 1.2, where $a^{[1]}$ denotes the prover's first message, $c_1^{[1]}, \ldots, c_{k_1}^{[1]}$ are sampled from $\mathsf{Ch}^{[1]}$, and so on. As above, we refer the interested reader to [ACK21b] and [AF22b] for further details.

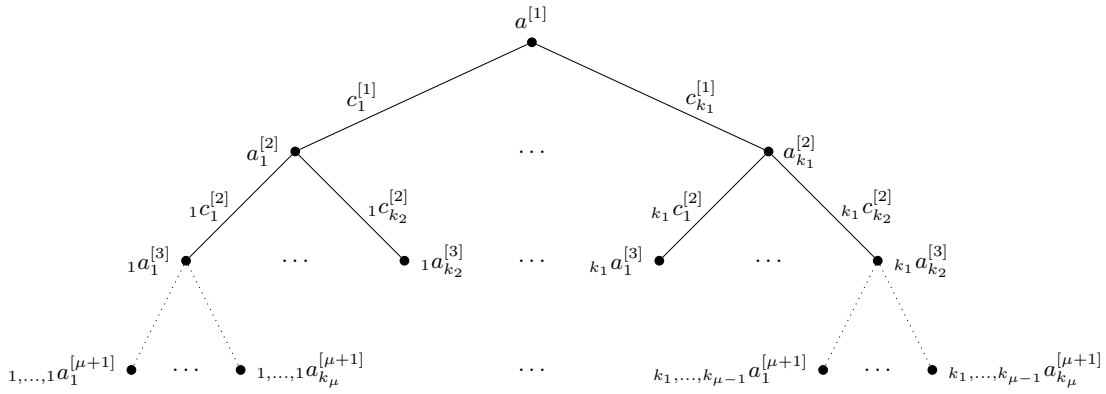

Figure 1.2: Graphical representation of a $(k_1, \ldots, k_\mu)-$tree of transcripts for a $(2\mu+1)-$round public-coin interactive proof. Left subscripts represent the ancestor nodes, superscripts represent the corresponding round, while right subscripts are used to enumerate edges originating from a node and their corresponding arrival nodes.

Definition 1.1.16 ($(k_1, \ldots, k_\mu)$-Special Soundness)**.** Let $k_1, \ldots, k_\mu$, $N_1, \ldots, N_\mu \in \mathbb{N}^*$ and $R \subseteq X \times Y$ be a binary relation. A $(2\mu+1)$-round public-coin interactive proof (P, V) for R, where V samples $i-$th challenges ($i \in \{1, \ldots, \mu\}$) from a set $\mathsf{Ch}^{[i]}$ of cardinality $N_i \geq k_i$, is $(k_1, \ldots, k_\mu)$-*out-of-*$(N_1, \ldots, N_\mu)$ *special sound*, or simply $(k_1, \ldots, k_\mu)$-*special sound*, if there exists a polynomial-time algorithm that, on input a true statement $x \in X$ and a $(k_1, \ldots, k_\mu)$-tree of accepting transcripts for (P, V) and relative to x, outputs a witness $y \in Y$ for x.

In the case of a Sigma protocol, it is immediate to prove that k-out-of-N-special soundness implies knowledge soundness with knowledge error $(k-1)/N$ [HL10; ACK21a]. The general $(2\mu+1)-$round case is much more involved, and it has only recently been shown [ACK21b] that $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$-special soundness tightly implies knowledge soundness, with knowledge error

$$\kappa = 1 - \prod_{i=1}^{\mu} \frac{(N_i - k_i + 1)}{N_i}.$$

Within the family of interactive protocols that satisfy the correctness, knowledge soundness and honest-verifier zero-knowledge properties, we are interested in those that have a

negligible knowledge-error. More specifically, with an eye toward cryptographic applications, given a security level λ we would like a dishonest prover to be able to make the verifier accept with probability at most $2^{-\lambda}$. Some interactive protocols, like the one proposed by Schnorr [Sch91], have a challenge space whose cardinality can be set equal to 2^{λ}. As a consequence, their knowledge error is $\kappa = 1/|\mathsf{Ch}| = 2^{-\lambda}$ and therefore they already possess this property. Despite these cases, the common scenario is another one [Ste90; Ste93; Vér97; CVA10; DW22; Tan+22; Bia+20; Bal+23a], where the challenge space is smaller and consequently the knowledge error κ is bigger (with the usual case being 1/2 or 1/3). Although this fact would seem to exclude these protocols from cryptographic applications, it is still possible to use them as building blocks of other interactive protocols, by repeating the basic units in parallel t times, i.e. the prover and the verifier run t parallel executions of the protocol and the verifier accepts if the resulting t transcripts are accepting. We denote by $(\mathsf{P}^t, \mathsf{V}^t)$ the t-fold parallel repetition of (P, V). While this technique has been broadly adopted, it was only in 2022 that Attema and Fehr [AF22a] proved that the t-fold parallel repetition of any $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round public-coin interactive proof optimally reduces the knowledge error from κ down to κ^t.

1.2 Digital Signatures

Informally, a digital signature scheme is a protocol conducted between two parties, a prover and a verifier. The prover is equipped with a pair $(\mathsf{sk}, \mathsf{pk})$ of secret and public key, respectively, while the verifier is only given pk. The flow of the protocol is as follows: the prover chooses a message m, and creates a signature σ using his secret key sk and the message m, getting a signed message (m, σ). The verifier reads (m, σ) and uses the public key pk to check that (with some probability) the sender is legit, and the message has not been altered. Formally, we provide the following definition [BS20].

Definition 1.2.1 (Signature Scheme)**.** A digital signature scheme $\mathcal{S}$ is a triple of algorithms $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vf})$ which are defined as follows:

- The key generation algorithm KeyGen is a probabilistic algorithm which, given as input 1^{λ}, where λ is the security parameter, outputs a pair of matching public and private keys $(\mathsf{pk}, \mathsf{sk})$;
- The signing algorithm Sign is probabilistic and takes as input a message $m \in \{0,1\}^*$ to be signed and returns a signature $\sigma = \mathsf{Sign}^{\mathsf{sk}}(m)$;
- The verification algorithm takes as input a message m and a signature σ. It returns $\mathsf{Vf}^{\mathsf{pk}}(m, \sigma)$ which is 1 if the signature is accepted and 0 otherwise. It is required that $\mathsf{Vf}^{\mathsf{pk}}(m, \sigma) = 1$ if $\sigma = \mathsf{Sign}^{\mathsf{sk}}(m)$.

This last condition described above directly translates into the correctness of the scheme. In this case, in fact, every signature produced honestly, following first the key generation algorithm KeyGen and then the signature algorithm Sign, will be such that the verification algorithm Vf will return 1 as output. However, correctness is not the only desirable property of a signature scheme. In particular, we would like a signature scheme to satisfy some notion of "security".

In the case of digital signature schemes, the security is modeled using a game between a challenger and an adversary $\mathcal{A}$, which is usually a a polynomial-time probabilistic Turing machine. The game models a possible scenario where $\mathcal{A}$ tries to break $\mathcal{S}$ using an attack

when the challenger is using the scheme $\mathcal{S}$. The security of a scheme is determined by two parameters, the kind of attack we are allowing and the kind of information we want to protect [GMR88]. One of these attack framework is the well known Existential Forgery (EUF) under Chosen-Message Attack (CMA), which is one of the strongest requirement when proving the security of a digital signature scheme. In this context, the adversary is allowed to adaptively ask for different signatures, and at the end it can forge any fresh message. We report the formalization proposed in [DAST19].

Definition 1.2.2 (EUF-CMA Security)**.** Let $\mathcal{S}$ be a signature scheme. A forger $\mathcal{A}$ is a $(t, q_{\text{hash}}, q_{\text{sign}}, \varepsilon)$-adversary in EUF-CMA against $\mathcal{S}$ if after at most q_{hash} queries to the hash oracle, q_{sign} signatures queries and t working time, it outputs a valid forgery with probability at least ε. We define the EUF-CMA success probability against $\mathcal{S}$ as:

$$Succ_S^{\text{EUF-CMA}}(t, q_{\text{hash}}, q_{\text{sign}}) := \max\{\varepsilon : \text{there exists a } (t, q_{\text{hash}}, q_{\text{sign}}, \varepsilon) \text{ adversary}\}.$$

The signature scheme S is said to be $(t, q_{\text{hash}}, q_{\text{sign}})$-secure in the EUF-CMA model if the above success probability is a negligible function of the security parameter λ. Losing a bit of precision, we normally content ourselves saying that $\mathcal{S}$ is EUF-CMA secure, meaning that for every probabilistic polynomial time $(t, q_{\text{hash}}, q_{\text{sign}}, \varepsilon)$-adversary $\mathcal{A}$, ε is negligible in the security parameter λ.

Normally, when showing the EUF-CMA security of a digital signature scheme, any hash functions involved in the scheme is simply modeled as a random oracle [BR93]. In this case we will talk about security in the Random Oracle Model (ROM). In the specific case in which the adversary is modeled in such a way as to be able to execute quantum algorithms, we will talk about security in the Quantum Random Oracle Model (QROM). The interested reader is referred to [Bon+11] for more details on this.

1.2.1 Digital Signatures From Interactive Proof Systems

It is possible to turn interactive proofs into non-interactive ones [Fis05; FS87a; Unr15; Pas03], and eventually link a message to the proof in order to create digital signatures. Following [BS20], we provide a formalization of this intuition with the definition below.

Definition 1.2.3 (Non-Interactive Proof)**.** A *non-interactive proof* for relation R is a pair (P, V) of (probabilistic) algorithms, a prover P and a polynomial time verifier V, such that (1) given $(x, y) \in R$, the prover $\mathsf{P}(x, y)$ outputs a proof π, and (2) given $x \in \{0,1\}^*$, a purported proof π, the verifier $\mathsf{V}(x, \pi)$ outputs 0 to reject or 1 to accept the proof.

The most common case is when both the prover and the verifier are given additional access to a random oracle. As for interactive definitions, a non-interactive proof is complete if honestly generated proofs for $(x, y) \in R$ are accepted by V with high probability. It is sound if it is infeasible to produce an accepting proof for a false statement. In the non-interactive setting, the soundness error, i.e., the success probability of a cheating prover depends on the number of queries it is allowed to make to the random oracle. The same holds true for knowledge soundness of non-interactive proofs. For a precise definition of the knowledge soundness for non-interactive proofs, we refer to [AFK22a]. We would like to have non-interactive proofs which inherits the main security properties (in the random oracle model) of the interactive version. A common way to achieve this is by the Fiat-Shamir transform [FS87c], which allows to turn any public-coin interactive proof into a non-interactive one. We start by presenting the transform for the specific case of Sigma protocols, showing how to modify it to obtain an EUF-CMA secure signature scheme.

Then, we generalize this construction to an interactive proof system with $(2\mu+1)$-rounds. The rough idea is to replace the random challenges, which are provided by the verifier in the interactive version, by the hash of the current message. In the following, we report the definition proposed in [BS20], adapting the terminology to make it consistent with our discussion.

Definition 1.2.4 (Fiat-Shamir Transform)**.** Let $\Sigma = (\mathsf{P}, \mathsf{V})$ be a Sigma protocol, where the challenge is sampled from set Ch, and let $H : \{0,1\}^* \longrightarrow \mathsf{Ch}$ be a hash function. The Fiat-Shamir (FS) transformation $\mathrm{FS}(\Sigma) = (\mathsf{P}_{\mathrm{fs}}, \mathsf{V}_{\mathrm{fs}})$ is the non-interactive proof where $\mathsf{P}_{\mathrm{fs}}(x, y)$ runs $\mathsf{P}(x, y)$ but instead of asking the verifier for the challenge c on message a, the challenge is computed as $c = H(x, a)$, and the response z is computed accordingly; the output is then the proof $\pi = (a, z)$. On input a statement x and a proof $\pi = (a, z)$, $\mathsf{V}_{\mathrm{fs}}(x, \pi)$ accepts if, for c as above V accepts the transcript (a, c, z) on input x.

By a small adjustment, where also the to-be-signed message is included in the hashes, the transformation turns any public-coin interactive proof into a signature scheme. The building blocks can be informally described as follows:

- a Sigma protocol $\Sigma = (\mathsf{P}, \mathsf{V})$ for a relation $R \subseteq X \times Y$;
- a key generation algorithm KeyGen for R;
- a hash function $H : \{0,1\}^* \longrightarrow \mathsf{Ch}$.

The Fiat-Shamir signature scheme derived from KeyGen and (P, V) works as follows:

- the key generation algorithm is KeyGen, so a public key is of the form $\mathsf{pk} = x$, where $x \in X$, and a secret key is of the form $\mathsf{sk} = (x, y) \in R$;
- to sign a message m using a secret key $\mathsf{sk} = (x, y)$, the signing algorithm runs as follows:
  - it starts the prover $\mathsf{P}(x, y)$, obtaining a commitment a;
  - it computes a challenge $c \longleftarrow H(m, a)$;
  - finally, it feeds c to the prover, obtaining a response z, and outputs the signature $\sigma := (z, a)$.
- to verify a signature $\sigma = (z, a)$ on a message m using a public key $\mathsf{pk} = x$, the verification algorithm computes $c \longleftarrow H(m, a)$, and checks that (a, c, z) is an accepting conversation for x.

For 3-round interactive protocols which satisfy the knowledge soundness and the honest-verifier zero-knowledge property, the Fiat-Shamir transformation offer a simple and nice way to construct digital signatures which satisfies the EUF-CMA security requirement [KMP16].

Recently, some new signature schemes have been proposed [Bal+20; Hül+16], starting from 5-step interactive protocols, which has made it necessary to study the transform also in this more generic context. The Fiat-Shamir transform can be easily generalized to the multi-round case. In this new context, it has been recently shown that the two aforementioned signature schemes satisfy the EUF-CMA security requirement [Hül+16].

Although this ensures that an attacker needs exponential time to forge a signature, this characterization is too coarse to determine the precise parameters needed for instantiating

a signature scheme. To address this, it is crucial to place greater emphasis on the security reduction. Unfortunately, in general the soundness error of the Fiat-Shamir transformed protocol degrades exponentially in the number of rounds [AFK22a]. Concretely, for any $(2\mu+1)$-move interactive proof that admits a cheating probability of at most ε, captured by the knowledge or soundness error, the Fiat-Shamir-transformed protocol admits a cheating probability of at most $(Q+1)^{\mu} \cdot \varepsilon$, where Q denotes the number of random-oracle queries admitted to the dishonest prover [AFK22a]. At the same time, this fact raises the question whether other classes of interactive proofs feature a milder security loss. These are still open problems in research, and only some partial results have been given. For the specific class of $(k_1, \ldots, k_\mu)$-special-sound protocols (which cover a broad class of use cases), the knowledge error degrades linearly in Q, instead of Q^μ, but this does not happen for the t-fold parallel repetitions of $(k_1, \ldots, k_\mu)$-special-sound protocols, when $t \geq \mu$. Both of these results are proven in [AFK22a]. In particular, in the latter case there is an attack that results in a non-negligible security loss [KZ20], which affects every signature based on this framework, such as [Bal+21a] and [Hül+16].

1.3 Linear Codes

We set out below the most relevant facts regarding linear codes. We start by stating some elementary results, after which we analyze the main NP-complete problems related to coding theory. With this new knowledge, we will be able to provide results regarding code-based digital signature schemes, which we will state at the end of the section. We refer to [WGR22] and [Bal14] for a more detailed description.

1.3.1 Elementary Results

Definition 1.3.1 (Linear Code)**.** Let $k \leq n$ be positive integers. Then, an $[n, k]$ *linear code* C over the field $\mathbb{F}_q$ is a k dimensional subspace of $\mathbb{F}_q^n$.

The parameter n is called the *length* of the code, the elements in the code are called the *codewords* and the ratio $R = k/n$ is called the *rate* of the code. In this work, we adopt the well-established notation of considering codewords as row vectors, and thus any vector v is treated as a row vector, so that its transpose $v^\top$ is a column vector. When we refer to a code C or to an $[n, k]$ code C without specifying anything else, we will always refer to an $[n, k]$ linear code. Since an $[n, k]$ code C is a linear subspace of $\mathbb{F}_q^n$, we can completely describe it using one of the two matrices defined below.

Definition 1.3.2 (Generator Matrix)**.** Let $k \leq n$ be positive integers and let C be an $[n, k]$ linear code over $\mathbb{F}_q$. Then, a matrix $\mathbf{G} \in \mathbb{F}_q^{k \times n}$ is called a *generator matrix* for C if the rows of $\mathbf{G}$ form a basis of C. Formally,

$$C = \left\{ x\mathbf{G} \mid x \in \mathbb{F}_q^k \right\}.$$

Definition 1.3.3 (Parity-Check Matrix)**.** Let $k \leq n$ be positive integers and let C be an $[n, k]$ linear code over $\mathbb{F}_q$. Then, a matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k) \times n}$ is called a *parity-check matrix* for C if

$$C = \left\{ y \in \mathbb{F}_q^n \mid \mathbf{H}y^\top = 0 \right\}.$$

Given a vector $v \in \mathbb{F}_q^n$, we define the syndrome of v with respect to $\mathbf{H}$ as the vector given by $\mathbf{H}v^\top$. Notice that all vectors in the same coset $v + C$ possess the same syndrome.

According to this, a vector $v \in \mathbb{F}_q^n$ belongs to the code C if and only if its syndrome (w.r.t. $\mathbf{H}$) is the null vector.

Definition 1.3.4 (Dual Code)**.** Let $k \leq n$ be positive integers and let C be an $[n, k]$ linear code over $\mathbb{F}_q$. The *dual code* of C, denoted with $C^\perp$, is the $[n, n-k]$ linear code over $\mathbb{F}_q$, defined as

$$C^\perp = \left\{ x \in \mathbb{F}_q^n \mid \sum_{i=1}^{n} x_i y_i = 0 \quad \forall y \in C \right\}.$$

Let $k \leq n$ be positive integers and C be an $[n, k]$ linear code over $\mathbb{F}_q$ with generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$. Then, for some permutation matrix $\mathbf{P}$ and some invertible matrix $\mathbf{U}$, the systematic form of the generator matrix $\mathbf{G}$ is $\mathbf{UGP} = (\mathbf{I}_k \ \mathbf{A})$, where $\mathbf{A} \in \mathbb{F}_q^{k \times (n-k)}$. Similarly, for some permutation matrix $\mathbf{P'}$ and some invertible matrix $\mathbf{U'}$ the systematic form of the parity-check matrix $\mathbf{H}$ is $\mathbf{U'HP'} = (\mathbf{B}\ \mathbf{I}_{n-k})$, where $\mathbf{B} \in \mathbb{F}_q^{(n-k) \times k}$. Whenever a generator matrix or a parity-check matrix will have this particular shape, we will say that the matrix is in *Systematic Form*.

In order to measure how far apart two vectors are, we endow $\mathbb{F}_q$ with a metric, which is generally induced by a weight function.

Definition 1.3.5. A *weight* over $\mathbb{F}_q$ is a function $\mathrm{w} \colon \mathbb{F}_q \to \mathbb{N}$ satisfying $\mathrm{w}(0) = 0$, $\mathrm{w}(x) > 0$ for all $x \neq 0$, $\mathrm{w}(x) = \mathrm{w}(-x)$ for all x, and $\mathrm{w}(x + y) \leq \mathrm{w}(x) + \mathrm{w}(y)$ for all $x, y \in \mathbb{F}_q$.

In coding theory, one of the most classical and important examples of weights is the Hamming weight, introduced by Hamming in 1950 for codes over finite fields [Ham50].

Definition 1.3.6 (Hamming weight)**.** Given a finite field $\mathbb{F}_q$, the *Hamming weight* of an element $a \in \mathbb{F}_q$ is given by

$$\mathrm{w}(a) \coloneqq \begin{cases} 0 & \text{if } a = 0, \\ 1 & \text{otherwise .} \end{cases}$$

We define the Hamming weight of an n-tuple $x \in \mathbb{F}_q^n$ additively by

$$\mathrm{w}(x) := \sum_{i=1}^{n} \mathrm{w}(x_i) \ .$$

A weight function induces a *distance* defined as $\mathrm{d} \colon \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{N}$, where $\mathrm{d}(x, y) := \mathrm{w}(x - y)$. The *Hamming support* of a vector $x \in \mathbb{F}_q^n$ is defined as the set of coordinates where x is non-zero. Namely, $\mathrm{Supp}(x) \coloneqq \{1 \leq j \leq n \mid x_j \neq 0\}$. Note that the Hamming weight of a vector $x \in \mathbb{F}_q^n$ is equal to the cardinality of its support, that is, $\mathrm{w}(x) = |\mathrm{Supp}(x)|$.

Remark 1.3.7. In this thesis we will only work with the Hamming metric but, as an alternative to this classical metric, we could also endow the ambient space with other metrics. A notable example of additive distance is the Lee distance, first proposed in [Lee58] as an extension of the Hamming metric for the binary field. It has recently garnered increasing attention in code-based cryptography [Bar+21b; BKW22; HTW20; Weg+24]. Another metric that has gained significant attention is the rank metric, which, unlike the Hamming and Lee metrics, is not additive. Rank-metric codes over finite fields were first studied in connection with association schemes by Delsarte in 1978 [Del78]. They were also independently introduced by Gabidulin in [Gab85], where rank-metric codes are described as $\mathbb{F}_q$-linear spaces of vectors over an extension field. In other words, codewords are matrices and the distance between two codewords is the rank of their difference.

Once a metric is defined, one can also consider the minimum distance of a code, i.e., the smallest distance achieved by two distinct codewords.

Definition 1.3.8 (Minimum distance)**.** Let C be a linear code over $\mathbb{F}_q$. The *minimum Hamming distance* of C is denoted by $d(C)$ and is given by

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

Thanks to the linearity of C, it is immediate to verify that $d(C) = \min\{\mathrm{w}(x) \mid x \in C \setminus \{0\}\}$. The minimum distance of a code C is one of its most important parameters, as it is related to its error detection and correction capabilities, which measure the number of words that a code is able to detect and correct, respectively. More in detail, we can define the (closed) ball of center x and radius t in $\mathbb{F}_q^n$ as

$$B(\mathbb{F}_q, x, n, t) := \{y \in \mathbb{F}_q^n \mid \mathrm{d}(x, y) \leq t\}.$$

When clear from the context, we will simplify the notation saying $B_t(x)$. Given $B_t(x)$, we say that a code can detect t errors if for all $c \in C$, $B_t(c) \cap C = \emptyset$. Similarly, we say that a code can correct t errors if, for all $c_1, c_2 \in C$ it holds that $B_t(c_1) \cap B_t(c_2) = \emptyset$. In particular, the following result holds.

Proposition 1.3.9 ([MS77], Theorem 2)**.** *Let C be a linear code with minimum distance d. Then:*

- *C can detect $d - 1$ errors.*
- *C can correct $\lfloor (d-1)/2 \rfloor$ errors.*

Applications normally require the correction of an error not only to be possible, but also computationally feasible. This is the reason why we are interested in studying codes with both a high minimum distance d and for which efficient decoding algorithms exist to correct up to t errors, for some $t \leq \lfloor (d-1)/2 \rfloor$.

The Gilbert-Varshamov bound

In coding theory, the Gilbert-Varshamov (GV) bound is a fundamental result that provides a lower bound on the maximum size of a code, given its length and minimum distance. We provide an overview for codes equipped with the Hamming metric. Note that the (closed) ball of center x and radius t, namely $B(\mathbb{F}_q, x, n, t)$, has the same size for any center x. Thus, in the following, we will not specify the center of the ball and we will denote with

$$B(\mathbb{F}_q, n, t) \coloneqq |\{y \in \mathbb{F}_q^n \mid \mathrm{d}(0, y) \leq t\}|$$

the volume of the (closed) ball of radius t in $\mathbb{F}_q^n$. Finally, we will denote by $A(\mathbb{F}_q, n, d)$ the maximum number of codewords of a code in $\mathbb{F}_q^n$ with minimum Hamming distance d.

Theorem 1.3.10. (Gilbert-Varshamov bound for the Hamming metric, [Rot06]) *For a positive integer n, assume $\mathbb{F}_q^n$ is equipped with the Hamming distance. The maximal size of a code in $\mathbb{F}_q^n$ having minimum distance d is*

$$A(\mathbb{F}_q, n, d) \geq \frac{q^n}{B(\mathbb{F}_q, n, d-1)} \ . \tag{1.1}$$

If C is a linear code in $\mathbb{F}_q^n$ equipped with the Hamming distance, we will say that C *lies on the GV bound* if

$$|C| - 1 < \frac{q^n}{B(\mathbb{F}_q, n, d-1)} \leq |C| \ .$$

In the following, we will specify the volume of the n-dimensional balls for the Hamming metric.

Proposition 1.3.11 (Volume of a Hamming-metric ball, [Rot06])**.** *Given n, w positive integers, the volume of the n-dimensional Hamming ball of radius w in $\mathbb{F}_q^n$ is*

$$B(\mathbb{F}_q, n, w) = \sum_{i=0}^{w} \binom{n}{i} (q-1)^i.$$

In its asymptotic form, the GV bound offers a lower limit on the rate of the code, rather than its cardinality. Let δ be the relative distance of the code, that is $d = \delta N$. We have that

$$\lim_{n\to\infty} \frac{1}{n} \log_q(A(\mathbb{F}_q, n, d)) \geq 1 - \lim_{n\to\infty} \frac{1}{n} \log_q(B(\mathbb{F}_q, n, d)). \tag{1.2}$$

It has been proven that the limit in the right-hand side of the previous equation exists and equals the q-ary entropy function h_q evaluated on δ.

In the Hamming metric over finite fields, it is well known that random linear codes asymptotically achieve the GV bound [BF02; Pie67]. This means that we can asymptotically estimate the minimum distance of a random code with good probability: denoting with R the left-hand side of Eq. (1.2), we have that $R \approx 1 - h_q(\delta)$, so that $\delta \approx h_q^{-1}(1-R)$.

1.3.2 Hard Problems on Codes

Two important computationally-hard algebraic problems arising in coding theory are the *Syndrome Decoding Problem* (SDP) and the *Codeword Finding Problem* (CFP), as stated in [BMVT78]. As we will see better at the end of this section, several code-based digital signature schemes are built upon SDP, and an adversary who wants to break such schemes is usually left with the only option of solving these problems. In addition to these classic problems that we describe, there are many others, for example DOOM (Decoding One Out of Many) and WE (Weight Enumerator). We will also encounter these problems during the course of this thesis. However, since their use will be limited to few specific contexts, we avoid reporting their definitions below, in order to leave the description as clean as possible. We state the SDP and CFP in the following.

Definition 1.3.12 (Syndrome Decoding Problem)**.** Let $\mathbb{F}_q$ be a finite field and let $k \leq n$ be positive integers. The *Syndrome Decoding Problem* (SDP) is the problem defined as follows:

Input: $(\mathbf{H}, y, \omega)$, where $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, $y \in \mathbb{F}_q^{n-k}$ is a vector, and $\omega \in \mathbb{N}$.

Output: $x \in \mathbb{F}_q^n$ such that $\mathbf{H}x^\top = y^\top$ and $\mathrm{w}(x) \leq \omega$.

The idea behind the SDP is that if we choose ω small enough, it is hard to find a vector with weight smaller than ω such that its syndrome is equal to y.

Definition 1.3.13 (Codeword Finding Problem)**.** Let $\mathbb{F}_q$ be a finite field and let $k \leq n$ be positive integers. The *Codeword Finding Problem* (CFP) is the problem defined as follows:

Input: $(\mathbf{H}, \omega)$, where $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ and $\omega \in \mathbb{N}$.

Output: $x \in \mathbb{F}_q^n$ such that $\mathbf{H}x^\top = 0$ and $\mathrm{w}(x) = \omega$.

The idea behind the CFP is that if ω is small enough it is hard to find a codeword of the code C with parity-check matrix $\mathbf{H}$ such that its weight is ω, i.e. it is hard to find codeword with small weight in a given linear code.

Our aim is to study the hardness of both the SDP and the CFP. A well known result in computational complexity theory [AB09] states that any decision problem belonging to the NP-Complete class has a search-to-decision reduction, i.e. it is possible to solve an instance of the search problem with a polynomial amount of calls to an oracle for the corresponding decision problem. The decisional version of the SDP is the Coset Weights Problem (CWP) and the decisional version of the CFP is the Subspace Weights Problem (SWP). Below, we give a more precise definition of these problems.

Definition 1.3.14 (Coset Weights Problem)**.** The Coset Weights Problem is the problem defined as follows:

Input: $\mathbf{H} \in \mathbb{F}_q^{n\times(n-k)}$, a vector $y \in \mathbb{F}_q^{n-k}$ and a non-negative integer ω.

Output: decide whether there exists a vector $x \in \mathbb{F}_q^n$ s.t. $\mathbf{H}x^\top = y^\top$ and $\mathrm{w}(x) \leq \omega$.

Definition 1.3.15 (Subspace Weights Problem)**.** The Subspace Weight Problem is the problem defined as follows:

Input: $\mathbf{H} \in \mathbb{F}_q^{n\times(n-k)}$ and a non-negative integer ω.

Output: decide whether there exists a vector $x \in \mathbb{F}_q^n$ s.t. $\mathbf{H}x^\top = 0$ and $\mathrm{w}(x) = \omega$.

In our case, if the Coset Weights Problem and the Subspace Weights Problem are NP-complete, then we would have automatically studied the hardness of both the SDP and the CFP and they would be as difficult as an NP-complete problem. Berlekamp, McEliece and van Tilborg famously proved in [BMVT78] the NP-completeness of these two problems for the case of binary linear codes equipped with the Hamming metric. In [Bar94], Barg generalized this proof to an arbitrary finite field.

Remark 1.3.16. In the following, we make the standard abuse of notation of saying that the Syndrome Decoding Problem and the Codeword Finding Problems are NP-complete, always intending that (the associated decisional versions of) the Syndrome Decoding Problem and the Codeword Finding Problems are NP-complete.

Regarding these two problems, it is worth reporting a consideration made by their authors.

> *"It would be very desirable to replace the phrase "of weight w" in Subspace Weight with the phrase "of weight $\leq w$", for this would show that the problem of finding the minimum weight of a general code is NP-complete. While we conjecture that this is the case, we have not been able to prove it, and propose it as a research problem."*
>
> E.R. Berlekamp, R.J. McEliece, H.C.A. van Tilborg, 1978

The problem they are referring to with these words is the following.

Definition 1.3.17 (Minimum Distance)**.** The Minimum Distance Problem is the problem defined as follows:

Input: $\mathbf{H} \in \mathbb{F}_2^{n\times(n-k)}$ and a non-negative integer ω.

Output: decide whether there exists a vector $x \in \mathbb{F}_2^n \backslash \{0\}$ s.t. $\mathbf{H}x^\top = 0$ and $\mathrm{w}(x) \leq \omega$.

The difficulty of computing the minimum distance was later established by Vardy in 1997 [Var97].

1.3.3 Generic Solvers

To date, two primary methods are recognized for decoding random linear codes: Information Set Decoding Algorithms (ISD) [Pra62; Ste89; LB88; MMT11; Bec+12] and the Generalized Birthday Algorithm (GBA) [WGR22]. ISD algorithms are particularly effective when the decoding problem yields only a small number of solutions, while GBA performs better in cases with numerous solutions. Other techniques, such as statistical decoding [Jab01], gradient decoding [AB98], and supercode decoding [BKT99], have been introduced but remain less competitive compared to ISD algorithms.

In the following, we focus on the leading approach for solving the Syndrome Decoding Problem: ISD algorithms. In particular, we slightly modify the standard ISD paradigm, to take into account the fact that we expect the algorithm to find more words than the desired weight. We first consider a high level and general description, which encompasses all algorithms of this family. This model will turn useful later with this thesis, when this type of algorithms is employed to estimate the weight distribution of a linear code. We refer to [WGR22] for a standard treatment of these algorithms. In this paper, we are mostly interested in random codes or, at the very least, codes whose structural properties cannot be used to somehow ease the search for codewords. To formalize this situation, we make use of the following assumption, which is standard and folklore for the study of ISD algorithms.

Assumption 1 (Randomness of codewords)**.** For a given code $C \subseteq \mathbb{F}_q^n$, let $\widetilde{C}_w \subseteq C_w$ such that, for any two distinct $c, c' \in \widetilde{C}_w$, it holds $c \neq ac'$ for any $a \in \mathbb{F}_q^*$. Then, for any integer $w \in \{0, \ldots, n\}$ and any codeword $c \in \widetilde{C}_w$, we assume that the support of c is uniformly random over all subsets of $\{1, \ldots, n\}$ of size w.

Theorem 1.3.18. *Let $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ be a random matrix with coefficients over $\mathbb{F}_q$. Furthermore, let $s \in \mathbb{F}_q^{n-k}$ and $w \in \mathbb{N}$. Then,*

$$\mathbb{E}\left[\left|\left\{x \in \mathbb{F}_q^n \ : \ \mathbf{H}x^\top = s^\top \wedge \mathrm{w}(x) = w\right\}\right|\right] = \frac{\binom{n}{w}(q-1)^w}{q^{n-k}}, \tag{1.3}$$

where the probability is taken over the random choice of $x \in \mathbb{F}_q^n$, considering the latter as a probability space with the uniform distribution.

Proof. Suppose we are given a random element $x \in \mathbb{F}_q$ with Hamming weight w. Since there are q^{n-k} syndromes and $\mathbf{H}$ is random, every syndrome is equally probable, and $\mathbb{P}(\mathbf{H}x^\top = s^\top) = 1/(q^{n-k})$. Since the number of vectors of length n and Hamming weight w is given by $\binom{n}{w}(q-1)^w$, the expected number of these vectors that will have syndrome equal to s will be given by $\binom{n}{w}(q-1)^w/q^{n-k}$. □

For big values of n we can estimate the volume of any Hamming Ball of fixed radius using the q-ary entropy function, so that, considering Eq. (1.3), we can say that the number of w-weight codewords of C is given by

$$N_C(w) \approx q^{\left(h_q(w/n)-(1-R)\right)n}.$$

Infomation Set Decoding Algorithms

By ISD, we refer to a randomized algorithm

$$\begin{aligned} \mathrm{ISD}\colon \mathbb{F}_q^{r\times n} \times \{0,\dots,n\} &\longrightarrow \mathcal{P}(C_w), \\ (\mathbf{H}, w) &\longmapsto X. \end{aligned}$$

The algorithm receives as input a description for the code (say, a parity-check matrix $\mathbf{H}$) and the desired weight w, and returns an element X in the powerset of C_w, i.e. a set of codewords with weight w. All ISD algorithms share a common procedure which is highlighted in Algorithm 1. In particular, the operations performed by any ISD algorithm can be divided into three main steps:

- *Partial Gaussian Elimination* (lines 1-4): a random permutation π of length n is sampled (line 2 in the algorithm). Then, Partial Gaussian Elimination (PGE) with parameter $\ell \in \mathbb{N}$, $0 \le \ell \le n-k$ is performed. In other words, one checks whether it is possible to apply a change of basis on the permuted parity-check matrix $\pi(\mathbf{H}) \in \mathbb{F}_q^{r\times n}$, so that a matrix with the following structure is obtained

$$\begin{pmatrix} \mathbf{A} & \mathbf{0}_{\ell\times(n-k-\ell)} \\ \mathbf{B} & \mathbf{I}_{n-k-\ell} \end{pmatrix},$$

  where $\mathbf{A} \in \mathbb{F}_q^{\ell\times(k+\ell)}$ and $\mathbf{B} \in \mathbb{F}_q^{(n-k-\ell)\times(k+\ell)}$. Notice that such a matrix is not guaranteed to exist: indeed, if the rightmost $n-k-\ell$ columns of $\pi(\mathbf{H})$ form a matrix whose rank is less than $n-k-\ell$, then PGE cannot be performed. In these cases, a new permutation is sampled.

- *Solving the small instance* (line 5): because of the PGE decomposition, we have

$$c = (c', c'') \in \pi(C) \iff \begin{cases} \mathbf{A}c'^{\top} = 0, \\ \mathbf{B}c'^{\top} + c''^{\top} = 0. \end{cases} \tag{1.4}$$

  Notice that c' has length $k+\ell$ and is, de facto, a codeword of the code whose parity-check matrix is $\mathbf{A}$. One restricts the search for c' by requiring that it has some (low) weight and some specific weight partition.

- *Producing solutions* (lines 6–1): once c' has been found, one can easily compute the associated c'' from the second row of the linear system in (1.4). In other words, we produce codewords of the form (c', c'') and check whether they have the desired weight w: any such codeword corresponds to the permutation of a codeword in C_w.

This is a very general way to study ISD algorithms, but allows to identify the main quantities we will use for our analysis. For what concerns the time complexity of each iteration, we can use the following estimate.

Algorithm 1 ISD operating principle

Input : $\mathbf{H} \in \mathbb{F}_q^{r \times n}$, $w \in \mathbb{N}$.
Output : set $Y \subseteq C_w$

// Apply random permutation and do PGE
repeat
  Sample $\pi \xleftarrow{\$} S_n$
  Apply PGE on $\pi(\mathbf{H})$
until PGE is successful
// Solve small instance
$X = \mathsf{Solve}(\mathbf{A}, \ell)$
// Test codewords associated to solutions for the small instance
$Y \leftarrow \varnothing$
for $c' \in X$ **do**
  Compute $c'' = -c'\mathbf{B}^\top$
  if $\mathrm{wt}(c') + \mathrm{wt}(c'') = w$ **then**
    Update $Y \leftarrow Y \cup \left\{\pi^{-1}((c', c''))\right\}$
  end if
end for
return Y

Proposition 1.3.19 (Cost of one iteration)**.** *On average, one iteration of ISD uses a number of elementary operations (sums and multiplications) over $\mathbb{F}_q$ counted by*

$$O\left(\frac{t_{\mathsf{PGE}}}{p_{\mathsf{inv}}} + t_{\mathsf{Solve}} + \widehat{|X|}\right),$$

where $p_{\mathsf{inv}} = \prod_{i=\ell+1}^{n-k}(1-q^{-i})$ is the probability that a random $(n-k) \times (n-k-\ell)$ matrix has rank $n-k-\ell$, t_{PGE} and t_{Solve} are the costs for PGE and of the subroutine Solve*, respectively, while $\widehat{|X|}$ is the average number of solutions which are found, for the small instance. Asymptotically, the cost is $O\left(t_{\mathsf{Solve}}\right)$.*

Proof. Performing PGE requires a number of operations which grows as $O(n^3)$, so it is a polynomial in n. The number of times we need to repeated the PGE step, on average, corresponds to the reciprocal of the probability that the chosen permutation π places, on the rightmost side of $\pi(\mathbf{H})$, $n-k-\ell$ columns which form a basis for a space with dimension $n-k-\ell$. Assuming that all columns of $\mathbf{H}$ behave as random vectors over $\mathbb{F}_q$, with length $n-k$, we get that this probability is lower bounded by a constant, since

$$\begin{aligned}
p_{\mathsf{inv}}(\ell) &= \prod_{i=0}^{n-k-\ell-1}\left(1 - \frac{q^i}{q^{n-k}}\right) \\
&= \prod_{i=0}^{n-k-\ell-1}\left(1 - q^{-(n-k-i)}\right) \\
&= \prod_{i=\ell+1}^{n-k}\left(1 - q^{-i}\right) \\
&\geq \prod_{i=1}^{n-k}\left(1 - q^{-i}\right) \geq 1 - 1/q + 1/q^2.
\end{aligned}$$

The operations in lines 8–10 take polynomial time, and are repeated, on average, for $\widehat{|X|}$ times. To prove the formula for the asymptotic cost, we first consider that $t_{\mathsf{PGE}}/p_{\mathsf{inv}}$ is a polynomial in n. Moreover, if Solve returns $|X|$ solutions, then its cost cannot be lower than $|X|$. Since on average the subroutine returns $\widehat{|X|}$ solutions, it must be $t_{\mathsf{Solve}} \geq \widehat{|X|}$: consequently, asymptotically, the cost of Algorithm 1 is dominated by t_{Solve}. □

Notice that, at this stage, we have provided all the necessary details apart from those of the subroutine Solve. Yet, its functioning is crucial to determine the computational cost of an ISD algorithm. For the moment, we keep it as a very general procedure and consider that it will only return the codewords in $\pi(C_w)$ that satisfy some constrains, e.g., some specific weight partition. We denote by $f_{\mathrm{ISD}} : \mathbb{F}_q^{k+\ell} \longrightarrow \{0,1\}$ such a constraint and assume that, whenever $f_{\mathrm{ISD}}(c')$ is equal to 1, c' is among the outputs of the subroutine.

Proposition 1.3.20. *Let $p_{ISD}(w)$ denote the probability that, for a uniformly random permutation π and a codeword $c \in C_w$, $\pi(c)$ satisfies the constrains imposed by f_{ISD}. Then, for a code with C with $N_C(w)$ codewords of weight w and under Assumption 1, the probability that a given permutation π is successful for one iteration of ISD is given by*

$$p^*_{ISD}(w) = 1 - \left(1 - p_{ISD}(w)\right)^{N_C(w)}. \tag{1.5}$$

Proof. By definition:

$$p^*_{\mathrm{ISD}}(w) = 1 - \mathbb{P}\left[f_{\mathrm{ISD}}\big(\pi(c)\big) = 0, \forall c \in C_w\right].$$

As a consequence of Assumption 1, we have that

$$\begin{aligned}
\mathbb{P}\left[f_{\mathrm{ISD}}\big(\pi(c)\big) = 0,\ \ \forall c \in C_w\right] &= \left(\mathbb{P}\left[f_{\mathrm{ISD}}\big(\pi(c)\big) = 0,\ c \xleftarrow{\$} C_w\right]\right)^{|C_w|} \\
&= \left(1 - \mathbb{P}\left[f_{\mathrm{ISD}}\big(\pi(c)\big) = 1,\ c \xleftarrow{\$} C_w\right]\right)^{N_C(w)} \\
&= \left(1 - p_{\mathrm{ISD}}(w)\right)^{N_C(w)},
\end{aligned}$$

from which the result follows. □

Notice that, on average, each call to Algorithm 1 returns $\widehat{|Y|} = N_C(w) \cdot p_{\mathrm{ISD}}(w)$ codewords. We finally consider repeated calls to Algorithm 1, and derive the probability that the algorithm returns something.

Proposition 1.3.21. *Asymptotically, the cost of ISD is $O\left(t_{\mathsf{Solve}}/p^*_{ISD}(w)\right)$.*

Proof. The probability that a call to Algorithm 1 returns a non empty set Y is $p^*_{\mathrm{ISD}}(w)$. So, on average, $1/p^*_{\mathrm{ISD}}(w)$ calls are required before at least one of the codewords in C_w is returned. □

Two very interesting ISD variants are those proposed by Lee and Brickell [LB88], and by Stern [Ste89]. Notice that, at least for the binary case, more advanced algorithms exist (e.g., MMT [MMT11] and BJMM [Bec+12]). However, they have space complexity (i.e., amount of used memory) which is typically much larger than that of Stern. Also, they do not have a non binary counterpart: so, to avoid distinguishing between the binary and non binary cases, we will omit them from our analysis. Below we report the main results for Lee&Brickell and Stern. Crucial quantities in our analysis will be the success probability, namely, the probability that a given permutation π is successful for an ISD algorithm. In particular, we show that, as long as we do not consider a specific code, the probability that a chosen permutation is valid will only be a function of (i) the weight w, and (ii) the constraints which are imposed by the considered ISD variant.

Lee & Brickell's ISD

In Lee & Brickell algorithm (LB for short) [LB88] the tacit assumption is that there exists at least one codeword with p non null components in a given information set I. We describe the algorithm taking into account its three main building blocks, namely the partial Gaussian elimination part, the resolution of the small instance, and the finalization phase. The Gaussian elimination is performed with $\ell = 0$ and can be considered a degenerate case of the general procedure. In the LB case, once this first phase is finished, the structure of the resulting matrix will be as follows:

$$\mathbf{H}' = \begin{pmatrix} \mathbf{B} & \mathbf{I}_{n-k} \end{pmatrix}, \tag{1.6}$$

where $\mathbf{B} \in \mathbb{F}_q^{(n-k)\times k}$. Given this, solving the small instance and producing the solution is straightforward: LB assumes the existence of a codeword of weight w with p nonzero components in the first k positions of the permuted code described by $\mathbf{H}'$. Let c be such a codeword. We have that $\pi(c) = (c', c'')$, with $\mathrm{w}(c') = p$ and $\mathrm{w}(c'') = w - p$, so that

$$\begin{aligned} c\mathbf{H}^\top = 0 &\iff \pi\left(c\mathbf{H}^\top\right) = 0 \\ &\iff \pi(c)\pi(\mathbf{H}^\top) = 0 \\ &\iff \left(c', c''\right)\left(\mathbf{B} \mid \mathbf{I}\right)^\top = 0 \\ &\iff c'\mathbf{B}^\top + c'' = 0. \end{aligned}$$

Therefore, it is sufficient to enumerate all the vectors c' of length k and weight p, and check whether $c'' = -c'\mathbf{B}^\top$ has weight $w - p$. The LB subroutine is depicted in Algorithm 2.

Algorithm 2 Lee & Brickell Solve subroutine

Input : $p \in \mathbb{N}$.
Output : set X made up of all vectors with length k and weight p

1: Set $X \leftarrow \left\{x' \in \mathbb{F}_q^k \mid x' \in V_{k,p}\right\}$ // Enumerate candidates for x'
2: **return** X

Notice that the complexity of solving this subroutine is simply given by the number of vectors of length k and weight p, which is given by $t_{\mathsf{Solve}}(p) = \binom{k}{p}(q-1)^p$. Similarly, the probability that, given a codeword of weight w, its permuted vector has exactly p non-null components in the first k entries and $w - p$ non-null components in the remaining $n - k$, is given by $p_{\mathrm{ISD}}(w) = \binom{k}{p}\binom{n-k}{w-p}/\binom{n}{w}$. Overall, the following result holds.

Proposition 1.3.22 (Performances of Lee&Brickell's ISD)**.** *The time complexity of the Lee&Brickell* Solve *subroutine, with parameter $p \in \mathbb{N}$, $0 \le p \le \min\{w, k\}$, is*

$$t_{\mathsf{Solve}}(p) = \binom{k}{p}(q-1)^p.$$

The probability that a codeword $c \in C_w$ is returned is

$$p_{\mathsf{ISD}}(w) = \frac{\binom{k}{p}\binom{n-k}{w-p}}{\binom{n}{w}}.$$

Stern's ISD

The main assumption on which this algorithm [Ste89] relies is the following: it assumes that there exist weight w codewords in $\pi(C)$ with $2p$ non-null entries in the first $k+\ell$ positions (respectively divided into two blocks of equal length $(k+\ell)/2$ and weight p) and $w-2p$ non-null entries in the rightmost $n-k-\ell$ positions. In particular, the algorithm aims to find the leftmost vector using the standard collision search technique (sometimes also called meet-in-the-middle): it creates two lists L_1 and L_2 through the enumeration of $V_{(\frac{k+\ell}{2}),p}$, together with their partial syndromes. The Stern's subroutine is described in Algorithm 3.

Algorithm 3 Stern Solve subroutine

Data : $p \in \mathbb{N}$, $p \leq \left\lfloor \frac{k+\ell}{2} \right\rfloor$
Input : $\mathbf{E} \in \mathbb{F}_q^{\ell \times (k+\ell)}$, $\ell \in \mathbb{N}$.
Output : set X with solutions of the small instance, with weight $2p$ equally partitioned

// Partition $\mathbf{E}$
Write $\mathbf{E} = (\mathbf{E}', \mathbf{E}'')$, where $\mathbf{E}' \in \mathbb{F}_q^{\ell \times \left\lfloor \frac{k+\ell}{2} \right\rfloor}$, $\mathbf{E}'' \in \mathbb{F}_q^{\ell \times \left\lceil \frac{k+\ell}{2} \right\rceil}$
Set $L_1 \leftarrow \left\{ (x', x'\mathbf{E}'^{\top}) \mid x' \in V_{(\frac{k+\ell}{2}),p} \right\}$
// Enumerate candidates for x' and x''
Set $L_2 \leftarrow \left\{ (x'', -x''\mathbf{E}''^{\top}) \mid x'' \in V_{(\frac{k+\ell}{2}),p} \right\}$
// Find collisions (using efficient strategy, e.g., sorting plus binary search)
Compute X, the set of all pairs $(x', x'') \in V_{(\frac{k+\ell}{2}),p} \times V_{(\frac{k+\ell}{2}),p}$ such that $x'\mathbf{E}'^{\top} = -x''\mathbf{E}''^{\top}$
return X

As it is well known, the merge can be efficiently computed using a sort plus binary search approach, taking time

$$O\big(\max\{|L_1| \cdot \log_2(|L_1|), |L_2| \cdot \log_2(|L_2|)\}\big).$$

Notice that the lists have sizes given by

$$|L_1| = \binom{\left\lfloor \frac{k+\ell}{2} \right\rfloor}{p} (q-1)^p,$$

$$|L_2| = \binom{\left\lceil \frac{k+\ell}{2} \right\rceil}{p} (q-1)^p.$$

Getting rid of the logarithmic factor, and taking into consideration that the resulting list C needs to be somehow allocated, we can consider that the overall cost of merging the two lists is given by

$$O\big(\max\{|L_1|, |L_2|, |C|\}\big).$$

When L_1 and L_2 are formed by elements without any relevant structure, we can safely consider that each pair of elements in L_1 and L_2 results in a collisions with probability $q^{-\ell}$. This is a frequently employed heuristic, which corresponds to the assumption that each entry in the associated syndromes is uniformly distributed over $\mathbb{F}_q$. In such a case, we can set

$$|C| = |L_1| \cdot |L_2| \cdot q^{-\ell}$$

For the sake of simplicity, we can neglect the floors and ceiling. This way, we have $|L_1| = |L_2| = L = \binom{\frac{k+\ell}{2}}{p}(q-1)^p$ and $|C| = L^2 q^{-\ell}$. This way, the cost of the overall subroutine Solve becomes

$$t_{\mathsf{Solve}} = L\left(2 + Lq^{-\ell}\right).$$

Similarly, the probability that, given a codeword of weight w, its permuted vector has exactly $2p$ non-null components in the first $k+\ell$ entries, distributed as described above, and $w-2p$ non null components in the remaining $n-k-\ell$ positions, is given by

$$p_{\mathsf{ISD}}(w) = \frac{\binom{(k+\ell)/2}{p}^2 \binom{n-k-\ell}{w-2p}}{\binom{n}{w}}.$$

In particular, the following result holds.

Proposition 1.3.23 (Performances of Stern's ISD)**.** *The time complexity of the Stern's* Solve *subroutine, with parameters $p, \ell \in \mathbb{N}$, where $0 \leq p \leq \left\lfloor \frac{k+\ell}{2} \right\rfloor$, $0 \leq \ell \leq n-k$ and such that $2p \leq w \leq n-k-\ell+2p$ is*

$$t_{\mathsf{Solve}}(p,\ell) = L^2/q^\ell + L,$$

where $L = \binom{\frac{k+\ell}{2}}{p}(q-1)^p$. The probability that a codeword $c \in C_w$ is returned is

$$p_{\mathsf{ISD}}(w) = \frac{\binom{(k+\ell)/2}{p}^2 \binom{n-k-\ell}{w-2p}}{\binom{n}{w}}. \tag{1.7}$$

As a final remark, we recall that ISD algorithms should not be considered as attacks in the classical sense. Instead, as they define the state-of-the-art regarding random decoding, they determine the choice of parameters for a given protocol, once fixed a desired security level.

1.3.4 Code-Based Digital Signature Schemes

Code-based digital signature schemes can be divided into two variants. On the one hand, there are schemes based on proofs of knowledge, which in turn can follow the Fiat-Shamir [FS87b] or the Schnorr-Lyubashevsky [Lyu09] approach, while on the other side there are schemes following the hash&sign paradigm.

The schemes following the first approach follow a work by Goldwasser, Micali and Rackoff [GMR19] which dates back to 1985, where the concept of zero-knowledge proof is introduced for the first time. Following this work, Fiat and Shamir built the first protocol [FS87b] that exploits this kind of proofs. From here, the zero-knowledge protocols have been declined in various versions, including that of the codes. The first one who worked in this direction was Stern, who proposed the first draft of a code-based Sigma protocol [Ste90], and a more practical version [Ste01] some years later. Several variants of this scheme have been proposed over the years: in 1997 Véron introduced a dual version of the scheme proposed by Stern [Vér97], and in 2010, together with other researchers, they proposed a scheme [CVA10] in which the instantiated code lies in $\mathbb{F}_q$ and no more in a binary field. However, we have to wait until 2020 to have some truly competitive ZKID code-based digital signature proposals [Bia+20]. Recently, the development of these protocols is going in some new directions, for example trying to improve these schemes with the "cut and choose" technique [GPS22; FJR23], or combining the theory of multiparty computation [FJR23; FJR22], or other techniques [**CROSS**], such as the seed tree, the multiple public key, and the fixed weight ones. We refer the interested reader to [Bor+23] for further details.

Differently from this first family, digital signature schemes which follow a hash&sign approach have had a more tumultuous development during the last years. Among the schemes, CFS [CFS01; Dal07] and KKS [KKS97] were the first to be introduced. The idea behind both schemes is the same: the digest of a message to be signed is considered as a corrupted codeword (more precisely, it is considered as a syndrome of a corrupted codeword) $c + e$ for a certain codeword $c \in C$ and a small-weight error e, and the owner of the private key is the only one capable of decoding, i.e. capable of finding e. The main difference between CFS and KKS is that in the first the digest is considered as a random syndrome, leading to an inefficient scheme, while the latter manipulates the digest to output a decodable syndrome, leading to a very efficient but insecure scheme [OT11]. In order to mitigate the efficiency issues of CFS, some authors have proposed CFS-like schemes in which the hash function is substituted with a map whose output is a syndrome with small-weight coset-leader, mixing some ideas behind both CFS and KKS. This approach has been adopted e.g. in [RZW+17], where the authors utilize a hash function based on the works of Augot, Finiasz and Sendrier [AFS05] and on the Merkle-Damgard construction [Dam89; Mer89]. This promising approach has however been proved to be insecure in [DMP21]. An important thing to notice is that in order for these schemes to work it is necessary that the underlying code has a particular structure which make the decoding phase efficient. For example, the CFS scheme uses high rated Goppa codes. This usually leads to a big problem regarding the security assumptions, indeed on one hand we assume the difficulty of solving SDP, but on the other one we also have to assume the indistinguishability of the underlying structured codes from a random linear one. This second assumption is much more delicate, and often turns out to be false. In this regard, a distinguisher for high-rated Goppa codes has been build recently [Fau+13], making the CFS scheme formally unsecure. In 2014 it is the time of RankSign, where the main idea is to leave the Hamming metric and work with the rank one, using augmented LRPC codes. However, even in this case there has been a structural attack [DAT18]. During the last years there have been different attempts to build similar digital signature schemes [Bal+13; Per18; Gab+14; Kim+22; Rit+23; FRX+17; Son+20; LXY20; PMT22] but it seems that the cryptographic community is far from achieving a trustable proposal [PT16; SBC19; Xag18; DG20; HW24; Ara+21; Bal+21b; PT23]. In this context, a remarkable hash&sign code based digital signature scheme that still remains unbroken is Wave [DAST19], which has been proposed in 2019 and relies on the indistinguishability of the normalized generalized $(U, U+V)$ codes.

Chapter 2

Construction

Regarding the evolution of code-based digital signature schemes following the hash&sign paradigm, the framework that we have outlined at the end of the previous chapter is quite turbulent. A question that naturally arises in this context is the following.

Is there a way to build a code-based Hash&Sign
digital signature scheme that is both secure and efficient?

In this chapter, we try to approach this question: we propose a post-quantum code-based digital signature algorithm whose security is based on the difficulty of decoding Quasi-Cyclic (QC) codes in systematic form, and whose trapdoor relies on the knowledge of a hidden Quasi-Cyclic Low-Density-Parity-Check (QC-LDPC) code. QC codes allows us to balance between security and key size, while the LDPC property lighten the encoding complexity, thus the signing algorithm complexity, significantly. Unfortunately, we are unable to answer this question in the affirmative: recently, the scheme we proposed suffered a critical attack that strongly limits its usefulness. We describe the rise and fall of this scheme in the following section.

2.1 A Post-Quantum DSA from QC-LDPC Codes

We start by recalling some elementary results on QC-LDPC codes, then moving on to expose our proposed signature scheme, discussing the general idea, its correctness, the parameters' choice, its security, and some experimental results. We then describe the attack we suffered, and finally state some conclusive remarks.

2.1.1 Basics on QC-LDPC Codes

Below we define Low-Density Parity-Check codes as well as Quasi-Cyclic codes. The purpose of this subsection is to establish the notation and state the computational problems from which we tried to build our scheme. For a more detailed treatment on this subject, the interested reader is referred to [Bal14].

LDPC Codes

Low-Density Parity-Check codes were first introduced by Gallager [Gal62] in the early 1960s and rediscovered by MacKay [MN97] in 1996. The idea of LDPC codes is to have a parity-check matrix that is sparse.

Definition 2.1.1 (LDPC Codes)**.** A code C is called *Low-Density Parity-Check* (LDPC) if it admits a sparse parity-check matrix $\mathbf{H}$. Informally, $\mathbf{H}$ is considered sparse if, for each row, the number of non-zero elements is logarithmic with respect to the length of the row and, for each column, the number of non-zero elements is logarithmic with respect to the length of the column.

In addition to describing error-correcting codes based on Low-Density Parity-Check matrices, Gallager also presented several practical decoding algorithms [Gal62]. One of these is called the Bit-Flipping algorithm, which corrects errors by iteratively flipping bits that contribute most to unsatisfied parity checks. The algorithm operates as follows:

1. The syndrome is computed by checking which parity-check equations are satisfied. Each parity-check equation corresponds to a row in the parity-check matrix, and an unsatisfied equation indicates an error in one or more of its associated bits.

2. For each iteration, bits contributing to the highest number of unsatisfied parity checks are identified as the most likely to be in error. These bits are "flipped" (i.e., their values are inverted) to correct potential errors.

3. The process is repeated iteratively until all parity-check equations are satisfied (indicating successful decoding) or until a maximum number of iterations is reached. In the later case we have a decoding failure.

According to this, LDPC codes are interesting from a cryptographic stand point, as they have no algebraic structure which might be detected by an attacker, but nevertheless have efficient decoding algorithms.

QC Codes

A cyclic code is a block code, where the circular shifts of each codeword gives another word that belongs to the code. Quasi-Cyclic codes are a generalization of this family of codes, as the following definition points out.

Definition 2.1.2 (Quasi-Cyclic Codes)**.** View a vector $c = (c_0, \ldots, c_{s-1})$ of $\mathbb{F}_2^{sn}$ as successive blocks of n-uples. An $[sn, k]$ linear code C is *Quasi-Cyclic* (QC) of index s if, for any $c = (c_0, \ldots, c_{s-1}) \in C$, the vector obtained after applying a simultaneous circular shift to every block $c_0, \ldots, c_{s-1}$ is also a codeword.

In the following, we present a useful object to better understand this family of codes.

Definition 2.1.3 (Circulant matrix)**.** Let $a = (a_0, \ldots, a_{n-1}) \in \mathbb{F}_2^n$. The *circulant matrix* induced by a is defined as follows:

$$\mathrm{circ}(a) := \begin{pmatrix} a_0 & a_{n-1} & \cdots & a_1 \\ a_1 & a_0 & \cdots & a_2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & \cdots & a_0 \end{pmatrix} \in \mathbb{F}_2^{n \times n}.$$

Consider the quotient ring $R := \mathbb{F}_2[x]/(x^n - 1)$ of all polynomials over $\mathbb{F}_2$ of degree less than n. Given a polynomial $a(x) \in R$ we denote by $a \in \mathbb{F}_2^n$ the vector whose coordinates are the coefficients of $a(x)$, namely, if $a(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$ then $a = (a_0, a_1, \ldots, a_{n-1})$. Due to this bijection, with some abuse of notation, we will denote in the same way (say a) both an element $a(x) \in R$ and its associated vector $a \in \mathbb{F}_2^n$. The

meaning of the underlying object will always be clear from the context. Consider two elements a and b in R. Their product $c \in R$ is the polynomial $a \cdot b$ and c is obtained by the formula $c^\top = \mathrm{circ}(a) \cdot b^\top$, where $\mathrm{circ}(a)$ is the $n \times n$ circulant matrix obtained from a. Notice that

$$a \cdot b = a \cdot \mathrm{circ}(b)^\top = \left(\mathrm{circ}(b) \cdot a^\top\right)^\top = b \cdot \mathrm{circ}(a)^\top = b \cdot a.$$

In this new context, multiplying an element $a \in R$ by x is equivalent to performing a circular shift of a by one position to the right. According to this fact, the characterization of Quasi-Cyclic codes as polynomials over R allows us to provide an alternative definition for this family of codes.

Definition 2.1.4 (Quasi-Cyclic Codes - alternative)**.** View a vector $c = (c_0, \ldots, c_{s-1})$ of $\mathbb{F}_2^{sn}$ as a successive blocks of n-uples, and consider each c_i as a polynomial in R. An $[sn, k]$ code C is *Quasi-Cyclic* (QC) of index s if for any $c = (c_0, \ldots, c_{s-1}) \in C$ it holds that $(x \cdot c_0, \ldots, x \cdot c_{s-1}) \in C$.

In the following, we will mainly work with Quasi-Cyclic codes in systematic form, for which we provide the relative definition below. Note that arbitrary QC-codes are not necessarily equivalent to a systematic QC-code.

Definition 2.1.5 (Systematic Quasi-Cyclic Codes)**.** A *systematic Quasi-Cyclic* $[sn, n]$ code of index s and rate $1/s$ is a quasi-cyclic code which admits a $(s-1)n \times sn$ parity-check matrix of the form:

$$\mathbf{H} := \begin{pmatrix} \mathbf{I}_n & 0 & \cdots & 0 & \mathbf{A}_0 \\ 0 & \mathbf{I}_n & \cdots & 0 & \mathbf{A}_1 \\ \vdots & \vdots & \ddots & 0 & \vdots \\ 0 & 0 & \cdots & \mathbf{I}_n & \mathbf{A}_{s-2} \end{pmatrix} \in \mathbb{F}_2^{(s-1)n \times sn}, \tag{2.1}$$

where $\mathbf{A}_0, \ldots, \mathbf{A}_{s-2}$ are $n \times n$ circulant matrices.

In the following, we work with 2-Quasi-Cyclic codes (2-QCC) in systematic form over $\mathbb{F}_2$, which are a particular family of binary codes of length $2n$ and dimension n. For these codes, the parity-check matrix is of the form $(\mathbf{I}_n \mid \mathrm{circ}(h))$ for some vector $h \in \mathbb{F}_2^n$. In particular, our scheme exploits the properties of 2-QCC whose parity-check matrices are sparse. That is, 2-Quasi-Cyclic Low-Density Parity Check codes. In the NIST Post-Quantum standardization process there were proposals for encryption schemes (e.g. BIKE [Ara+17], LEDAcrypt [Bal+19] and HQC [Mel+18a]) based on the same coding problems on which we base our protocol, but to date there are still no proposals for a signature scheme which exploits these ideas. Using QC codes allows us to have smaller keys because our parity-check matrix, which is public of dimension $n \times 2n$, can be described using only n bits, while the LDPC property, on which our trapdoor relies, is important from an implementation point of view, as the computation using sparse matrices speeds up the signing algorithm significantly.

Since we do not work with random-like linear codes, when we describe the security of the scheme we will refer to the following problems.

Definition 2.1.6 (s-QCCFP)**.** The s-Quasi-Cyclic Codeword Finding Problem (s-QCCFP) is the computational problem which takes as input a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(sn-n) \times sn}$ of a systematic QC code C of index s, a list of non-negative integers $w_0, \ldots, w_{s-1}$, and asks to find $x = (x_0, \ldots, x_{s-1}) \in \mathbb{F}_2^{sn}$ such that $\mathbf{H}x^\top = 0$, with $\mathrm{w}(x_i) = w_i$ for every $i \in \{0, \ldots, s-1\}$.

Definition 2.1.7 (s-QCSDP)**.** The s-Quasi-Cyclic Syndrome Decoding Problem (s-QCSDP) is the computational problem which takes as input a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(sn-n)\times sn}$ of a systematic QC code C of index s, a vector $y \in \mathbb{F}_2^{sn-n}$, a list of non-negative integers $w_0, \ldots, w_{s-1}$, and asks to find $x = (x_0, \ldots, x_{s-1}) \in \mathbb{F}_2^{sn}$ such that $\mathbf{H}x^\top = y^\top$, with $\mathrm{w}(x_i) \leq w_i$ for every $i \in \{0, \ldots, s-1\}$.

It is still under investigation whether these problems are difficult, but the cryptographic community seems to agree in this direction, as there are well-established schemes whose security is based on this assumption [GG07; Mel+18b]. Although these problems are considered difficult, we must still take into account the best solvers that currently exist to solve a given instance. In this regard, the best known attacks are still exponential, but it is possible to exploit the cyclic structure in order to have a polynomial speed-up factor compared to an attack we could mount on a random code. In the following, we derive the speed-up factor. Suppose that C is a QC code with parity check matrix $\mathbf{H}$ defined as in Eq. (2.1). If we find a solution vector $c = (c_0, \ldots, c_{s-1})$ such that $\mathbf{H}c^\top = 0$ and $\mathrm{w}(c_i) = w_i$, where c_i are vectors of length n, then we can circularly shift each c_i by the same number of elements and find another vector that is still a solution of our problem. To be precise, if we define the shift function as

$$\begin{aligned} \mathrm{sh}\colon \quad \mathbb{F}_2^n &\longrightarrow \mathbb{F}_2^n \\ (v_0, \ldots, v_{n-1}) &\longmapsto (v_{n-1}, v_0, \ldots, v_{n-2}) \end{aligned},$$

and $c = (c_0, \ldots, c_{s-1})$ is a solution of the CFP, then $(\mathrm{sh}(c_0), \ldots, \mathrm{sh}(c_{s-1}))$ is still a solution. According to this, we conclude that if at least one solution exists, then there are at least n solutions, where n is the size of the circulant matrices. In each iteration of ISD, the probability of success is increased by a factor of n, so the algorithm speeds up by the same factor. Similar reasoning is used for the SDP. These problems share a lot of similarities with the Decoding One Out of Many (DOOM) problem [Sen11], which can be stated in its full generality as follows.

Definition 2.1.8 (DOOM Problem)**.** The Decoding One Out of Many Problem takes as input a triple $(\mathbf{H}, S, w)$, where $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$ is a parity check matrix of some code, S is a set of syndromes and w is a non-negative integer. The problem asks us to find x such that $\mathrm{w}(x) \leq w$ and $\mathbf{H}x^\top \in S$.

In fact, solving the SDP with QC codes has the same complexity as solving the DOOM problem. For the sake of simplicity, we consider the case in which the QC code used has index 2. Let $\mathbf{H} \in \mathbb{F}_2^{n\times 2n}$ be a parity check matrix of some systematic QC code of index 2, so that $\mathbf{H} = (\mathbf{I}_n \mid \mathbf{A})$, where $\mathbf{A}$ is an $n \times n$ circulant matrix. Let s be the syndrome. Our problem asks us to find a vector $x = (x_0, x_1)$ such that $\mathbf{H}x^\top = s^\top$ and x respects the constraint on the weight. Notice that if we denote $\mathbf{H}' = (\mathbf{I}_n \mid \mathbf{A} \mid \mathbf{I}_n)$, then we have $\mathbf{H}'(x_0, x_1, s)^\top = 0$. It is straightforward to note that $\mathbf{H}'(\mathrm{sh}(x_0), \mathrm{sh}(x_1), \mathrm{sh}(s))^\top = 0$, which means that (x_0, x_1) is a solution for the SDP with syndrome s if and only if $(\mathrm{sh}(x_0), \mathrm{sh}(x_1))$ is a solution for the SDP with syndrome $\mathrm{sh}(s)$. Thus, if we are able to find a solution $x' = (x'_0, x'_1)$ for SDP with syndrome $\mathrm{sh}(s)$ then we can easily find the solution for SDP with syndrome s. We can conclude that instead of directly solving the SDP we can solve the DOOM problem with S the set of all possible circular shifts of the initial syndrome s and then retrieve the solution we are looking for. The DOOM algorithm is well explained in its original paper [Sen11] and achieves a speed-up factor of $\sqrt{n}$ (where n is the dimension of the circulant blocks) with respect to ISD algorithms working with the general version of the SDP.

In particular, the scheme we propose always deals with 2-QC codes. We adapt the associated problems below.

Definition 2.1.9 (2-QCCFP)**.** The 2-Quasi-Cyclic Codeword Finding problem (2-QCCFP) is the computational problem that takes as input a matrix $\mathbf{H} = (\mathbf{I}_n \mid \mathrm{circ}(h)) \in \mathbb{F}_2^{n\times 2n}$, for some $h \in \mathbb{F}_2^n$, two non-negative integers w_1, w_2, and asks to find $x = (x_1, x_2) \in \mathbb{F}_2^{2n}$ such that $\mathbf{H}x^\top = 0$, with $\mathrm{w}(x_1) = w_1$ and $\mathrm{w}(x_2) = w_2$.

Definition 2.1.10 (2-QCSDP)**.** The 2-Quasi-Cyclic Syndrome Decoding Problem (2-QCSD) is the computational problem that takes as input a matrix $\mathbf{H} = (\mathbf{I}_n \mid \mathrm{circ}(h)) \in \mathbb{F}_2^{n\times 2n}$, for some $h \in \mathbb{F}_2^n$, a vector $y \in \mathbb{F}_2^n$, two non-negative integers w_1, w_2, and asks to find $x = (x_1, x_2) \in \mathbb{F}_2^{2n}$ such that $\mathbf{H}x^\top = y^\top$, with $\mathrm{w}(x_1) \leq w_1$ and $\mathrm{w}(x_2) \leq w_2$.

2.1.2 The Scheme

In the following, we present our proposal for a digital signature scheme involving QC-LDPC codes.

Setup. The parameters of the scheme are:

- n, the dimension of the vector space $\mathbb{F}_2^n$. It is a prime number such that 2 is a primitive root modulo n;
- the weights w, w_{pq}, w_r, where $\omega, \omega_{pq}, \omega_r$ are integers smaller than n, w_{pq} is odd and w_r is even;
- two intervals I and I_t;
- H_{ω_r}, a hash function whose digests have weight ω_r over $\mathbb{F}_2^n$.

In the following, we describe the key generation algorithm KeyGen, the signature algorithm Sign and the verification algorithm Vf. We assume that the global parameters $(n, \omega, \omega_{pq}, \omega_r, I, I_t, H_{\omega_r})$ are known to anyone and we do not specify them as input to the algorithms.

Key Generation. The Key Generation algorithm KeyGen is the following.

Algorithm 4 Key generation algorithm KeyGen

Input $\emptyset$

Output $(\mathsf{pk}, \mathsf{sk})$

1: Generate randomly $e_1, e_2 \in R := \mathbb{F}_2[x]/(x^n - 1)$, with $\mathrm{w}(e_1) = \mathrm{w}(e_2) = \omega$
2: Generate randomly $p, q \in R$, with $\mathrm{w}(p) = \mathrm{w}(q) = \omega_{pq}$
3: Define $h := pq^{-1}$
4: Define $s := e_1 + he_2$
5: Output the public key $\mathsf{pk} = (h, s)$ and the private key $\mathsf{sk} = (e_2, q)$

Note that the actual secret key is $\mathsf{sk} = (e_2, q)$ but we want to keep (e_1, p) secret as well, although they are ephemeral and their knowledge is not required in the signature phase, because if an attacker can retrieve p or e_1, then it can retrieve at least part of the secret key sk.

Signature Algorithm. The signing algorithm Sign is as follows.

Algorithm 5 Signature algorithm Sign

Input $m, \mathsf{pk}, \mathsf{sk}$
Output A valid signature σ for m under pk and sk

1: Generate the following values:
- $r := H_{\omega_r}(m, \mathsf{pk}, \texttt{nonce})$, where nonce is a randomly chosen bitstring
- $t \in R$ such that $\mathrm{w}(t) \in I_t$
- $\alpha := qt + re_2$ and $\beta := \alpha h + sr$. If $\mathrm{w}(\alpha)$ or $\mathrm{w}(\beta)$ do not lie in I then change the nonce and repeat the signing phase

2: Output the signature $\sigma = (\alpha, \texttt{nonce})$

Verification Algorithm. The verification algorithm Vf is as follows.

Algorithm 6 Verification algorithm Vf

Input m, σ, pk
Output Accept/Reject

1: Compute $r := H_{\omega_r}(m, \mathsf{pk}, \texttt{nonce})$ and $\beta := \alpha h + sr$
2: Check that both $\mathrm{w}(\alpha)$ and $\mathrm{w}(\beta)$ lie in I
3: If these conditions are satisfied outputs Accept, otherwise Reject

Correctness

In the following, we deal with the correctness of the scheme. During the setup phase, we put some constraints on our parameters, namely, we required n to be a prime such that 2 is a primitive root modulo n, and w_{pq} to be odd. These choices are linked to the invertibility of q in the ring R. In particular, with this choice of parameters, q will always be invertible, as a direct consequence of the following facts.

Definition 2.1.11 (n-th cyclotomic field)**.** Let n be a positive integer. The splitting field of $x^n - 1$ over a field K is called the n-th cyclotomic field over K and is denoted by $K^{(n)}$. The roots of $x^n - 1$ in $K^{(n)}$ are called n-th roots of unity over K and the set of all these roots is denoted by $E^{(n)}$.

It is well known that if the characteristic of the field K does not divide the positive integer n, then $E^{(n)}$ is a cyclic group of order n. In this case, a generator of the cyclic group $E^{(n)}$ is called a primitive n-th root of unity over K.

Definition 2.1.12 (n-th cyclotomic polynomial)**.** Let K be a field of characteristic p, n a positive integer not divisible by p, and ζ a primitive n-th root of unity over K. Then, the polynomial

$$\phi_n(x) := \prod_{\substack{s=1 \\ (s,n)=1}}^{n} (x - \zeta^s)$$

is called the n-th cyclotomic polynomial over K.

It is well known that if K is a field of characteristic p and n is a positive integer not divisible by p, then

$$x^n - 1 = \prod_{d|n} \phi_d(x) \ ,$$

where $\phi_d(x)$ is the d-th cyclotomic polynomial. Furthermore, the following result [LN97] holds.

Proposition 2.1.13. *Let $\mathbb{F}_q$ be a finite field. If n is an integer such that $(n, q) = 1$, then $\phi_n(x)$ factors into $\varphi(n)/d$ distinct monic irreducible polynomials in $\mathbb{F}_q[x]$ of the same degree d, $\mathbb{F}_q^{(n)}$ is the splitting field of any such irreducible factor over $\mathbb{F}_q$, and $[\mathbb{F}_q^{(n)} : \mathbb{F}_q] = d$, where d is the least positive integer such that $q^d \equiv 1 \pmod{n}$.*

As we are working with $R = \mathbb{F}_2[x]/(x^n + 1) = \mathbb{F}_2[x]/(x^n - 1)$, since n is prime, $x^n - 1$ factors as $\phi_1(x)\phi_n(x)$ where both $\phi_1(x)$ and $\phi_n(x)$ are irreducible in $\mathbb{F}_2[x]$. To prove this, note that the latter factors into $\phi(n)/d$ irreducible polynomials where d is the order of 2 modulo n. Since we have chosen n such that 2 is a primitive root modulo n, we have that $d = n - 1 = \varphi(n)$. As a consequence, an element in R is invertible if and only if it is not divisible by $\phi_1(x)$ and by $\phi_n(x)$, but in R the only element divisible by $\phi_n(x)$ is itself, thus an element different from $\phi_n(x)$ is invertible in R if (and only if) it is not divisible by $\phi_1(x) = x + 1$.

Proposition 2.1.14. *Let $f \in R$. If f is divisible by $x + 1$, then $\mathrm{w}(f)$ is even.*

Proof. Let f be divisible by $x + 1$, so that we can write $f = (x + 1)f_1$. Notice that $f(1) = (1 + 1)f_1(1) = 0 \pmod 2$, so that f must have an even number of coefficients different from zero. Said otherwise, $\mathrm{w}(f)$ is even. □

According to these results, if we execute the protocol honestly, q will always be invertible in R.

Remark 2.1.15. Notice that it is possible to create hash functions which produce words of weight w_r, indeed H_{ω_r} could work in the following way. Let H be a cryptographically secure hash function with digest of 256 bit, and let n, w_r be the parameters defined in the setup phase. The goal of this function is to output a list of w_r integers in the range $[1, N]$, which will be the positions of the ones of the digest message. First, H_{ω_r} computes $\bar{m} = (m, \texttt{nonce})$ and defines $H_1 = H(\bar{m})$. In order to have a number less or equal than N we need $l = \lceil \log_2(N) \rceil$ bit, so the algorithm takes the first l bits of H_1 and, if the number associated with is in $[1, N]$, it will denote the position of the first one, otherwise it will discard the number. Following this pattern, the algorithm takes the second l bits, do the same as before, and so on. If it can no longer take l bits, the algorithm computes $H_2 = H(H_1)$ and continues until it has generated w_r different integers.

To prove the correctness of our scheme, we need the following preliminary result, whose proof can be found in [Mel+18a].

Proposition 2.1.16. *Let $v = (v_0, \ldots, v_{n-1})$ be a random vector chosen uniformly among all binary vectors of weight ω_v and let $u = (u_0, \ldots, u_{n-1})$ be a random vector chosen uniformly among all vectors of weight ω_u and independently of v. Then, denoting $z = u \cdot v$, we have that for every $k \in \{0, \ldots, n-1\}$, the k-th coordinate z_k of z is Bernoulli distributed with parameter $\tilde{p} = P(z_k = 1)$ equal to:*

$$\tilde{p} = \frac{1}{\binom{n}{w_v}\binom{n}{w_u}} \sum_{\substack{1 \le l \le min(w_v, w_u) \\ l \ odd}} C_l \ ,$$

where $C_l = \binom{n}{l}\binom{n-l}{w_v-l}\binom{n-w_v}{w_u-l}$.

As a consequence, in the following corollary we derive the probability distribution of the polynomials α and β.

Corollary 2.1.17. *Using the previous notation, α and β has the same weight distribution. In particular, α and β are distributed as a Binomial with parameter $p^* = p_1(1-p_2) + p_2(1-p_1)$, where*

$$p_1 = \frac{1}{\binom{n}{w_{pq}}\binom{n}{w_t}} \sum_{\substack{1 \le l \le min(w_{pq}, w_t) \\ l \text{ odd}}} \binom{n}{l}\binom{n-l}{w_{pq}-l}\binom{n-w_{pq}}{w_t-l},$$

$$p_2 = \frac{1}{\binom{n}{w_r}\binom{n}{w}} \sum_{\substack{1 \le l \le min(w_r, w) \\ l \text{ odd}}} \binom{n}{l}\binom{n-l}{w_r-l}\binom{n-w_r}{w-l}.$$

Proof. Consider $\alpha = q \cdot t + r \cdot e_2$ and let $p_1 = \mathbb{P}((q \cdot t)_i = 1)$. Recall that $\mathrm{w}(q) = \omega_{pq}$ and $\mathrm{w}(t) = \omega_t \in I_t$. Using proposition 1 we have that:

$$p_1 = \frac{1}{\binom{n}{\omega_{pq}}\binom{n}{\omega_t}} \sum_{\substack{1 \le l \le \min(\omega_{pq}, \omega_t), \\ l \text{ odd}}} \binom{n}{l}\binom{n-l}{\omega_{pq}-l}\binom{n-\omega_{pq}}{\omega_t-l}.$$

If we define $p_2 = \mathbb{P}((r \cdot e_2)_i = 1)$ with $\mathrm{w}(r) = \omega_r$ and $\mathrm{w}(e_2) = \omega$, using proposition 1 we have that:

$$p_2 = \frac{1}{\binom{n}{\omega_r}\binom{n}{\omega}} \sum_{\substack{1 \le l \le \min(\omega_r, \omega), \\ l \text{ odd}}} \binom{n}{l}\binom{n-l}{\omega_r-l}\binom{n-\omega_r}{\omega-l}.$$

Let $p^* = \mathbb{P}(\alpha_i = 1)$. We have that:

$$\begin{aligned} p^* &= \mathbb{P}((q \cdot t + r \cdot e_2)_i = 1) \\ &= \mathbb{P}((q \cdot t)_i = 1)\mathbb{P}((r \cdot e_2)_i = 0) + \mathbb{P}((q \cdot t)_i = 0)\mathbb{P}((r \cdot e_2)_i = 1) \\ &= p_1(1-p_2) + (1-p_1)p_2. \end{aligned}$$

We can conclude that the weight distribution of α is a Binomial of parameter p^*. If we consider β we have the following:

$$\begin{aligned} \beta &= h \cdot \alpha + s \cdot r \\ &= h \cdot q \cdot t + h \cdot r \cdot e_2 + r \cdot e_1 + r \cdot h \cdot e_2 \\ &= p \cdot q^{-1} \cdot q \cdot t + p \cdot q^{-1} \cdot r \cdot e_2 + r \cdot e_1 + r \cdot p \cdot q^{-1} \cdot e_2 \\ &= p \cdot t + r \cdot e_1. \end{aligned}$$

We have just to observe that β has the same form as α, $\mathrm{w}(p) = \mathrm{w}(q) = \omega_{pq}$ and $\mathrm{w}(e_1) = \mathrm{w}(e_2) = \omega$, then the thesis follows. □

According to Corollary 2.1.17, the public parameters n, w, w_{pq}, w_r and I_t determine the probability distribution of the weight of α and β, and thus we can find an interval I such that, if the scheme is executed honestly, the failure probability is negligible.

Security

In the following, we discuss some techniques that an hypothetical attacker might try to exploit to forge a new signature or to obtain the private key, proving that the success of these attacks is linked to the solutions of problems which are known to be difficult. According to this, we start by analyzing some key recovery and forgery attacks, ending with a discussion of the role of t.

Key Recovery from s. Regarding the relationship between private and public keys, notice that s satisfies the linear equation $s = e_1 + he_2$, which can be written as

$$s^\top = \left(\mathbf{I}_n \mid \mathrm{circ}(h) \right) \begin{pmatrix} {e_1}^\top \\ {e_2}^\top \end{pmatrix},$$

where e_1 and e_2 are two vectors of weight $\mathrm{w}(e_1) = \mathrm{w}(e_2) = \omega$.

Note also that if an attacker can retrieve (e_1, e_2) from s and also has access to a valid signature $(\alpha = qt + re_2, \texttt{nonce})$, then it can produce a forgery. In fact, from α it can compute qt and then select a message m', computing r' such that $\alpha' = qt + r'e_2$ is a valid signature for m'. However, the problem of finding e_1, e_2 from s is related to the 2-QCSDP. In our case, the parity-check matrix is given by $\mathbf{H} = (\mathbf{I}_n \mid \mathrm{circ}(h))$ and we have to take into consideration the sparsity of the matrix if we aim to fully understand the security of the scheme. In fact, if $\mathbf{H}$ is a sparse matrix, then the dual of the code generated by $\mathbf{H}$ is an LDPC code and in that case it is well known that the SDP, and therefore the 2-QCSDP, can be solved in polynomial time [Gal62]. However, under the assumption that h is indistinguishable from a randomly chosen vector (of odd weight) over $\mathbb{F}^n$, the 2-QCSDP does not seem to be efficiently solvable. Notice that the indistinguishability of h from a random vector is assumed to be true also in well-known works (e.g., BIKE [Ara+17] and LEDAcrypt [Bal+19]). The fact that with very high probability $\mathbf{H} = (\mathbf{I}_n \mid \mathrm{circ}(h))$ is not a sparse matrix allows us to conclude that if we could solve the problem of finding (e_1, e_2) from $s = e_1 + he_2$, then we could solve an instance of the computational 2-QCSDP, which is believed to be difficult to solve. Therefore, breaking the scheme in this way does not seem to be a viable option.

Key Recovery from h. Here we consider the possibility to exploit the knowledge of h, together with the information $h = pq^{-1}$, to retrieve p and q. A way to try to retrieve p and q is the following: construct the matrix $\mathbf{M} := \left(\mathbf{I}_n \mid \mathrm{circ}(h)^\top\right)$ and notice that

$$q \cdot \mathbf{M} = (q_0, q_1 \ldots, q_{n-1}) \cdot \begin{pmatrix} 1 & 0 & \cdots & 0 & h_0 & h_1 & h_2 & \cdots & h_{n-1} \\ 0 & 1 & \cdots & 0 & h_{n-1} & h_0 & h_1 & \cdots & h_{n-2} \\ \vdots & \vdots & \ddots & 0 & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & h_3 & \cdots & h_0 \end{pmatrix} = (q, p),$$

so that (q, p) is a codeword of low weight of the code with generator matrix $\mathbf{M}$. As a consequence, (q, p) is a solution of an instance of the 2-QCCFP, which we assumed to be difficult. Similarly to the previous attempt, breaking the scheme in this way does not seem to be a viable option.

Key Recovery from a Valid Signature. Here we consider the case in which an attacker has access to a valid signature $(\alpha, \texttt{nonce})$ of a given message m. The attacker can compute r and try to find the vector (qt, e_2) simply by solving

$$\alpha^\top = (\mathbf{I}_n \mid \mathrm{circ}(r))(qt, e_2)^\top.$$

Observe that the matrix $\mathbf{H} = (\mathbf{I}_n \mid \mathrm{circ}(r))$ is a sparse matrix and so the dual of the code generated by $\mathbf{H}$ is an LDPC code. This tells us that the syndrome decoding related to this code is efficiently solvable. In particular, we know an efficient algorithm that takes as input a parity-check matrix $\mathbf{H}$, a syndrome s, a weight w, and outputs a solution x such that $\mathbf{H}x^\top = s^\top$ and $\mathrm{w}(x) \leq w$. It is easy to observe that if someone succeeds in carrying this attack with non-negligible probability, then, in addition to being able to steal a part of the private key, it would be able to perform a forgery with non-negligible probability. In the following, we prove that this attack can succeed only with negligible probability. Notice that we know the weights of each polynomial involved in the expression (qt, e_2), so we can compute a weight $w_{\max}$ such that with very high probability the weight of (qt, e_2) is less than $w_{\max}$. In other words, we have a reasonable weight for the SDP with matrix $\mathbf{H} = (\mathbf{I}_n \mid \mathrm{circ}(r))$ and syndrome $\alpha^\top$. We show that, given $\mathbf{H}$, α and w, the number of vectors x that are solutions of the associated SDP are just too many to hope to find (qt, e_2) among these. Let $\alpha \in \mathbb{F}_2^n \setminus \{0\}$ be a syndrome and define $C_\alpha := \{v \in \mathbb{F}_2^{2n} \mid \mathbf{H}v^\top = \alpha^\top\}$. Assume that the weight distribution of the elements in C_α can be approximated with a binomial distribution with length $2n$ and probability $p = 1/2$. If we randomly extract a vector $v \in \mathbb{F}_2^{2n}$, then

$$\mathbb{P}(\mathrm{w}(v) = i) = \frac{\binom{2n}{i}}{2^{2n}}.$$

It is easy to show that $|C_\alpha| = 2^n$, therefore:

$$\begin{aligned}
\big|\{v \mid v \in C_\alpha \text{ and } \mathrm{w}(v) \leq w_{\max}\}\big| &= 2^n \cdot \sum_{i=0}^{w_{\max}} \mathbb{P}\big(\mathrm{w}(v) = i\big) \\
&= 2^n \cdot \sum_{i=0}^{w_{\max}} \frac{\binom{2n}{i}}{2^{2n}} \\
&= \frac{1}{2^n} \cdot \sum_{i=0}^{w_{\max}} \binom{2n}{i}.
\end{aligned}$$

In conclusion, with this kind of approach and with the parameters used to instantiate the scheme, the probability of finding the correct vector (qt, e_2) among these is negligible.

Forging a Signature of a Chosen Message. In the following, we consider the case where an attacker is interested in making a forgery with the only knowledge of the public key. To produce a forgery of a given message, an adversary is required to determine two values α and β such that their weights lie in the interval I. According to the previous subsections, a valid user is capable of producing with non-negligible probability a signature due to the knowledge of the private key. On the other hand, an adversary capable of producing a forgery is also capable of solving the problem

$$\begin{cases} \mathrm{w}(\alpha h + sr) \in I \\ \mathrm{w}(\alpha) \in I \end{cases}. \tag{2.2}$$

Notice that the adversary has no control over the value of r because this is the digest of a cryptographically secure hash function. If an attacker can find a correct triple (α, β, r) that satisfies the verification phase, it is still necessary to find a message m such that $H_{\omega_r}(m, \mathsf{pk}, \mathsf{nonce}) = r$. This means that the attacker is able to invert a hash function, but this is computationally unfeasible. On the other hand, a signature generated differently would not be linked to any known message with overwhelming probability. According to

this, an attacker can choose a message m, compute $r = H_{\omega_r}(m, \mathtt{pk}, \mathtt{nonce})$, and only then try to find α and β such that Eq. (2.2) is satisfied. If we denote by $\omega_{\max}$ the greatest integer that is still in the interval I, finding α and β is equivalent to solving the following problem.

$$\begin{cases} (sr)^\top = \left(\mathbf{I}_n \mid \mathrm{circ}(h)\right) \begin{pmatrix} \beta^\top \\ \alpha^\top \end{pmatrix} \\ \mathrm{w}(\alpha) \leq \omega_{\max} \\ \mathrm{w}(\beta) \leq \omega_{\max} \end{cases} .$$

As before, this is an instance of 2-QCSDP, which we assumed to be difficult, thus suggesting that breaking the scheme in this way is not feasible.

The usage of t. Notice that it is mandatory to use the ephemeral value t just once. As usual for such signatures, it would be a dab idea to send different messages using the same ephemeral key. In our case we would end up with two signatures $(\alpha_1, \mathtt{nonce}_1), (\alpha_2, \mathtt{nonce}_2)$ such that

$$\begin{cases} \alpha_1 = qt_1 + r_1 e_2 \\ \beta_1 = h\alpha_1 + sr_1 \end{cases} \quad \text{and} \quad \begin{cases} \alpha_2 = qt_1 + r_2 e_2 \\ \beta_2 = h\alpha_2 + sr_2 \end{cases} ,$$

from which it immediately follows that $\alpha_1 + \alpha_2 = (r_1 + r_2)e_2$. If $(r_1 + r_2)$ was invertible than we could compute $(\alpha_1 + \alpha_2)(r_1 + r_2)^{-1}$ and find e_2. However, this attack is not possible because $r_1 + r_2$ is always non-invertible in our framework. Indeed, a simple argument on the weights of r_1, r_2 and their sum shows that $r_1 + r_2$ is invertible in R if an only if only one of the two addends is invertible, as shown below.

Proposition 2.1.18. *Let $r_1, r_2 \in R$. The sum $r_1 + r_2$ is invertible if and only if r_1 is invertible (non-invertible) and r_2 is non-invertible (invertible).*

Proof. Suppose that $r_1 + r_2$ is invertible. Then, there exists $g \in R$ such that $(r_1 + r_2)g = 1$. In particular, we have $r_1 g = r_2 g + 1$. Suppose that r_2 is invertible. In this case $r_2 g$ is also invertible, so that $\mathrm{w}(r_2 g)$ is odd, $\mathrm{w}(r_2 g + 1) = \mathrm{w}(r_1 g)$ is even, and $r_1 g$ is non-invertible. The only reason this case can occur is when r_1 is non-invertible. The same reasoning can be done if we take r_2 as non-invertible. Here, $r_2 g + 1$ is invertible, but $r_2 g + 1 = r_1 g$, and so we have that r_1 is invertible. Conversely, if the sum $r_1 + r_2$ defines a polynomial which is non-invertible, then we have $(r_1 + r_2)\phi_n = 0$, where $\phi_n = 1 + x + x^2 + \cdots + x^{n-1}$ is the n-cyclotomic polynomial. This means that $r_1\phi_n = r_2\phi_n$. Now, observe what happens if we multiply some $r = (r_0, r_1, \ldots, r_{n-1})$ for ϕ_n:

$$r \cdot \phi_n = (r_0, \ldots, r_{n-1}) \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix} = \left(\sum_{i=0}^{n-1} r_i, \ldots, \sum_{i=0}^{n-1} r_i \right),$$

where the sum is intended modulo 2. Now, we know that if r_1 is invertible, then its weight is odd, and in this case $r_1\phi_n = (1, \ldots, 1)$. Similarly, if r_1 is non-invertible, then its weight is even, and $r_1\phi_n = (0, \ldots, 0)$. In conclusion, $r_1\phi_n = r_2\phi_n$ holds only if $\mathrm{w}(r_1)$ and $\mathrm{w}(r_2)$ has the same parity, i.e. r_1 and r_2 are both invertible or non-invertible. □

According to this, $r_1 + r_2$ is non-invertible if and only if r_1 and r_2 are both invertible or non-invertible. Despite this, an attacker can still try to find e_2 by solving the linear system of equations generated by $\mathrm{circ}(r_1+r_2)e_2^\top = (\alpha_1+\alpha_2)^\top$. Hence, we have to consider the number of possible solutions of that system. According to this, we limit ourselves to consider the linear system described by $\mathrm{circ}(r)e_2 + \alpha$, where α and r are known, r has even weight, and we ask for how many e_2 this system admits a solution. Notice that the elements that are solutions of the previous system are exactly the elements that belong to the fiber of $\mathcal{L}_{r,\alpha}$ at zero, where

$$\begin{aligned} \mathcal{L}_{r,\alpha}\colon\ \mathbb{F}_2^n &\longrightarrow \mathbb{F}_2^n \\ x &\longmapsto \mathrm{circ}(r)x + \alpha \end{aligned}.$$

Since we are interested in computing $(\mathcal{L}_{r,\alpha})^{-1}(0)$ and the function $x \mapsto x+\alpha$ is a translation map, we can study the fiber of $\mathcal{L}_{r,0}$ at zero. It is easy to show that $\mathcal{L}_{r,0}$ is an homomorphism of vector spaces, so that the First Isomorphism Theorem guarantees that there is a bijection between the image space $\mathcal{L}_{r,0}(\mathbb{F}_2^n)$ and the quotient space $\mathbb{F}_2^n/\ker(\mathcal{L}_{r,0})$, from which it follows that

$$|\ker(\mathcal{L}_{r,0})| = \frac{|\mathbb{F}_2^n|}{|\mathcal{L}_{r,0}(\mathbb{F}_2^n)|}.$$

This computation is achievable; indeed, the rank of a circulant matrix $\mathrm{circ}(r)$ is equal to $n - d$, where d is the degree of the polynomial $(r, x^n - 1)$, as stated below.

Proposition 2.1.19 ([Kna06])**.** *Let $r \in R$ and consider the matrix $\mathbf{M} := circ(r) \in \mathbb{F}_2^{n\times n}$. The rank of $\mathbf{M}$ is of the form $n - d$, where d is the degree of the polynomial $(r, x^n - 1)$.*

In our case $x^n - 1 = (x+1)(1+x+\cdots+x^{n-1})$, so the greatest common divisor between $x^n - 1$ and r is $x + 1$. It follows immediately that $d = 1$ and the rank of $\mathrm{circ}(h)$ is $n - 1$. As a consequence, $|\ker(\mathcal{L}_{r,0})| = 2^n/2^{n-1} = 2$, that is, it contains only e_2 and the zero element. Going back to the original problem, we have that $\mathrm{circ}(r_1 + r_2)e_2 = \alpha_1 + \alpha_2$ has two solutions. Moreover, if e_2' is a solution, then $e_2' + (1, \ldots, 1)$ is the second solution: Indeed, we have seen that $r_1 + r_2$ can not be invertible, so that

$$(r_1 + r_2)(e_2' + (1, \ldots, 1)) = (r_1 + r_2)e_2' + (r_1 + r_2)(1, \ldots, 1) = (\alpha_1 + \alpha_2) + 0 = (\alpha_1 + \alpha_2).$$

In conclusion, since there are only two possible solutions of that system, we must take care of the usage of t, changing its value every time we sign a document.

Parameters

In the following, we describe the choice of the parameters for the security levels $1, 3$ and 5, as required by the NIST. The choice is justified by some related experimental results. According to the description of the setup phase, described in Subsection 2.1.2, we choose w_{pq} and w to have approximately half the number of bits of n. The weight of r is taken in such a way that $w_r \approx 3\log n$, and I_t is the interval of length $\sqrt{n}$ and centered in $\sqrt{n}$. This choice allows us to compute I in such a way that it will be possible for a genuine signer to sign efficiently and it will be unfeasible for an attacker to break the protocol. The choice of the size of these parameters is summarized in Table 2.1.

Parameter	Size
ω_{pq}	$\approx \sqrt{n}$
I_t	$[\lceil \frac{\sqrt{n}}{2} \rceil, \ldots, \lceil \frac{2\sqrt{n}}{3} \rceil]$
ω_r	$\approx 3\log(n)$
ω	$\approx \sqrt{n}$

Table 2.1: Size of the parameters.

In Table 2.2 we define the different values for n for the standard security levels and then describe the best attacks to our scheme.

Security level	n
128	14627
192	18143
256	21067

Table 2.2: Values for the security levels.

As shown in the previous subsection, the security of $h = pq^{-1}$ is strictly related to the hardness of CFP. This problem in NP-Complete in the general case and in the case of Quasi-Cyclic codes the attack is just slightly improved (DOOM attack [Sen11]), as shown in the introductory subsection. In order to estimate the complexity of an attack, we used a Python script which can be found in [BE21]. We used the same test to analyze the security level of $s = e_1 + he_2$, which rely on the difficulty of the 2-QCSDP. The results are summarized in Table 2.3.

n	Security bits (attack to h)	Security bits (attack to s)
14627	250	250
18143	277	277
21067	297	297

Table 2.3: Security bits for attacks to h and s.

The size of n may seem overestimated with respect to the security parameters, but this is not the only point on which the security lies. We also have to take into account the distance that the weight distribution of α has with respect to a binomial distribution with parameters $(n, \frac{1}{2})$. We know that the weight distribution of α is a binomial random variable with parameter (n, p^*), where p^* is computed as described in Corollary 2.1.17. Since the weight of t is not fixed, but can vary within the range I_t, p^* can also vary accordingly. In Table 2.4 we summarize the range of p^* and the interval I for the different security levels. The interval I is computed in such a way that, if the signer is honest, the probability that the weight of α and β results outside the range I is approximately 10^{-10}.

Security level	Range of p^*	I
128	$[0.38918, 0.42088]$	$[5339, 6515]$
192	$[0.38549, 0.41790]$	$[6601, 7981]$
256	$[0.38174, 0.41564]$	$[7620, 9186]$

Table 2.4: Ranges for α and β.

So, an attacker may try to guess an α in the range I and check if $\beta = \alpha h + rs$ also belongs to the range I. Notice that if an attacker can find such α, then it can create a forgery. In this case, since we do not have any constraint on the weight of h, β appears as a totally random vector, i.e., its weight distribution is a binomial random variable with parameters $(n, \frac{1}{2})$. The security bits for this type of attack for security level $128, 192$ and 256 are respectively given by $133, 196$ and 256.

Notice that an attacker could try to solve the SDP related to α, i.e. it can try to find (qt, e_2) knowing that $(\mathbf{I}_n \mid \mathrm{circ}(r))(qt, e_2)^\top = \alpha^\top$. Observe that this instance of the SDP is easily solvable, since the associated code is an LDPC code [Gal62]. In Table 2.5 we can observe that the total number of solutions is too large to hope that the vector v given from the decoder is exactly the sensible information (qt, e_2).

Security level	p_{qt}	$\omega_{\max}$	Number of solutions
128	0.37153	5652	$\approx 10^{1905}$
192	0.37118	6977	$\approx 10^{2333}$
256	0.37169	8092	$\approx 10^{2670}$

Table 2.5: Attack to α. $\mathrm{w}(qt)$ is distributed as a binomial random variable with parameters (n, p_{qt}), and $\omega_{\max}$ is such that the probability that the weight of (qt, e_2) is greater than $\omega_{\max}$ is negligible.

At the end, we can study the hardness of forging a signature by solving the SDP instance defined in Subsection 2.1.2. In order to do this we use the best algorithm to attack the SDP and the complexity of these attacks can be computed using [BE21]. Since, in this case, the parameters are very big, we could not compute the complexity of all the attacks. Table 2.6 provides an idea about the complexity of computing a forgery in this way.

Security level	Security bits for forgery exploiting SDP
128	≈ 7000
192	≈ 8000
256	≈ 9000

Table 2.6: Forgery complexity by attacking SDP.

With the above discussion, we have provided hints on the security of the scheme. Nevertheless, this topic would have required a deeper analysis and possibly a formal security proof that would have reduced the security of the scheme to the complexity of solving the underlying difficult problems. Without this reduction, we left ample space for more sophisticated attacks such as the one presented in the next subsection.

2.1.3 A Recent Attack

One of the most delicate aspects of the scheme is the following. Notice that, according to Corollary 2.1.17, the distribution of the Hamming weight of α and β can be modeled as a binomial with parameter $p^* \neq \frac{n}{2}$, and thus is distinguishable from the distribution of random vectors. This feature, although fundamental to the success of the signing process, may be linked to a possible information leakage. This peculiarity of our scheme was exploited in 2023 to mount a partial key-recovery, from which a forgery attack immediately follows. We describe the salient moments of this attack in the following, referring the reader to [PT23] for a comprehensive discussion.

The main point behind the attack is this: if we consider $\alpha = qt + re_2$ and consider two indexes $i, j \in \{0, \dots, n-1\}$ such that $r_{i-j \pmod n} = 1$, then $\mathbb{P}(\alpha_i = 1)$ depends on whether $(e_2)_j = 1$ or $(e_2)_j = 0$.

To show this fact, consider $\alpha = qt + re_2$ and notice that $\alpha_i = (qt)_i + r^{(i)}{e_2}^\top$, where $r^{(i)}$ denotes the i-th row of $\mathrm{circ}(r)$. According to this, we have that

$$\alpha_i = (e_2)_j + (qt)_i + \sum_{\substack{0 \le \ell \le n-1 \\ \ell \neq j}} r_{i-\ell \pmod n}(e_2)_\ell.$$

We want to compute the distribution of the random variable α_i. A first step in this direction is to compute the distribution of the sum on the right-hand side of the equation above. To simplify the notation, we introduce the following definition.

Definition 2.1.20. Let $u, v \in \mathbb{F}_2^n$ be two binary vectors of length n, in such a way that $\mathrm{w}(u) = w_u$ and $\mathrm{w}(v) = w_v$. We denote by $P(n, w_u, w_v)$ the probability that $u \cdot v^\top = 1$.

Notice that this probability can be easily computed from Prop. 2.1.16. Given this, we have the following.

$$\mathbb{P}\left(\sum_{\substack{0 \le \ell \le n-1 \\ \ell \neq j}} r_{i-\ell \pmod n}(e_2)_\ell = 1 \right) = \begin{cases} P(n-1, w, w_r - 1) & \text{if } (e_2)_j = 0 \\ P(n-1, w-1, w_r - 1) & \text{if } (e_2)_j = 1 \end{cases}.$$

Denote $p_0 := P(n-1, w, w_r - 1)$, $p_1 := P(n-1, w-1, w_r - 1)$, and $p_{qt} := P(n, w_{pq}, w_t)$. Now, suppose $(e_2)_j = 0$. In this case $\alpha_i = (qt)_i + \sum_{\ell \neq j} r_{i-\ell \pmod n}(e_2)_\ell$ and

$$\begin{aligned} \mathbb{P}(\alpha_i = 1 \mid (e_2)_j = 0 \wedge r_{i-j \pmod n} = 1) &= \mathbb{P}((qt)_i + \sum_{\ell \neq j} r_{i-\ell \pmod n}(e_2)_\ell = 1) \\ &= p_{qt}(1 - p_0) + (1 - p_{qt})p_0. \end{aligned}$$

On the other hand, if $(e_2)_j = 1$, $\alpha_i = 1 + (qt)_i + \sum_{\ell \neq j} r_{i-\ell \pmod n}(e_2)_\ell$ and

$$\begin{aligned} \mathbb{P}\left(\alpha_i = 1 \mid (e_2)_j = 1 \wedge r_{i-j \pmod n} = 1\right) &= \mathbb{P}\Big(1 + (qt)_i + \sum_{\ell \neq j} r_{i-\ell \pmod n}(e_2)_\ell = 1\Big) \\ &= p_{qt}p_1 + (1 - p_{qt})(1 - p_1). \end{aligned}$$

Numerical values for the three security levels are reported in Table 2.7.

Security level	$\mathbb{P}(\alpha_i = 1 \mid (e_2)_j = 1 \wedge r_{i-l \pmod n} = 1)$	$\mathbb{P}(\alpha_i = 1 \mid (e_2)_j = 0 \wedge r_{i-l \pmod n} = 1)$
128	0.577417	0.423024
192	0.582959	0.417421
256	0.585075	0.415277

Table 2.7: Numerical values for $\lambda \in \{128, 192, 256\}$.

.

For every security level, the probabilities reported in Table 2.7 always have 0.5 as intermediate value between them. According to this, the authors of [PT23] decided to set a threshold value $\delta = 0.5$ in order to attempt to distinguish the distributions. We report in Algorithm 7 the steps which describes the attack and recovers one coordinate $(e_2)_j$ of e_2. It is clearly possible to repeat the algorithm n times in order to fully recover e_2.

Algorithm 7 Recovering $(e_2)_j$ for $j \in \{0, \dots, n-1\}$

Input A list $(m^u, \alpha^u, \texttt{nonce}^u)_{u=1}^N$ of signed messages, $j \in \{0, \dots, n-1\}, \delta \in [0, 1]$.
Output $(e_2)_j$.

Set $S = 0$;
for $1 \leq u \leq N$ **do**
  Compute $r^u = H_{\omega_r}(m_u, \texttt{pk}, \texttt{nonce}_\texttt{u})$
  for $0 \leq i \leq n-1$ **do**
    if $r^u_{i-j \mod n} = 1$ **then**
      $S = S + \alpha_i$
    end if
  end for
end for
$S = S/(N \cdot w_r)$
if $S < \delta$ **then**
  $(e_2)_j = 0$
else $(e_2)_j = 1$
  $(e_2)_j = 1$
end if

Once the reconstruction of e_2 has been successfully completed, it is possible to mount a forgery attack, as already outlined in the previous subsections, as well as in [PT23].

As a conclusive remark, we emphasize the major issues behind this proposal. We have only partially modeled the adversary, which has led to an attack that we had not anticipated. This is one of the reasons why modern cryptography is shifting towards "provably secure" schemes. Informally, this means that, given a scheme and some reasonable assumptions, it is possible to reduce the security of the scheme to the security of another problem, typically hard to solve. In the next chapter, we focus not on the construction of new signature schemes, but rather on the cryptanalysis of the existing ones.

Chapter 3

Cryptanalysis

In the previous chapter, we attempted to design a digital signature scheme. In this chapter, we look at the other side of the coin, studying the security of other signature schemes.

In the first section, we analyze the security of HWQCS, a signature scheme which aims to address the issues of the scheme proposed in the previous chapter. In particular, we will show that the preventive measures taken into account to counter well-known attacks are still insufficient to make the scheme secure. Specifically, we will demonstrate how an UF-CMA attack can be mounted, requiring only about a dozen valid signatures.

In the second section, we study the formal security of Wave, which is essentially based on two assumptions: on the one hand, the difficulty of solving the SDP; on the other, the indistinguishability of the structured codes used by the algorithm. We will attempt to construct a distinguisher capable of determining whether a given input code originates from a random distribution or from the family of codes used in Wave. Our attempt heavily exploits the weight distribution of these codes, as Wave's low-weight codewords have a distribution which is different from that of a random code. Unfortunately, our attempt will not succeed. We will then analyze the reasons for this failure.

The third section stems from the second: while attempting to construct a distinguisher for the codes used in Wave, we found it necessary to introduce estimators that, regardless of the outcome of the cryptanalysis, have standalone significance. We will present algorithms for estimating the weight distribution of a random code, comparing them with what is already available in the literature.

3.1 A Successful Cryptanalysis: HWQCS

In 2023, a new attempt to build a code-based signature scheme, called HWQCS [TP24], has been made. HWQCS uses QC-LDPC codes with the Hamming metric and introduces the use of high-weight errors to make the decoding problem harder for an attacker. The construction of this scheme is designed so as to prevent other known attacks from being effective. We wondered the following question.

Is the HWQCS digital signature scheme secure?

In this section, we answer this question in the negative, showing that the signatures of HWQCS leak substantial information concerning the ephemeral keys and formally describe this behavior. Furthermore, we show that for each security level, we can exploit the leakage to efficiently reconstruct partial secret data from very few signatures, and finally mount a universal forgery attack.

The notation we use in this section mimics the one already introduced in the previous chapter. In particular, we let $R := \mathbb{F}_2[x]/(x^k - 1)$ be the ring of all polynomials over $\mathbb{F}_2$ of degree less than k. For $a \in \mathbb{F}_2^k$, we denote by $a(x)$ the unique polynomial $a(x) = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1} \in R$. Given a vector $a \in \mathbb{F}_2^k$, we denote its j-th entry by a_j when $a = (a_0, \ldots, a_{k-1})$ is specified, and by $(a)_j$ otherwise. Similarly, we denote the j-th coefficient of a polynomial $a(x) \in R$ as $(a(x))_j$. As before, with some abuse of notation, $a(x) \in R$ and its counterpart $a \in \mathbb{F}_2^k$ will be denoted with the same symbol a, simplifying notation and exposure at the price of a little imprecision throughout our discussion. Finally, we denote by $V_{k,w} \subset \mathbb{F}_2^k$ the set of vectors of length k and Hamming weight w.

3.1.1 The Scheme

This subsection describes the algorithms of HWQCS [TP24] for key generation KeyGen, signature Sign and verification Vf. We assume that the global parameters k, w_f, w_u, w_e, w_c, w_s, w_t, R and H_{ω_c} provided in the setup are public, and we do not specify them as input of the algorithms.

Setup. The parameters of the scheme are the positive integers $k, w_f, w_u, w_e, w_c, w_s, w_t$, the quotient ring $R = \mathbb{F}_2[x]/(x^k - 1)$ and H_{ω_c} a hash function with fixed output Hamming weight w_c

$k, w_f, w_u, w_e, w_c, w_s, w_t$	Positive integers.
R	The quotient ring $\mathbb{F}_2[x]/(x^k - 1)$.
H_{ω_c}	Hash function with fixed output Hamming weight w_c.

Key Generation. The Key Generation algorithm KeyGen works as follows. First, two elements f_1 and f_2 are randomly chosen from the set V_{k,w_f}. These vectors must satisfy the condition that their associated polynomials f_1 and f_2 are invertible in R. Then, the public key pk is the product of the inverse of f_1 by f_2. Finally, the private key sk is the pair (f_1, f_2). Algorithm 8 describes the key generation process for HWQCS.

Algorithm 8 Key generation algorithm KeyGen

Input : $\emptyset$
Output : a public key $\mathsf{pk} \in R$ and a private key $\mathsf{sk} \in R^2$

1: Choose random $f_1, f_2 \in V_{k,w_f}$ such that $f_1, f_2 \in R$ are invertible
2: Compute $h := f_1^{-1} \cdot f_2 \in R$
3: Output the public key $\mathsf{pk} = h$ and the private key $\mathsf{sk} = (f_1, f_2)$

Signature Algorithm. The signing algorithm Sign takes as inputs a message m and the public and secret keys $(\mathsf{pk}, \mathsf{sk})$, and works as follows. It starts by randomly selecting vectors e_1 and e_2 from the set V_{k,w_e}, as well as u_1 and u_2 from V_{k,w_u}. Next, two auxiliary polynomials are computed. The first is b, which depends on h, e_1 and e_2. The second is d, derived from the private key components f_1 and f_2 along with u_1 and u_2. A hash function H_{ω_c} is then applied to the message m and the values b, d, and h, producing a vector c in V_{k,w_c}. After, the values s_1 and s_2 are computed as $s_i := u_i f_i + c e_i$, for $i = 1, 2$. Notice that,

according to the parameters that will be chosen later, each s_i is obtained by multiplying and summing sparse polynomials and, moreover, one of such polynomials is public. This will be the starting point for our attack. The algorithm then verifies that the weights of s_1, s_2, and d meet the required constraints not exceeding some threshold parameters w_s and w_t. If any of these weights exceed such limitations, the algorithm restarts. The signature σ consists of the elements (c, b, s_1, s_2). Algorithm 9 describes the process of generating a signature σ for a message m using a public key pk and a private key sk.

Algorithm 9 Signing algorithm Sign

Input : $m, \mathsf{pk} = h, \mathsf{sk} = (f_1, f_2)$.
Output : $\sigma = (c, b, s_1, s_2)$

1: Choose random $e_1, e_2 \in V_{k,w_e}$ and $u_1, u_2 \in V_{k,w_u}$
2: Set $b := e_1 h + e_2 h^{-1} \in R$ and $d = u_1 f_2 + u_2 f_1$
3: Compute $c := H_{\omega_c}(m, b, d, h) \in V_{k,w_c}$
4: Compute $s_i := u_i f_i + c e_i$ for $i = 1, 2$
5: If $w(s_1) > w_s$ or $w(s_2) > w_s$ or $w(d) > w_t$ then repeat from Step 1
6: Output $\sigma = (c, b, s_1, s_2)$

Verification Algorithm. The verification algorithm begins by computing the polynomial t using the signature (c, b, s_1, s_2) and the public key pk. Specifically, t is computed as $s_1 h + s_2 h^{-1} - cb$. Next, a hash digest $c' \in V_{k,w_c}$ is computed applying H_{ω_c} to the message m and the values b, t, and h. The algorithm then checks if $c = c'$, the weight $\mathrm{w}(t)$ is less than or equal to w_t, and $t \neq 0$. If all conditions are satisfied, the signature is accepted as valid; otherwise, it is rejected. Algorithm 10 describes the process of verifying a signature σ for a message m using a public key pk.

Algorithm 10 Verification algorithm Vf

Input : $m, \mathsf{pk} = h, \sigma = (c, b, s_1, s_2)$
Output : `Accept` / Reject

1: Compute $t := s_1 h + s_2 h^{-1} - cb$
2: Compute $c' := H_{\omega_c}(m, b, t, h) \in V_{k,w_c}$
3: If $c = c'$, $\mathrm{w}(t) \leq w_t$ and $t \neq 0$, then `Accept`, otherwise Reject

The sets of parameters suggested for 128,192 and 256 bits of security are reported in Table 3.1.

Security	k	ω_f	ω_u	ω_e	ω_c	ω_s	ω_t
128	12539	145	33	141	31	4844	4937
192	18917	185	41	177	39	7450	7592
256	25417	201	51	191	51	10111	10216

Table 3.1: Suggested parameters for security levels 128,192 and 256 of HWQCS.

3.1.2 Information Leakage

We start by giving an overview of our attack, which serves as a motivation for the study of the objects developed in the rest of this analysis, where we show that the signatures

generated by the Sign algorithm of HWQCS leak a critical amount of information about the ephemeral values e_1 and e_2.

Overview of the attack

The idea is to use information set decoding techniques in a simplified scenario, which allows us to perform fast recovery of secret data, hence forge signatures. In other words, our objective is to find error-free positions in the vector e. To this end, we apply a similar analysis to [SBC19], which manipulates each intercepted signature in such a way that it is possible to separate a good number of zero bits of e_1 and e_2 from the one bits. Furthermore, we can carefully guess additional zero bits. Once k zero bits have been recovered, we can perform linear algebra using a submatrix of the public matrix

$$\mathbf{H} = \left[\mathsf{circ}(h) \mid \mathsf{circ}(h)^{-1}\right]. \tag{3.1}$$

to reconstruct the values e_1, e_2 completely. Finally, compute $u_i f_i = s_i - ce_i$ for $i = 1, 2$. At this point it is possible to start forging valid signatures, see Subsection 3.1.3 for a detailed explanation and Algorithm 11 for the concrete attack.

In the following, we will formally describe the behavior of the information leakage, which will be useful in determining the necessary objects for our attack. Our analysis is identical for both $i = 1, 2$, therefore we will not fix a value of i.

Statistical analysis of the information leakage

We describe the technique introduced in [SBC19] adapted to the setting of HWQCS. Let $\sigma = (c, b, s_1, s_2)$ be an intercepted signature where $s_i := u_i f_i + ce_i$, as per Sign. We can expose many one bits of e as follows:

- Let $v \in \mathrm{Supp}(c)$ and write

$$\begin{aligned} x^{-v} s_i &= x^{-v}\left(u_i f_i + ce_i\right) \\ &= x^{-v} u_i f_i + \left(1 + x^{-v} \sum_{l \in \mathrm{Supp}(c)\setminus\{v\}} x^l\right) e_i \\ &= e_i + x^{-v} u_i f_i + \sum_{l \in \mathrm{Supp}(c)\setminus\{v\}} x^{l-v} e_i. \end{aligned} \tag{3.2}$$

- Finally, compute

$$d_i := \sum_{v \in \mathrm{Supp}(c)} x^{-v} s_i \in \mathbb{Z}[x], \tag{3.3}$$

where the sum in 3.3 is taken over $\mathbb{Z}$. Notice that $x^{-v} u_i f_i$ is a polynomial whose coefficient vector is a circular shift to the left of the coefficient vector of $u_i f_i$ of v positions. The same holds for $x^{l-v} e_i$, so that for every $v \in \mathrm{Supp}(c)$, the value $x^{-v} s_i$ is given by e_i plus some random noise, see (3.2). Therefore, we expect that the larger coefficients of d_i are associated with the entries of e_i equal to 1. Figure 3.1 gives a visual representation of the information leakage for a random signature, of an instance of HWQCS for security level 128. The same behavior happens for security levels 192 and 256.

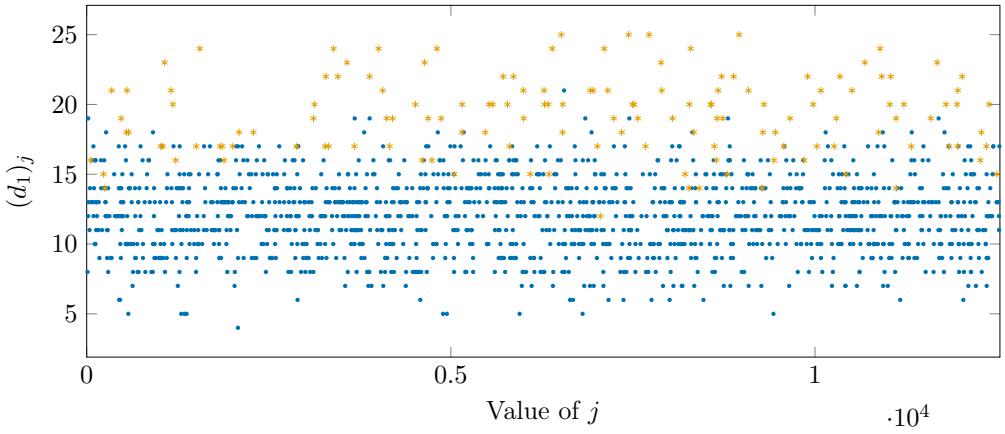

Figure 3.1: Information leakage of e_1 of an instance of HWQCS with security parameter 128. We highlighted with blue dots the entries j for which $(e_1)_j = 0$, and with orange asterisks the entries j for which $(e_1)_j = 1$.

From Figures 3.1, it can easily be seen that the one entries of e_i (in the examples in the figures $i = 1$) are pushed to the upper half of the plot, meaning that they correspond to the higher coefficients of d_i. This means that, with great probability, a random entry $(d_i)_j \in \mathbb{Z}$ which is small enough will correspond to a zero entry $(e_i)_j$. The rest of this subsection is dedicated to the proof of Theorem 3.1.1, which describes the behavior of the entries of $d_i \in \mathbb{Z}^k$.

Theorem 3.1.1. *Let u, f, e and c be random elements of R, such that $\mathrm{w}(u) = w_u, \mathrm{w}(f) = w_f, \mathrm{w}(e) = w_e$ and $\mathrm{w}(c) = w_c$, respectively. Let $v \in \mathrm{Supp}(c)$ and define*

$$d := \sum_{v \in \mathrm{Supp}(c)} \left(e_i + x^{-v} u_i f_i + \sum_{l \in \mathrm{Supp}(c) \setminus \{v\}} x^{l-v} e_i \right) \in \mathbb{Z}[x].$$

Assume that the random variables $x^{-v} u_i f_i$ and $\sum_{l \in \mathrm{Supp}(c)\setminus\{v\}} x^{l-v} e$ are independently distributed, as v and l vary. Then, for $j \in [k-1]$, we have that $(d)_j$ is binomially distributed as

$$(d)_j \sim Bin\left(w_c, \prod_{i=1}^{3} p_i + \sum_{i=1}^{3} p_i \prod_{j \in [3], j \neq i} (1 - p_j) \right), \tag{3.4}$$

where

$$p_1 = w_e/k, \qquad p_2 = (1 - (1 - 2w_e/k)^{w_c - 1})/2$$

and

$$p_3 = \frac{1}{\binom{k}{w_u}\binom{k}{w_f}} \sum_{\substack{1 \leq l \leq \min(w_u, w_f) \\ l \text{ odd}}} \binom{k}{l} \binom{k-l}{w_u - l} \binom{k - w_u}{w_f - l}.$$

Let $j \in [k-1]$. We will analyze the distribution of $(x^{-v} s_i)_j$ for a fixed $v \in \mathrm{Supp}(c)$, breaking down the analysis on the three summands of the last equation in (3.2), treated as random variables. That is, we are going to compute the probability distributions of $(x^{-v} u_i f_i)_j$ and of $\left(\sum_{l \in \mathrm{Supp}(c)\setminus\{v\}} x^{l-v} e_i\right)_j$, while we consider $(e_i)_j$ to be Bernoulli distributed with parameter ω_e/n. Afterward, we will provide the explicit probability distribution of the sum of the three random variables and describe the distribution of

$(d_i)_j = \left(\sum_{v \in \mathrm{Supp}(c)} x^{-v} s_i\right)_j$, where the sum is taken over $\mathbb{Z}$.

The following result is useful for computing the probability distribution of the entry $(x^{-v} u_i f_i)_j$.

Lemma 3.1.2. *[Mel+18a, Proposition 2.4.1] Let $u, f \in R$ be random elements such that $\mathrm{w}(u) = \omega_u$ and $\mathrm{w}(f) = \omega_f$. Set $z = uf$, then for every $j \in \{0, \ldots, k-1\}$, we have that $(z)_j$ is distributed as a Bernoulli random variable with parameter $p = P(z_j = 1)$ equal to:*

$$p = \frac{1}{\binom{k}{w_u}\binom{k}{w_f}} \sum_{\substack{1 \le l \le min(w_u, w_f) \\ l \text{ odd}}} \binom{k}{l}\binom{k-l}{w_u - l}\binom{k-w_u}{w_f - l}. \tag{3.5}$$

Lemma 3.1.2 provides us with a way to compute the distribution of $(u_i f_i)_j$.

Remark 3.1.3. Note that $x^{-v} u_i f_i$ is a polynomial whose coefficient vector is a circular left shift of the coefficient vector of $u_i f_i$ of v positions. Therefore, the two random variables $(u_i f_i)_j$ and $(x^{-v} u_i f_i)_j$ are identically distributed.

We move on to the distribution of $\sum_{l \in \mathrm{Supp}(c) \setminus \{v\}} x^{l-v} e_i$. We will treat $(x^{l-v} e_i)_j$ as independent random variables, as l and v vary.

Lemma 3.1.4. *Let $X_1, X_2, \ldots, X_k$ be k independent random variables following a Bernoulli distribution with parameter $q < \frac{1}{2}$ and let $X = \sum_{h=1}^{k} X_h$ be their sum over $\mathbb{F}_2$. Then X is Bernoulli distributed with parameter $p = \mathbb{P}(X = 1)$ equals to:*

$$p = \left(\frac{1}{2} - \frac{(1-2q)^k}{2}\right). \tag{3.6}$$

Proof. The random variable X is Bernoulli distributed with parameter recursively given by $T(k) = T(k-1)(1-q) + q(1 - T(k-1))$, with $T(0) = 0$. Our goal is to convert this expression into a closed formula. Consider the formal power series $f := \sum_{h=1}^{\infty} T(h) x^h$ where we consider the real interval $x \in (0, 1)$. We have that

$$\begin{aligned}
f &= \sum_{h=1}^{\infty} T(h) x^h \\
&= qx + \sum_{h=2}^{\infty} T(h) x^h \\
&= qx + \sum_{h=2}^{\infty} (T(h-1)(1-q) + q(1 - T(h-1))) x^h \\
&= qx + (1-q) \sum_{h=2}^{\infty} T(h-1) x^h + \sum_{h=2}^{\infty} q x^h - q \sum_{h=2}^{\infty} T(h-1) x^h \\
&= \sum_{h=1}^{\infty} q x^h + (1-q) xf - qxf \\
&= \sum_{h=1}^{\infty} q x^h + (1-2q) xf.
\end{aligned}$$

We can rewrite f as

$$\begin{aligned} f &= \frac{\sum_{h=1}^{\infty} qx^h}{1-x+2qx} \\ &= \frac{q \cdot (\frac{x}{1-x})}{1-x+2qx} \\ &= \frac{qx}{(1-x)\cdot(1-(1-2q)x)} \\ &= \frac{\frac{1}{2}}{(1-x)} - \frac{\frac{1}{2}}{(1-(1-2q)x)} \end{aligned}$$

For every constant A such that $|Ax| < 1$ we have that $\sum_{h=0}^{\infty}(Ax)^h = 1/(1-Ax)$, therefore we can rewrite both $1/(1-x)$ and $1/(1-(1-2q)x)$ as formal power series, obtaining

$$f = \frac{1}{2}\sum_{h=0}^{\infty} x^h - \frac{1}{2}\sum_{h=0}^{\infty}(1-2q)^h x^h = \sum_{h=1}^{\infty}\left(\frac{1}{2} - \frac{(1-2q)^h}{2}\right)x^h.$$

We conclude that $T(h) = \frac{1}{2} - \frac{(1-2q)^h}{2}$, as claimed.

□

We have all the necessary tools to be able to take a big step forward towards the proof of Theorem 3.1.1.

Proposition 3.1.5. *Let u, f, e and c be random elements of R, in such a way that $\mathrm{w}(u) = w_u, \mathrm{w}(f) = w_f, \mathrm{w}(e) = w_e$ and $\mathrm{w}(c) = w_c$, respectively. Let $v \in \mathrm{Supp}(c)$, define $s := uf + ce$ and consider the expression*

$$x^{-v}s = e_i + x^{-v}u_i f_i + \sum_{l \in \mathrm{Supp}(c)\setminus\{v\}} x^{l-v}e_i.$$

Assume that the random variables $\sum_{l\in\mathrm{Supp}(c)\setminus\{v\}} x^{l-v}e$ are independently distributed, as l and v vary. Then, for $j \in [k-1]$, we have that $(x^{-v}s)_j$ is distributed as a Bernoulli random variable with parameter $p = \mathbb{P}((x^{-v}s)_j = 1)$ equal to:

$$p = \prod_{i=1}^{3} p_i + \sum_{i=1}^{3} p_i \prod_{j\in[3], j\neq i}(1-p_j), \tag{3.7}$$

where

$$p_1 = w_e/k, \qquad p_2 = (1-(1-2w_e/k)^{w_c-1})/2$$

and

$$p_3 = \frac{1}{\binom{k}{w_u}\binom{k}{w_f}} \sum_{\substack{1\le l\le \min(w_u,w_f) \\ l \text{ odd}}} \binom{k}{l}\binom{k-l}{w_u-l}\binom{k-w_u}{w_f-l}.$$

Proof. Write

$$(d)_j = \left(\sum_{v\in\mathrm{Supp}(c)} x^{-v}s\right)_j = \sum_{v\in\mathrm{Supp}(c)} \left(x^{-v}s\right)_j$$

Fix the value of $v \in \mathrm{Supp}(c)$ and consider the summand $(x^{-v}s)_j$. We can expand it as in (3.2) as

$$(x^{-v}s)_j = (e)_j + (x^{-v}uf)_j + \left(\sum_{l \in \mathrm{Supp}(c) \setminus \{v\}} x^{l-v} e \right)_j . \tag{3.8}$$

The random variable $(e)_j$ is Bernoulli distributed with parameter

$$p_1 = w_e/k.$$

Note that $(x^{l-v}e)_j$ and $(e)_j$ are identically distributed, as the coefficient vector of the former is the right circular shift of the coefficient vector of the latter, by $l-v$ positions. Therefore, from Lemma 3.1.4 we obtain that $\left(\sum_{l \in \mathrm{Supp}(c) \setminus \{v\}} x^{l-v} e\right)_j$ is Bernoulli distributed with parameter

$$p_2 = (1 - (1 - 2w_e/k)^{w_c - 1})/2.$$

Finally, from Lemma 3.1.2 and Remark 3.1.3 we have that the random variable $(x^{-v}uf)_j$ is Bernoulli distributed with parameter

$$p_3 = \frac{1}{\binom{k}{w_u}\binom{k}{w_f}} \sum_{\substack{1 \le l \le \min(w_u, w_f) \\ l \text{ odd}}} \binom{k}{l} \binom{k-l}{w_u - l} \binom{k - w_u}{w_f - l}.$$

Bearing in mind that we are considering events over $\mathbb{F}_2$, the sum of the three random variables is Bernoulli distributed with parameter

$$p = \prod_{i=1}^{3} p_i + \sum_{i=1}^{3} p_i \prod_{j \in [3], j \neq i} (1 - p_j),$$

i.e. $(x^{-v}s)_j$ succeeds if either only one or all the three variables produce success. □

At this point, the proof of Theorem 3.1.1 is straightforward.

Proof of Theorem 3.1.1. The statement directly follows from Prop. 3.1.8 and the aforementioned assumptions of independence of the random variables involved in the sum. □

In the settings of our attack, we are going to specialize Theorem 3.1.1 to the case where $p_1 \in \{0, 1\}$. This allows us to study the effect of the j-th coordinate of the vector e_i on the distribution of the j-th coordinate of d_i. Notice that, to have a clear understanding of the behavior of the j-th component of d, as defined in Eq. 3.3, we needed to sum the $(x^{-v}s)_j$ random variables over all the $v \in \mathrm{Supp}(c)$, where in this case the events are intended over $\mathbb{Z}$. While modelling this, we made the simplifying assumption that the random variables $x^{-v}u_i$ and $\sum_{l \in \mathrm{Supp}(c) \setminus \{v\}} x^l e$ are independently distributed, as v varies in the support of c. Experimental results in support of this assumption will be provided, showing that this assumption still manages to satisfactorily capture the behavior of d, and, more importantly, it does not affect the outcome of the cryptanalysis. Figure 3.2 gives a visual representation of how Theorem 1 approximates the behavior of d_i.

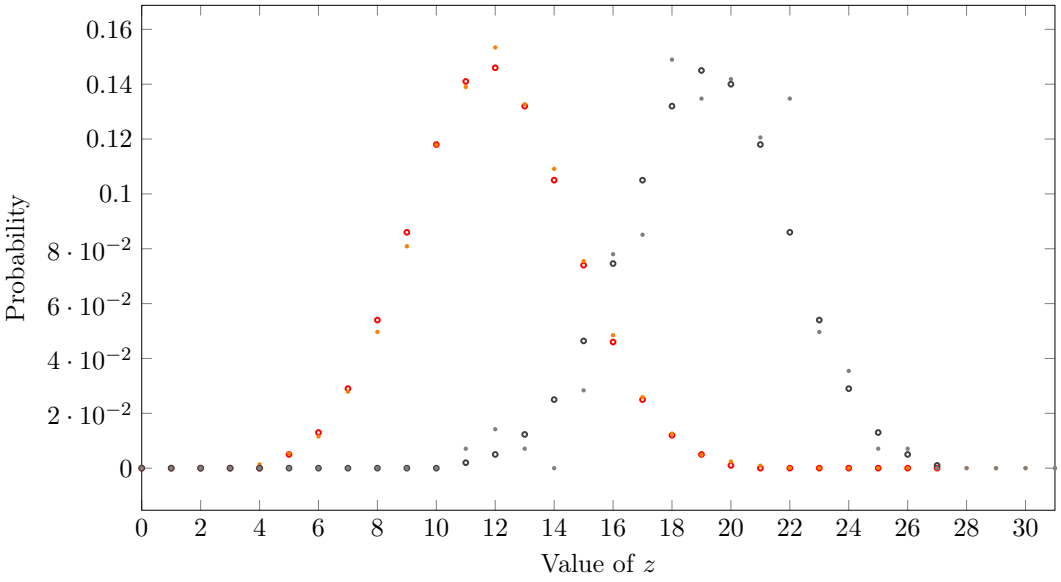

Figure 3.2: In red (resp. dark gray) the theoretical estimates $\mathbb{P}(d_i)_j = h \mid (e_i)_j = 0)$ (resp. $\mathbb{P}(d_i)_j = h \mid (e_i)_j = 1)$), as h varies in the interval $[w_c]$, for security level 128. In orange (resp. light gray) the associated experimental results obtained for a random instance of HWQCS with 128 bit of security.

Recovering k zero bits of e

As mentioned in subsection 3.1.2, we aim at finding k zero entries in e. The columns of H corresponding to the remaining k positions will constitute a square $k \times k$ matrix which, in the case of being invertible, can be used to calculate the entire value $e = (e_1, e_2)$. The idea is to use the knowledge of the distribution of the entries $(d_i)_j$, provided in Theorem 3.1.1, and find $\lceil k/2 \rceil$ error-free positions in both the left side e_1 and the right side e_2 of e, adding up to (at least) k error-free positions. We search for an optimal threshold value τ such that the probability $\mathbb{P}((e_i)_j = 0 \mid (d_i)_j < \tau)$ is large enough. For our attack, we choose τ as follows:

$$\tau := \max \left\{ h \in [w_c] \mid w_e \sum_{l=0}^{h} \mathbb{P}((d_i)_j = l \mid (e_i)_j = 1) < 1 \right\}, \tag{3.9}$$

i.e. the largest integer such that, for each signature, the expected number of entries j such that $(e_i)_j = 1$ and $(d_i)_j < \tau$ is less than 1. This means that, on average, all the coordinates j such that $((d_i)_j) < \tau$ will be error free. Let N_0 and N_1 be the expected number of error-free positions and error positions that we can find by taking all j with $(d_i)_j < \tau$, respectively. Then

$$N_0 := (k - w_e) \sum_{h=0}^{\tau-1} \mathbb{P}((d_i)_j = h \mid (e_i)_j = 0),$$

and

$$N_1 := (w_e) \sum_{h=0}^{\tau-1} \mathbb{P}((d_i)_j = h \mid (e_i)_j = 1).$$

Therefore, the probability that $(e_i)_j = 0$ for all $j \in [k]$ such that $(d_i)_j < \tau$ is given by

$$p_{succ} := \prod_{j\in[k],(d_i)_j<\tau} \mathbb{P}((e_i)_j = 0 \mid (d_i)_j < \tau) = \left(\frac{N_0}{N_0+N_1}\right)^{N_0+N_1}.$$

The values for τ, N_0, N_1 and of p_{succ} for each suggested parameter set of HWQCS are given in Table 3.2. These values indicate how many error-free bits of e_i we can expect to recover with our strategy. Surprisingly enough, for security level 256 we can reconstruct more than $\lceil k/2 \rceil$ error-free positions of e_i. The same does not hold for levels 128 and 192, meaning that we need to guess $\lceil k/2 \rceil - N_0$ error-free positions on each side of e.

Security level	$\lceil \frac{k}{2} \rceil$	τ	N_0	N_1	p_{succ}	$\lceil \frac{k}{2} \rceil - N_0$
128	6270	12	5564.3997	0.3995	0.6707	706
192	9459	15	7574.2963	0.2521	0.7771	1885
256	12709	21	13560.6279	0.3457	0.7077	-

Table 3.2: For each security level, we report the threshold value τ, the values N_0 and N_1 of zero and one bits that we expect to find among the j's such that $(\mathbf{d}_i)_j < \tau$, and the probability p_{succ} that this event occurs. The value $\lceil \frac{k}{2} \rceil - N_0$ is the number of positions that we still need to guess to fully reconstruct $\lceil k/2 \rceil$ entries.

Table 3.6 reports the probability distribution and expected values using our approximation of $(d_i)_j$ given by Theorem 3.1.1.

Remark 3.1.6. We are considering $\lceil k/2 \rceil$ in order to make sure we are recovering enough positions as k is an odd number for all three parameter sets. This implies that the probabilities we are computing are actually underestimates. Also, for security 256 we might select a subset of $\lceil k/2 \rceil$ of the N_0 expected error free positions, which would increase the value of the success probability to $p_{succ} = 0.7233$. We will exploit this in subsection 3.1.4 to give a speed-up of our attack.

Remark 3.1.7. The security level 256 would have values as reported in Table 3.1.7.

Security level	$\lceil \frac{k}{2} \rceil$	τ	N_0	N_1	p_{succ}
256	12709	22	16336.7984	0.8193	0.4407

Table 3.3: Value of τ as given in Equation 3.9, and the values of N_0 and N_1 of zero and one bits that we expect to find among the j's such that $(\mathbf{d}_i)_j < \tau$, as well as the probability p_{succ} that this event occurs.

As mentioned at the beginning of this subsection, we are searching for a total of $\lceil k/2 \rceil$ error free entries in e_i. The value of N_0 for security level 256 with $\tau = 22$, accounts for many more positions than actually needed, with somewhat smaller success probability compared to the levels 128 and 192. Decreasing the value of τ to 21 still gives N_0 larger than $\lceil k/2 \rceil$ and increases the success probability by more than 0.25.

We are left with the problem of finding the missing $\lceil \frac{k}{2} \rceil - N_0$ zero positions of e_i for security levels 128 and 192. According to the behavior of the $(d_i)_j$ entries, we aim at finding these values among the j's such that $(d_i)_j = \tau$, i.e. the set of positions that have

the least probability of containing an error, after those we already chose. Let M_0 and M_1 be the expected number of error free and error positions j such that $(d_i)_j = \tau$, respectively. Then

$$M_0 = (k - w_e)\mathbb{P}((d_i)_j = \tau \mid (e_i)_j = 0)$$

and

$$M_1 = (w_e)\mathbb{P}((d_i)_j = \tau \mid (e_i)_j = 1).$$

Therefore the probability such that $\lceil \frac{k}{2} \rceil - N_0$ randomly chosen positions j such that $(d_i)_j = \tau$ will be error free is given by

$$q_{succ} := \left(\frac{M_0}{M_0 + M_1} \right)^{\lceil \frac{k}{2} \rceil - N_0}.$$

The values of M_0, M_1 and q_{succ} for levels 128 and 192 are reported in Table 3.4.

Security level	$\lceil \frac{k}{2} \rceil - N_0$	M_0	M_1	q_{succ}
128	706	1806.5588	0.7402	0.7491
192	1885	2433.8479	0.4371	0.7129

Table 3.4: For security parameters 128 and 192, $\lceil \frac{k}{2} \rceil - N_0$ denotes the number of entries which is still necessary to guess to perform our attack. Values M_0 and M_1 represent the expected number of error free and error positions among the j's such that $(\mathbf{d}_i)_j = \tau$.

The overall probability of a correct recovery of $\lceil k/2 \rceil$ zero bits of e_i is then

$$p_{tot} := p_{succ} \cdot q_{succ}.$$

For the case of security level 256 we consider $q_{succ} = 1$ as $N_0 > \lceil \frac{k}{2} \rceil$. Applying this technique on both e_1 and e_2 we can recover k error free positions of e with probability of success p_{tot}^2. Values for each security level are reported in Table 3.5.

Security level	k	p_{tot}^2
128	12539	0.2524
192	18917	0.3070
256	25417	0.5008

Table 3.5: For each security level, k and p_{tot}^2 are the number of error free positions we need to recover and the probability of recovering them correctly, respectively.

In other words, Table 3.5 says that we can successfully recover k error free coordinates of e of around one fourth of the signatures of security level 128, slightly less than one third of those of level 192 and of half of the signatures of level 256. In the next subsection, we show how to completely reconstruct e using the k error free positions we recovered in this subsection.

h	$\mathbb{P}((d_i)_j = h \mid (e_i)_j = 0)$	Exp. number of entries	$\mathbb{P}((d_i)_j = h \mid (e_i)_j = 1)$	Exp. number of entries
0	$3.078 \cdot 10^{-7}$	0.004	$1.248 \cdot 10^{-13}$	$1.759 \cdot 10^{-11}$
1	$5.935 \cdot 10^{-6}$	0.074	$6.220 \cdot 10^{-12}$	$8.770 \cdot 10^{-10}$
2	$0.006 \cdot 10^{-2}$	0.686	$1.500 \cdot 10^{-10}$	$2.115 \cdot 10^{-8}$
3	$0.003 \cdot 10^{-1}$	4.128	$2.331 \cdot 10^{-9}$	$3.286 \cdot 10^{-7}$
4	0.001	17.973	$2.623 \cdot 10^{-8}$	$3.698 \cdot 10^{-6}$
5	0.005	60.371	$2.277 \cdot 10^{-7}$	$0.003 \cdot 10^{-2}$
6	0.013	162.726	$1.586 \cdot 10^{-6}$	$0.002 \cdot 10^{-1}$
7	0.029	361.500	$9.109 \cdot 10^{-6}$	0.001
8	0.054	674.586	$0.004 \cdot 10^{-2}$	0.006
9	0.086	1072.334	$0.001 \cdot 10^{-1}$	0.025
10	0.118	1467.441	$0.006 \cdot 10^{-1}$	0.090
11	0.141	1742.590	0.002	0.276
12	0.146	1806.558	0.005	0.740
13	0.132	1642.366	0.0123	1.739
14	0.105	1313.477	0.025	3.595
15	0.074	925.951	0.0464	6.550
16	0.046	575.964	0.0746	10.530
17	0.025	316.115	0.105	14.937
18	0.012	152.936	0.132	18.678
19	0.005	65.088	0.145	20.545
20	0.001	24.292	0.140	19.818
21	0.0006	7.914	0.118	16.688
22	0.0001	2.237	0.086	12.195
23	0.00004	0.544	0.054	7.671
24	$9.109 \cdot 10^{-6}$	0.112	0.029	4.11
25	$1.586 \cdot 10^{-6}$	0.019	0.013	1.850
26	$2.277 \cdot 10^{-7}$	0.002	0.005	0.686
27	$2.623 \cdot 10^{-8}$	$0.003 \cdot 10^{-1}$	0.001	0.204

Table 3.6: Values of the distribution of $(d_i)_j$ for security level 128. For every h, we report the probability that $(d_i)_j = h$, conditioning on the event $(e_i)_j = 0, 1$. Two more columns report the expected number of entries j associated to $(d_i)_j = h$.

Completing the reconstruction of e

In this subsection, we exploit the analysis performed so far to fully reconstruct the ephemeral values e_1, e_2 with a certain probability. Recall that

$$b = \mathbf{H}e^\top, \tag{3.10}$$

where $e = (e_1, e_2)$ and $\mathbf{H} = \big(\mathsf{circ}(h), \mathsf{circ}(h)^{-1}\big)$.

Let $J \subset \{0, \dots, 2k-1\}$ the set of k positions recovered using the strategy outlined in Subsection 3.1.2 and let $I := \{0, \dots, 2k-1\} \setminus J$. Finally, let $\mathbf{H}_I$ be the submatrix of $\mathbf{H}$ which consists of the columns indexed by I. We can treat $\mathbf{H}_I$ as a random matrix in $\mathbb{F}_2^{k \times k}$. Assume that $\mathbf{H}_I$ is invertible. Then given the syndrome equation $b = \mathbf{H}e^\top$ of e as in Sign, we can compute $\bar{e} = \mathbf{H}_I^{-1} b$ and thus reconstruct $e = (e_0, \dots, e_{2k-1})$ as

$$e_h = \begin{cases} \bar{e}_h & \text{if} \quad h = i \text{ for some } i \in I \\ 0 & \text{otherwise,} \end{cases} \tag{3.11}$$

for each $h \in \{0, \dots, 2k-1\}$. The probability for a random $k \times k$ matrix to be invertible is given by Proposition 3.1.8.

Proposition 3.1.8. *Let* $\mathbf{H} \in \mathbb{F}_2^{k\times k}$ *be a random square matrix. The probability that* $\mathbf{H}$ *is invertible is given by*

$$p_{inv} := \prod_{i=0}^{k-1}\left(1 - \frac{1}{2^{k-i}}\right). \tag{3.12}$$

Proof. For $k = 1$ the result is trivial. Suppose k is greater than one. If we have $k-1$ independent vectors, the probability that the k-th vector is independent of the previous is $(2^k - 2^{k-1})/2^k$. Multiplying the probability for which these $k-1$ vectors are independent and the probability that the k-th vector is independent of the others yields

$$\frac{2^k - 2^{k-1}}{2^k} \cdot \prod_{i=0}^{k-2}\left(1 - \frac{1}{2^{k-i}}\right) = \prod_{i=0}^{k-1}\left(1 - \frac{1}{2^{k-i}}\right).$$

□

For each set of suggested parameters for HWQCS, the value of the probability of $\mathbf{H}_I$ being invertible is $p_{inv} := 0.2888$.

Remark 3.1.9. The code used to compute data for this subsection can be found at https://github.com/triki96/Cryptanalysis-of-HWQCS

3.1.3 Universal Forgery

We exploit the reconstruction of e_1, e_2 to mount a universal forgery attack. Suppose that an adversary $\mathcal{A}$ intercepts a signature (c, b, s_1, s_2) used by an honest signer to sign a given message m. Note that:

$$\begin{aligned} b &= e_1 h + e_2 h^{-1}, \\ c &= H_{w_c}(m, b, u_1 f_2 - u_2 f_1, \mathsf{pk}), \\ s_i &= u_i f_i + c e_i. \end{aligned}$$

Now, suppose that $\mathcal{A}$ is given a random message m', and that it is asked to sign that message. The adversary needs $\sigma' = (c', b', s_1', s_2')$ which satisfies $\mathsf{Vf}(m', \mathsf{pk}, \sigma') = 1$. Therefore, $\mathcal{A}$ computes the signature in the following way:

$$\begin{aligned} b' &= b, \\ c' &= H_{w_c}(m', b', s_1 h + s_2 h^{-1} - cb, \mathsf{pk}), \\ s_i' &= s_i - c e_i + c' e_i = u_i f_i + c' e_i. \end{aligned}$$

The verifier computes

$$\begin{aligned} t' &= s_1' h + s_2' h^{-1} - c' b' \\ &= s_1' h + s_2' h^{-1} - c' b \\ &= (u_1 f_1 + c' e_1) h + (u_2 f_2 + c' e_2) h^{-1} - c'(e_1 h + e_2 h^{-1}) \\ &= u_1 f_2 + u_2 f_1. \end{aligned}$$

Clearly $t' \neq 0$ and $w(t') \leq w_t$, as this is the same value computed in the verification phase of the first sign. It remains to check whether $c'' := H_{w_c}(m', b', t', \mathsf{pk})$ is equal to c', but

$$c'' = H_{w_c}(m', b', t', \mathsf{pk}) = H_{w_c}(m', b, u_1 f_2 + u_2 f_1, \mathsf{pk}) = c'.$$

This concludes the forgery. Algorithm 11 is a probabilistic algorithm that sequentially analyzes the HWQCS signatures to retrieve the ephemeral values e_1, e_2 and forge a new signature for m'.

Algorithm 11 Attack on HWQCS

Input $\mathsf{pk} = (h)$ and τ according to Table 3.2.
Output True/False.

function ATTACKHWQCS(h, τ)
  Compute $\mathbf{H} = \left(\mathsf{circ}(h), \mathsf{circ}(h)^{-1}\right)$;
  Request $(m, \sigma) = (m, (c, b, s_1, s_2))$;
  Compute $d_i := \sum_{v \in \mathrm{Supp}(c)} x^{-v} s_i \in \mathbb{Z}[x]$;
  Set
  - $J_{1,1} := \{j \in [k-1] \mid (d_1)_j < \tau\}$ and
  - $J_{1,2} := \{j \in [k-1] \mid (d_2)_j < \tau\}$;

  Set random
  - $J_{2,1} \subset \{j \in [k-1] \mid (d_1)_j = \tau\}$ and
  - $J_{2,2} \subset \{j \in [k-1] \mid (d_2)_j = \tau\}$

  of size $\lceil k/2 \rceil - \#J_1$;
  Set $I := [2k-1] \setminus (J_{1,1} \cup J_{1,2} \cup J_{2,1} \cup J_{2,2})$;
  Set $\mathbf{H}_I$ the submatrix of $\mathbf{H}$ made by columns of $\mathbf{H}$ indexed by I;
  if $\mathbf{H}_i$ is not invertible **then**
    Go to (3);
  else
    Compute $\bar{e} = \mathbf{H}_I^{-1} b$;
    Set $e' = (e'_0, \ldots, e'_{2k-1}) = (e'_1, e'_2)$ as

$$e'_h = \begin{cases} \bar{e}_h & \text{if} \quad h = i \text{ for some } i \in I, \\ 0 & \text{otherwise.} \end{cases}$$

  end if
  Let m' be a new message to be signed;
  Compute $b' = b$;
  Compute $c' = H_{w_c}(m', b', (s_1 h + s_2 h^{-1} - c b', h)$;
  Compute $s'_i = s_i - c e'_i + c' e'_i$;
  if $\mathsf{Vf}(m', h, (c', b', s'_1, s'_2))$ **then**
    1. Return true;
  else
    1. Go to (3);
  end if
end function

3.1.4 Complexity of the Attack

We estimate the success probability of Algorithm 11. The attack succeeds in reconstructing the data needed for a universal forgery if we can correctly obtain k error-free positions of e and if the matrix $\mathbf{H}_I$, which depends on the recovered positions, is invertible. We computed the success probabilities p_{tot}^2 and p_{inv} of both events in Subsections 3.1.2 and 3.1.2, respectively. Therefore, the success probability of Algorithm 11 is given by the product

$$p_{break} := p_{tot}^2 \cdot p_{inv},$$

and thus the expected number of attempts to achieve success is given by $\lceil 1/p_{break} \rceil$. The cost of each attempt is dominated by the inversion of the matrix $\mathbf{H}_I \in \mathbb{F}_2^{k \times k}$, which is in $\mathcal{O}(k^{2.37})$. Therefore, Algorithm 11 runs in time $\mathcal{O}(2^{\log(k^{2.37}/p_{break})})$. The values of p_{break}, the expected number of needed attempts, and the cost of the attack for each security level of HWQCS are reported in Table 3.7.

Security level	p_{break}	Number of signatures	Cost
128	0.0727	14	36.04
192	0.0887	12	37.25
256	0.1446	7	37.48

Table 3.7: For each security level, p_{break} denotes the probability that our attack successfully retrieves the ephemeral values $e1, e2$ from a given signature. The expected number of signatures for our attack to succeeds and the $\log_2$ of the cost of the attack are reported.

We implemented an unoptimized version of our attack in SageMath and ran it on a Linux Mint virtual machine. The code stops after recomputing e' and checking it against the real error vector e. If the two match, the forgery succeeds. Table 3.8 reports the average, in 10 runs of the attack for each security level, of the number of signatures needed to recover e and the average time consumed analyzing each signature.

Security level	Number of signatures	Consumed time (s)
128	17	40.33
192	9	92.50
256	3	133.25

Table 3.8: Average number of signatures needed to mount our universal forgery attack, and average time needed to analyse each signature, on 10 runs of our attack for each security level.

Further Optimizations

Due to the large value of N_0 as an effect of our choice of $\tau = 21$ for security level 256, see Table 3.2, it is possible to slightly improve our attack by intercepting only one signature. Indeed, we can take advantage of this surplus of potentially error-free positions that we obtain by taking all j such that $(d_i)_j < \tau$. The idea is to randomly pick a subset J_i of cardinality $\lceil k/2 \rceil$ of the $N_0 + N_1$ positions such that $(d_i)_j < \tau$ for both $i = 1, 2$. In other words, we are applying plain information set decoding on a potentially error-free set of positions to find an invertible submatrix of $\mathbf{H}$. Note that J_i has probability of being error-free equal to $(N_0/(N_0 + N_1))^{\lceil k/2 \rceil}$. Since we do this for both the left and right hand side of e, the overall probability of $J := J_1 \cup J_2$ being error-free is $p := (N_0/(N_0 + N_1))^k$. According to the values in Table 3.2, we have $p = 0.5231$. Set now $I := [2k - 1] \setminus J$ and let $\mathbf{H}_I$ be the matrix consisting of the columns of $\mathbf{H}$ indexed by I. According to Proposition 3.1.8, the probability that $\mathbf{H}_I$ is invertible is $p_{inv} = 0.2888$. In the case where $\mathbf{H}_I$ is invertible, we continue in the same way as in our attack, leading to a forgery. The success probability of recovering the correct e then becomes

$$p_{break} = p \cdot p_{inv} = 0.1511,$$

which gives an improvement of 0.065 on the probability 0.1446 reported in Table 3.7. The expected number of attempts to find an invertible matrix $\mathbf{H}_I$ is $\lceil 1/p_{break} \rceil = 5$, which improves on table 3.7 by 2.

As a conclusive remark, through a detailed analysis of the information leakage on ephemeral keys during the signing process, we demonstrated how partial secret data can be efficiently reconstructed from a limited number of signatures, regardless of the chosen security level (128, 192, or 256 bits). This enabled the development of a universal forgery attack that exploits this leakage to produce valid signatures for arbitrary messages. We proposed a probabilistic algorithm, implemented in SageMath, which empirically supports the effectiveness and efficiency of our attack strategy, particularly at the 256-bit security level, where an additional optimization further accelerates the recovery of secret data.

3.2 An Unsuccessful Cryptanalysis: Wave

This work represents an attempt to construct a new distinguisher for normalized generalized $(U, U+V)$ codes. This is the class of codes used by Wave [DAST19], a recently proposed "hash and sign" digital signature scheme. Our attempt heavily exploits the weight distribution of these codes, for which low-weight codewords have a different distribution from that of a random code. In particular, the security of Wave [DAST19] is based on the following two problems:

1. Decoding One Out of Many (DOOM) Problem [JJ02];
2. indistinguishability of permuted normalized generalized $(U, U+V)$ codes.

The Decoding One Out Of Many Problem is a variant of the Syndrome Decoding Problem, where multiple instances are considered at once. We have already introduced this problem in Def. 2.1.8, but we choose to briefly recall it here, for the further purpose of fixing the notation we will use later in the discussion.

> *Input*: a prime q, $w \in \mathbb{N}$, $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$, m syndromes $s_1, \ldots, s_m \in \mathbb{F}_q^{n-k}$.
>
> *Output*: a word x s.t. $\mathrm{w}(x) = w$ and $s_i^\top = \mathbf{H}x^\top$ for some $i \in \{1, \ldots, m\}$.

Normalized generalized $(U, U+V)$ codes are a specific type of generalized $(U, U+V)$ codes, which in turn are a generalization of $(U, U+V)$ codes, and the indistinguishability of the latter is an NP-complete problem [DAST17] if the dimensions k_U and k_V of the two building codes are such that $k_U < k_V$. However, in order for the Wave decoder to work properly, it must be $k_U \geq k_V$. In this case, the problem is no longer shown to be NP-complete. Furthermore, in the special case of $q = 2$ there is an algorithm which is able to efficiently solve the distinguishing problem [DAST17]. This algorithm applies only to the binary case, and this motivates the choice for Wave to work with a non-binary finite field. In the midst of the turbulent history that has seen many hash&Sign code-based signature schemes as protagonists, Wave is among the few for which cryptanalysis does not seem to have shown any weakness. A question that would certainly be interesting to answer is the following.

Is the Wave digital signature scheme secure?

As with the previous section, we have voluntarily decided to leave this question vague, meaning that the term *secure* should be more strictly defined. Differently from the previous section, in this case, saying *secure*, we mean *formally* secure. That is, we ask whether the security assumptions holds. In particular, our analysis focuses on the indistinguishability of the codes used by this scheme. In an attempt to answer this question in the affirmative, we tried to devise a distinguisher that was capable of recognizing with good probability whether a given input code came from a random distribution, or whether it was a permuted generalized $(U, U+V)$ code. In this section we will describe our attempt, and the main idea on which it is based, showing the great obstacle we ran into and which did not allow us to clearly answer the main question. Despite the negative outcome, we still decided to include this work in the thesis, as the the follow-up presented in the following section arose in a very natural way from this first attempt.

3.2.1 Normalized Generalized $(U, U+V)$ Codes

The codes employed in Wave are called normalized generalized $(U, U+V)$ codes, and are defined as follows.

Definition 3.2.1 (Normalized Generalized $(U, U+V)$ Codes)**.** Let $U, V \subseteq \mathbb{F}_q^{n/2}$ be two linear codes with length $n/2$ and respective dimensions k_U and k_V. Let $a, b, c, d \in \mathbb{F}_q^{n/2}$ be four vectors with length $n/2$ and such that

- for every $i \in \{1, 2, \ldots, n/2\}$, $a_i \neq 0$ and $c_i \neq 0$;
- for every $i \in \{1, 2, \ldots, n/2\}$, $a_i d_i - b_i c_i = 1$.

Let $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D} \in \mathbb{F}_q^{n/2 \times n/2}$ be the diagonal matrices obtained from the vectors a, b, c and d. Then, the *normalized generalized* $(U, U+V)$ *code* $\mathcal{C}(U, V)$ is defined as the linear code

$$\mathcal{C}(U, V) = \{(u\mathbf{A} + v\mathbf{B}\ ,\ u\mathbf{C} + v\mathbf{D}) \,|u \in U,\ v \in V\}\,,$$

and has length n and dimension $k = k_U + k_V$.

A generator matrix $\mathbf{G}$ and a parity-check matrix $\mathbf{H}$ for $\mathcal{C}(U, V)$ have the following form:

$$\mathbf{G} = \begin{pmatrix} \mathbf{G}_U\mathbf{A} & \mathbf{G}_U\mathbf{C} \\ \mathbf{G}_V\mathbf{B} & \mathbf{G}_V\mathbf{D} \end{pmatrix} \qquad \mathbf{H} = \begin{pmatrix} \mathbf{H}_U\mathbf{D} & -\mathbf{H}_U\mathbf{B} \\ -\mathbf{H}_V\mathbf{C} & \mathbf{H}_V\mathbf{A} \end{pmatrix}, \tag{3.13}$$

where $\mathbf{G}_U \in \mathbb{F}_q^{k_U \times n/2}$ and $\mathbf{G}_V \in \mathbb{F}_q^{k_V \times n/2}$ are the generator matrices for U and V, $\mathbf{H}_U \in \mathbb{F}_q^{r_U \times n/2}$ and $\mathbf{H}_V \in \mathbb{F}_q^{r_V \times n/2}$ are parity-check matrices for U and V, respectively, and $r_U = n/2 - k_U$, $r_V = n/2 - k_V$.

We recall that for a generic code C and a given integer w, we denoted by $N_C(w)$ the number of codewords of C of weight w. For a random linear code C of dimension k over $\mathbb{F}_q^n$, it holds that

$$N_C(w) = \frac{\binom{n}{w}(q-1)^w}{q^{n-k}}.$$

This is due to the fact that, denoting with $\mathbf{H}$ the parity check matrix associated to C, for every non zero vector $x \in \mathbb{F}_q^n$ the product $\mathbf{H}x^\top$ spreads uniformly over $\mathbb{F}_q^{n-k}$, and the number of vectors of length n and weight w is exactly $\binom{n}{w}(q-1)^w$. For the sake of simplicity, in the following a normalized generalized $(U, U+V)$ code will always be denoted by $\mathcal{C}(U, U+V)$. In the following, we state a very useful result, which will be of fundamental importance in the construction of the distinguisher for normalized generalized $(U, U+V)$ codes.

Proposition 3.2.2. *Let $U, V \subseteq \mathbb{F}_q^{n/2}$ be random linear codes and let $\mathcal{C}(U, U+V) \subseteq \mathbb{F}_q^n$ be a normalized generalized $(U, U+V)$ code. Then*

$$N_{\mathcal{C}(U,U+V)}(2w) \geq N_U(w).$$

Proof. Notice that for every $u \in U$ of weight w the vector

$$(u, 0) \cdot \begin{pmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{B} & \mathbf{D} \end{pmatrix} = (u\mathbf{A}, u\mathbf{C})$$

lies in $\mathcal{C}$ and has weight $2w$. This is due to the fact that, by construction, both $\mathbf{A}$ and $\mathbf{C}$ are diagonal matrices with non-zero entries along the main diagonal, and so do not change the weight of the vector they are multiplied to. Finally, from [DAST19] we know that $\mathbf{G}$ is a bijection between $\{(u, v) \mid u \in U, v \in V\}$ and its image through $\mathbf{G}$, and the desired result follows easily. □

3.2.2 A Distinguisher Attempt

Notice that for the Wave decoder to work it is necessary that the dimensions k_U and k_V of U and V are such that $k_U \geq k_V$ (equivalently, $R_U \geq R_V$). According to this, to reach 128 bits of security, the following set of parameters is chosen:

$$n = 8492,\ k_U = 3558,\ k_V = 2047.$$

If the code V has rate $R_V \ll R_U$, then it happens that the minimum distance of V is much greater than that of U. Let d_U and d_V denote respectively the minimum distance of U and V. In our setting, the GV bound provides $d_U = 150$ and $d_V = 712$, and we know from Proposition 3.2.2 that, for every even w greater than 300, we can lower the bound $N_{\mathcal{C}(U,U+V)}(w)$ as $N_U(w/2)$. Notice that there exist several values of $w \geq d_U$ for which $N_V(w) \approx 0$. This means that, for low values of w, this quantity is also a good estimate of the exact number of codewords for $\mathcal{C}(U, U+V)$, since the words coming from any combination of $u \in U \setminus \{0\}$ and $v \in U \setminus \{0\}$ typically have higher weights.

This implies that the weight distribution of low-weight codewords of a normalized generalized $(U, U+V)$ code is very different from that of a random linear code. In fact, if C was a random code, thanks to Theorem 1.3.10, we can compute the associated GV bound as

$$d_C = \min\left\{ d \in \mathbb{N} \,\middle|\, \binom{n}{d}(q-1)^d q^{-r} \geq 1 \right\}.$$

Since random codes asymptotically reach this bound, we can set the minimum distance of C to be approximately $d_C \approx h_q^{-1}(1 - R_C)n$, where h_q is the q-ary entropy function and $R_C = k_C/n$. Notice that for a random linear code with the same parameters as Wave, we expect $d_{\mathcal{C}}$ to be equal to 788. Hence, for a random code, we expect $N_C(w) \approx 0$ for any $w < d_C$. Furthermore, for $w > d_C$, we expect that

$$N_{\mathcal{C}}(w) = \frac{\binom{n}{w}(q-1)^w}{q^{n-k}}.$$

Figure 3.5 shows the weight distribution for both types of code where w is small.

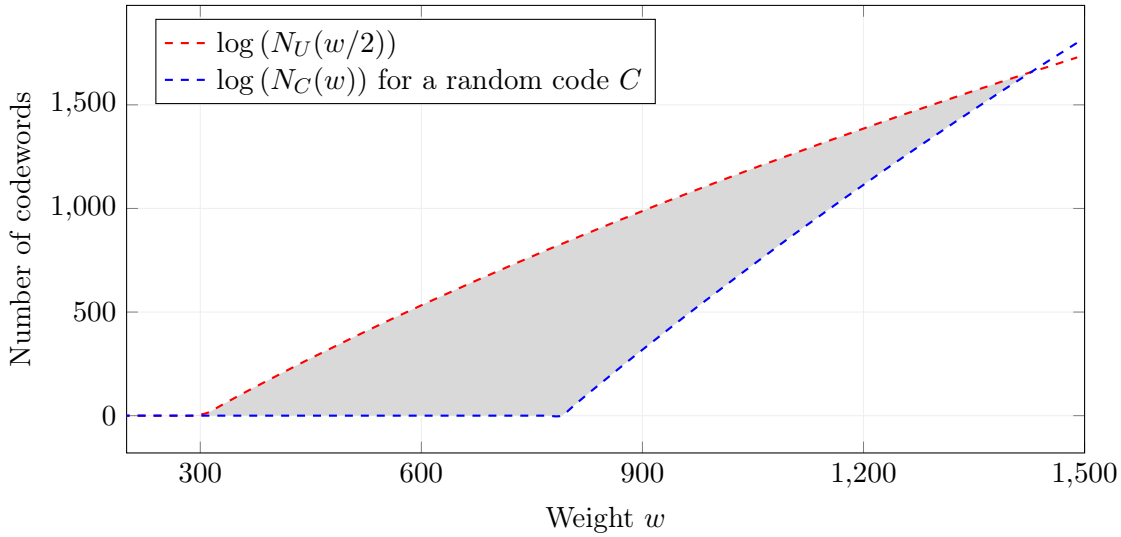

Figure 3.3: (log of) number of codewords of Wave hidden code and (log of) number of codewords of a random linear code with the same parameters.

This peculiarity of the code used by Wave opens the way to two types of attacks. First, since the code U has a minimum distance that is much smaller than d_C, there exists a wide range of weights $w \in [2d_U,\ d_C)$ for which w-codewords exist in $\mathcal{C}(U, U+V)$, but are not expected to exist in a truly random code C with the same parameters. A natural distinguisher that exploits this property could work as follows: on input a parity check $\mathbf{H}$ of a code $\mathcal{C}$, for $w \in [2d_U, d_C)$, call an ISD algorithm for N times, searching for a codeword with weight w in the code whose parity-check matrix is $\mathbf{H}$. If all calls are unsuccessful, output 0; if at least one call is successful, output 1. Unfortunately, the complexity of such a distinguisher is just prohibitive for Wave parameters. However, Figure 3.5 suggests another way to proceed: even if w is greater than $d_{\mathcal{C}}$, there is still a large interval where the number of w-codewords of a normalized generalized $(U, U+V)$ code is larger than the number of w-codewords of a random code $\mathcal{C}$. In essence, this means that the ISD algorithms work better in the first case. We tried to exploit the complexity of these algorithms to understand if we are dealing with a random code or not, in particular, for the distinguishing game we consider the following algorithm:

1. *Setup*:
   - sample $\mathbf{H}' \xleftarrow{\$} \mathbb{F}_q^{r\times n}$;
   - sample a random normalized generalized $(U, U+V)$ with redundancy r and length n with parity-check matrix $\mathbf{H}'_C$ as in (3.13). Sample a random full rank $\mathbf{S} \in \mathbb{F}_q^{r\times r}$ and a random permutation matrix $\mathbf{P} \in \mathbb{F}_q^{n\times n}$ and set $\mathbf{H}'' = \mathbf{S}\mathbf{H}'_C\mathbf{P}$;
   - sample at random $b \in \{0, 1\}$;
   - if $b = 0$, set $\mathbf{H} = \mathbf{H}'$, else set $\mathbf{H} = \mathbf{H}''$;
   - output $\mathbf{H}$.

2. *Distinguisher*:

- Fix an appropriate weight w, and compute the theoretical complexity of our ISD algorithm, both in the case of a random code and in the case of a normalized generalized $(U, U+V)$ code;
- Run a statistical test;
- Compare the experimental results with our previous estimates. If the results match the theoretical complexity of a normalized generalized $(U, U+V)$ code, the output is 1, otherwise the output is 0.

Complexity (Theoretical Estimates)

We compute the complexity of finding codewords of small weight w both in the case of a random code and in the case of a normalized generalized $(U, U+V)$ code. We take into consideration Stern's algorithm [Ste89], which is one of the most widely used ISD algorithms, as well as one of the fastest on a classical computer. Notice that Stern works with two additional parameters, which will be denoted as ℓ and p. According to this, the complexity of the algorithm is given by Theorem 1.3.23:

$$T_{\mathsf{S}} = \frac{n^3 + L + L^2/q^\ell}{p(n,k,w)}, \tag{3.14}$$

where

$$p(n,k,w) = \frac{\binom{(k+\ell)/2}{p}^2 \binom{n-k-\ell}{w-2p}}{\binom{n}{w}}, \qquad L = \binom{(k+\ell)/2}{p}(q-1)^p.$$

Since we are working with multiple solutions to our problem, we also have to take into account the expected number of codewords of weight w. In the case of a code $\mathcal{C}$, this quantity is given by $N_{\mathcal{C}}(w)$. Notice that $p(n,k,w)$ is the success probability of one iteration of the considered ISD algorithm. Since there are $N_C(w)$ codewords with the desired weight, and we are satisfied with any of them, we can consider a larger success probability, that is,

$$p^*(n,k,w) = 1 - (1 - p(n,k,w))^{N_C(w)} \approx \min\left\{1\,;\ N_C(w)p(n,k,w)\right\}.$$

Consequently, we can modify Stern's complexity as suggested by Proposition 1.3.21, obtaining

$$T_{\mathsf{S}} = \frac{n^3 + L + L^2/q^\ell}{p^*(n,k,w)}. \tag{3.15}$$

We have calculated the complexity of this algorithm for different parameters choices, ending in selecting $\ell = 29$ and $p = 4$. For this choice Stern complexity is depicted in Figure 3.4, and the best result for Wave $(U, U+V)$ code is achieved for $w = 970$, for which $T_{\mathsf{S}} \approx 2^{117}$.

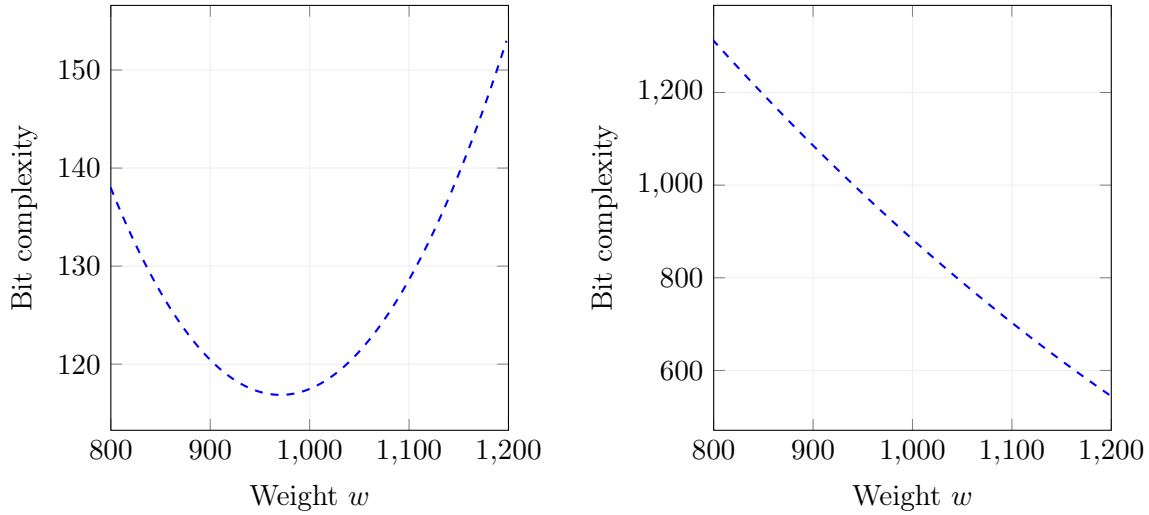

Figure 3.4: On the left: bit complexity of Stern's algorithm with fixed parameters $\ell = 29, p = 4$, applied to Wave $(U, U + V)$ code. On the right: the complexity of the same algorithm applied to a random linear code with the same parameters.

Complexity (Experimental Results)

In an attempt to validate our results, we simulated our algorithm by scaling Wave parameters. Surprisingly, the results deviate significantly from our estimate, making the behavior of the codes used in Wave apparently indistinguishable from random codes. In particular, in the Wave case, we expected that Stern's ISD could find a codeword with a lower complexity. In practice, this does not happen. Recall that Stern's algorithm chooses $k + \ell$ columns among n of (a permuted version of) $\mathbf{H}$. The probability that a codeword c of weight w has its non-null components arranged as:

- p components in the first $\frac{k+\ell}{2}$ entries,
- p components in the Second $\frac{k+\ell}{2}$ entries,
- $w - 2p$ components in the last $n - k - \ell$ entries,

is given by

$$p(n, k, w) = \frac{\binom{(k+\ell)/2}{p}\binom{(k+\ell)/2}{p}\binom{n-k-\ell}{w-2p}}{\binom{n}{w}}.$$

Notice that this holds just for *random codes*. Since we are considering weights for which the number of codewords $N_{\mathcal{C}}(w)$ is greater than 1, the probability that a solution for the small instance is also a solution for the big instance is given by

$$p^*(n, k, w) = 1 - (1 - p(n, k, w))^{N_{\mathcal{C}}(w)}$$

Notice that this holds only if the solutions are *independent*. Once we fix the values of w and p, we can estimate:

- the number of collisions (considering the parameters of the small instance);
- (using the above formula) the number of solutions for the complete instance.

Remark 3.2.3. According to our simulations, for random codes these checks are always verified (this leads us to think that the theoretical estimates are correct); however, for a normalized generalized $(U, U+V)$ code it happens that even if the number of collisions for the small instance matches the theoretical estimate, almost none of these collisions leads to a solution c for $\mathbf{H}c^\top = 0$. Said otherwise, once fixed a vector of weight $2p$ in the first $k+\ell$ entries, it is very unlikely that ISD is able to "complete it" to a solution for $\mathbf{H}c^\top = 0$ of weight w.

Where are we wrong? There are two options:

1. We fail to count the number of codewords $N_{\mathcal{C}}(w)$. However, in the worst case, our estimates are a lower bound for the number of w-codewords, so we would get better results.

2. We fail to consider the w-codewords to be independent. For the weights w that we are considering, the w-codewords all come from the code U, that is, they are all words of the type $(u\mathbf{A}, u\mathbf{C})$. We have seen that for random codes the theoretical estimates are correct, so the thing that creates problem is the structure of the code U inside the $(U, U+V)$ construction. In this case U is repeated two times, each time multiplied by a different diagonal matrix.

Notice that in an attempt to build a distinguisher, we used ISD algorithms in the context in which the number of codewords that we expect is greater than 1. ISD is designed to work in regimes where there is only one solution, and in this new context we can try to devise better algorithms. In the next section, we take advantage of the generalized ISD paradigm that we have already introduced, and take this basic difference into account. Starting from this new description, we will be able to provide new estimators for the weight distribution of a random code.

3.3 ISD for Weight Distribution

The problem of determining the weight distribution of a given linear code, that is, how many codewords of some weight does the code contain, has been studied for a long time. Such a quantity represents a sort of aggregate way to measure the ability of the code to correct errors in noisy channels: roughly speaking, if the weight distribution is centered around large values, then we expect the code to behave better. In the following, we state the main problem we are interested in.

Definition 3.3.1 (Weight Enumerator Problem)**.** Let $\mathbb{F}_q$ be a finite field, and let $k \leq n$ be positive integers. The *Weight Enumerator Problem* (WEP) is the problem defined as follows:

Input: a linear code $C \subseteq \mathbb{F}_q^n$ and $w \in \mathbb{N}$.

Output: $N_C(w)$.

Not only has this problem been known for a long time, but it also appears to be very difficult to solve, since the determination of the distribution implies the determination of the minimum distance. We will also consider the approximate version of the problem, that is, we are satisfied even if we are able to determine whether $N_C(w)$ lies in some (bounded) range.

Definition 3.3.2 (Approximate Weight Enumerator Problem)**.** Let $\mathbb{F}_q$ be a finite field, and let $k \leq n$ be positive integers. The *Approximate Weight Enumerator Problem* (ApproxWEP$^\varepsilon$) is the problem defined as follows:

Input: a linear code $C \subseteq \mathbb{F}_q^n$, $w \in \mathbb{N}$ and $\varepsilon \geq 0$.

Output: compute $N_C^*(w)$ so that $N_C(w)(1-\varepsilon) \leq N_C^*(w) \leq N_C(w)(1+\varepsilon)$.

In our case, interest arose in studying the codes underlying Wave, but there are many other cases of practical interest. For instance, Leon's algorithm [Leo88] requires to find all codewords having some fixed weight, which may not be the minimal one. This algorithm is used to compute the automorphism group of a linear code [Leo82], and finds applications in the resolution of the code-equivalence problem [Bou07]. For some codes, the associated weight distribution is known, but in general this problem is far from easy. We wondered the following.

Can we propose new algorithms to estimate
the weight distribution of a linear code?

In the following, we use ISD algorithms to build estimators for this quantity. The description of the algorithm is accompanied by a study of the statistical reliability and the time complexity. To do this, we analyze the behavior of ISD in the not so studied regime of large weights (say, much larger than the minimum distance).

Related works. Compare with [HMM05], where the authors describe how the success probability can be used as an estimator for the weight distribution of a code. The idea is based on the heuristic assumption that the codewords of a given code behave as random vectors of fixed weight, when considering the performances of ISD algorithms. Since the number of codewords has an impact on the success probability of each iteration, the authors propose to measure the success probability empirically and then use a simple formula to derive the number of codewords. This approach has a major limit in the fact that, for moderately large weights, the success probability saturates, and learning something about the number of codewords becomes unfeasible.

Our contribution. We exploit the general framework already introduced for ISD algorithms to consider the need for some distinct codewords of the same weight. Then, we improve on the approach in [HMM05] for two aspects. From a computational point of view, we consider an approach that can sometimes be faster than standard ISD. Then, we use quantities different from the simple success probability. Indeed, we also show that the average number of solutions produced can be used to obtain a good approximation for the number of codewords of a given weight. To verify the performances of our algorithm, we use it to obtain the weight distribution of some well-known codes.

3.3.1 Estimators for the Weight Distribution

The naive approach to solve WEP would be to list all the codewords of C and count the number of weight-w codewords. By omitting scalar multiples from the enumeration, this would cost

$$O(q^{k-1}) = O(q^k).$$

A better approach would be to enumerate only codewords whose weight is properly bounded. In fact, let $\mathbf{G}$ be a generator for C: one can put the matrix in systematic form and then

enumerate only the codewords obtained from a linear combination with weight less or equal than $\min\{w, k\}$. This would cost

$$\sum_{i=1}^{\min\{w,k\}} \binom{k}{i}(q-1)^i \approx \sum_{i=1}^{\min\{w,k\}} q^{kh_q(i/k)},$$

which asymptotically, taking the maximum of the considered terms, is

$$\begin{cases} q^{k \cdot h_q(w/k)} & \text{if } w < k, \\ q^k & \text{if } w \geq k. \end{cases} \tag{3.16}$$

The idea of using ISD as an estimator has first appeared in [HMM05]. The authors exploit the success probability of the algorithm to provide an estimate for the number of codewords of a fixed low weight in a code C. In case the code C contains $N_C(w)$ words of weight w, we can say that on average the number Y of codewords returned by ISD is

$$\mathbb{E}[|Y|] = N_C(w) \cdot p_{\text{ISD}}(w). \tag{3.17}$$

Since in [HMM05] ISD is modeled as an algorithm that can either fail or return only one codeword of fixed weight, $\mathbb{E}[|Y|]$ is interpreted as a measure of how successful a call to ISD is. Stated differently, in the case where $N_C(w)$ is greater than 1, the quantity $\mathbb{E}[|Y|] \in [0, 1]$ measures on average how many ISD iterations went well. This value can be observed with numerical simulations, and additionally depends on $N_C(w)$, indeed let B_w be the number of the weight w vectors which are found through m iterations of ISD. Then, the average ratio of founded codewords is given by B_w/t. Therefore, the following holds:

$$N_C(w) \cdot p_{\text{ISD}}(w) \approx \frac{B_w}{m}. \tag{3.18}$$

In other words, we dispose of something that we are able to compute and that depends on $N_C(w)$. So, launching ISD and keeping track of the average number of iterations before one correct codeword is returned, we are able to estimate $N_C(w)$ by simply reverting (3.18). The procedure is reported in Alg. 12.

Algorithm 12 Estimator from [HMM05]

Input : $\mathbf{H} \in \mathbb{F}_q^{r \times n}$, weight $w \in \mathbb{N}$, number $m \in \mathbb{N}$ of ISD calls.
Output : $N_C^*(w)$

Set $B_w \leftarrow 0$ // Number of successful calls to ISD
for $i \in \{1, \ldots, m\}$ **do**
  if $|\text{ISD}(\mathbf{H}, w)| \geq 1$ **then**
    $B_w \leftarrow B_w + 1$
  end if
end for
$N_C^*(w) \leftarrow B_w/(m \cdot p_{\text{ISD}}(w))$
return $N_C^*(w)$

While this approach is useful for low-weight codewords, it has a major limit for moderately large weights. In this case the value $N_C(w)$ grows very quickly and saturates the right side of Equation (3.18): we have $B_w \approx m$ and, from (3.18), we get that also the estimate saturates as well, since

$$N_C^*(w) \approx \frac{1}{p_{\text{ISD}}(w)}.$$

The most natural way to solve this problem would be to iterate the ISD algorithm several times, in order to have a reasonable number of samples to draw conclusions about the approximate value of B_w. However, this approach is computationally unfeasible for the rate of growth with which $N_C(w)$ increases. We describe a way to overcome this problem in the following.

Estimator I

We describe a simple, but rather effective, improvement on the method of [HMM05]. Our idea consists of using not only the average success probability, but also the codewords found (and their number) to obtain a good estimate for $N_C(w)$. The first new estimator we present is detailed in Algorithm 13. The approach is based on the fact that the average number of codewords found, per ISD call, is $\mathbb{E}[|Y|] = N_C(w) \cdot p_{\text{ISD}}(w)$. What the algorithm does is simply counting the number of codewords that are returned by one call to ISD and reverting (3.17). To reach a more granular estimate, we can run ISD m times and then compute $\frac{1}{m}\sum_{i=1}^{m} \widehat{N}_i$ (where $\widehat{N}_i$ is the number of weight-w codewords that have been found in the i-th call to ISD) as an estimate for $\mathbb{E}[|Y|]$.

Algorithm 13 Estimator I

Input : $\mathbf{H} \in \mathbb{F}_q^{r \times n}$, weight $w \in \mathbb{N}$, number $m \in \mathbb{N}$ of ISD calls
Output : $N_C^*(w)$

1: **for** $i \in \{1, \ldots, m\}$ **do**
2: $\quad Y \leftarrow \text{ISD}(\mathbf{H}, w)$
3: $\quad$ Set $\widehat{N}_i \leftarrow |Y|$ // Number of found codewords in i-th call to ISD
4: **end for**
5: $N_C^*(w) \leftarrow \frac{1}{m \cdot p_{\text{ISD}}(w)} \sum_{i=1}^{m} \widehat{N}_i$
6: **return** $N_C^*(w)$

Estimator II

The second estimator that we consider is shown in Algorithm 14. Denote by m the number of ISD calls that we make. The algorithm considers that, for any of the $N_C(w)$ codewords in C_w, the probability that it is found in at least one ISD call is $1 - \big(1 - p_{\text{ISD}}(w)\big)^m$. Hence, the average size of A (that is, the average number of distinct codewords) is

$$\mathbb{E}[|A|] = N_C(w)\Big(1 - \big(1 - p_{\text{ISD}}(w)\big)^m\Big).$$

So, we can obtain an estimate for $N_C(w)$ as

$$N_C^*(w) = \frac{|A|}{1 - \big(1 - p_{\text{ISD}}(w)\big)^m} \tag{3.19}$$

Algorithm 14 Estimator II

Input : $\mathbf{H} \in \mathbb{F}_q^{r \times n}$, weight $w \in \mathbb{N}$, number $m \in \mathbb{N}$ of ISD calls
Output : $N_C^*(w)$

$A \leftarrow \emptyset$ // The set of all found codewords
for $i \in \{1, \dots, m\}$ **do**
  $Y \leftarrow \mathrm{ISD}(\mathbf{H}, w)$
  Update $A \leftarrow A \cup Y$ // Update set of found codewords
end for
$N_C^*(w) \leftarrow \frac{|A|}{1-\left(1-p_{\mathrm{ISD}}(w)\right)^m}$
return $N_C^*(w)$

3.3.2 Comparison Between the Approaches

It is easy to see that both approaches I and II are somewhat more general than the one presented in [HMM05]. Let us first compare approach I with [HMM05]. If $p_{\mathrm{ISD}}(w) \cdot N_C(w) \ll 1$ (that is, when the estimate (1.5) does not saturate), we have

$$p^*_{\mathrm{ISD}}(w) \approx p_{\mathrm{ISD}}(w) \cdot N_C(w) = \mathbb{E}[|Y|].$$

So, our estimator behaves in the same way as the one in [HMM05]. Notice that the above approximation does not work when $p_{\mathrm{ISD}}(w) \cdot N_C(w)$ is moderately large. In such cases, as we have already argued, the estimator in [HMM05] saturates, while our Estimator I is able to return a valid value.

Example 3.3.3. To give an idea of this fact, we took into consideration a random linear code with $n = 100$ and $k = 50$. For this code we estimated the number of words following the distribution described in Thm. 1.3.18. In addition, for each weight w, we estimated $N_C(w)$ using the approach proposed in [HMM05] and approach 1 described above. For each of the two cases we used Stern, calling it 20 times and averaging the result. The final estimates are shown in Fig. 3.5.

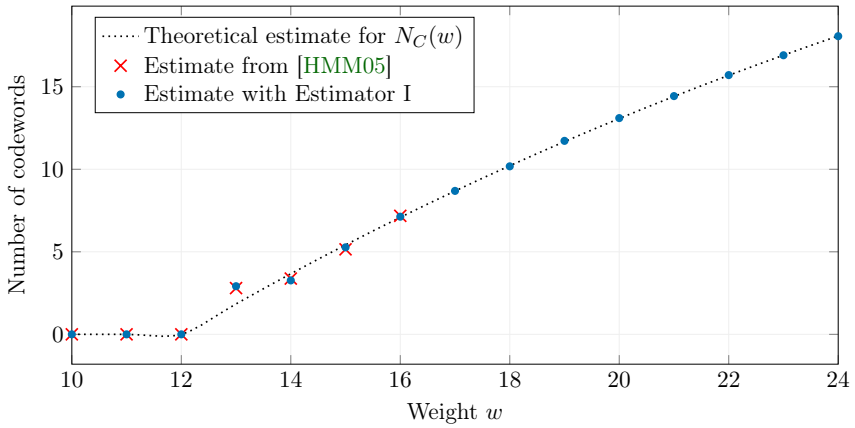

Figure 3.5: Theoretical estimate of the (log of) number of codewords of a [100,50] random linear code. The estimator used has been tested on 20 takes. From $w = 17$ the estimator from [HMM05] saturates.

We observe that for small values of w, the two estimates are almost identical, while for bigger weights, the method from [HMM05] does not give any appreciable result. This is

due to the fact that in these cases the value p^*_{ISD} is too close to one to be observed with only 20 calls to ISD. We could try to fix this issue augmenting the number of calls. This is possible for small instances, but in general, for bigger numbers, the rate of growth of $N_C(w)$ is so high that this approach quickly becomes computationally infeasible. Notice that this is exactly the reason for which the authors of [HMM05] said this approach works only for small values of w.

A comparison between estimators I and II requires some technicalities which we will introduce in the next subsections. Yet, it is easy to see that, at least for some regimes, the two estimators have the same behaviour. Indeed, when $N_C(w)$ is extremely large, the probability that the same codeword is returned by more than one ISD call is negligible. Assuming $m \cdot p_{\text{ISD}}(w) \ll 1$, we have

$$\frac{|A|}{1-\big(1-p_{\text{ISD}(w)}\big)^m} \approx \frac{|A|}{m \cdot p_{\text{ISD}}(w)} \approx \frac{\sum_{i=1}^m \widehat{N}_i}{m \cdot p_{\text{ISD}}(w)},$$

which is exactly the estimate used inside Estimator I.

3.3.3 Statistical Reliability and Overall Time Complexity

In this subsection, we derive the number of ISD calls we would need, in order to achieve some bounded error in our estimate. Starting from this, we are able to derive a closed formula for the time complexity of our approach. We now compare the two new estimators considering the following constraints:

- same computational complexity, i.e., same number of ISD calls (which we denote by m) and same setting for the ISD (same algorithm, same parameters);
- same approximation factor, i.e., same value of ε.

To do this, we study estimators as probabilistic algorithms that solve the approximate version of WEP with some probability $\zeta \geq 0$. Using the Chebychev bound, we derive lower bounds for ζ.

Theorem 3.3.4 (Success probability of estimator I)**.** *Let us consider estimator I with input parameters w and m. For any $\varepsilon \geq 0$, the error in the estimate is lower than ε with probability at least*

$$1 - \frac{1-p_{\mathsf{ISD}}(w)}{m\varepsilon^2 N_C(w) p_{\mathsf{ISD}}(w)}.$$

Proof. We model $\widehat{N}_i$ as the sum of $N_C(w)$ random variables, each one associated to a codeword in C_w. Each variable is equal to 0 if the associated codeword is not returned, and 1 otherwise. Indeed, under Assumption 1, any weight-w codeword behaves as a random codeword over C^*_w. Hence, it will be returned as the output of ISD with probability $p_{\mathsf{ISD}}(w)$. Since $\widehat{N}_i$ is the sum of $N_C(w)$ independent Bernoulli variables, with parameter p_{ISD}, we have

$$\mathbb{E}[\widehat{N}_i] = p_{\mathsf{ISD}}(w) N_C(w), \qquad \text{Var}(\widehat{N}_i) = p_{\mathsf{ISD}}(w)\big(1-p_{\mathsf{ISD}}(w)\big) N_C(w).$$

We will make an error lower than some $\varepsilon \geq 0$ whenever

$$(1-\varepsilon) N_C(w) < N^*_C(w) < (1+\varepsilon) N_C(w).$$

Since $N_C^*(w) = \frac{1}{m \cdot p_{\mathsf{ISD}}(w)} \sum_{i=1}^{m} \widehat{N}_i$, we obtain

$$m(1-\varepsilon)p_{\mathsf{ISD}}(w)N_C(w) < \sum_{i=1}^{m} \widehat{N}_i < m(1+\varepsilon)p_{\mathsf{ISD}}(w)N_C(w). \tag{3.20}$$

Let $X = \sum_{i=1}^{m} \widehat{N}_i$, with expected value $\mu = \mathbb{E}[X] = m \cdot N_C(w) \cdot p_{\mathsf{ISD}}(w)$ and variance

$$\mathrm{Var}(X) = \sum_{i=1}^{m} \mathrm{Var}(\widehat{N}_i) = mN_C(w)p_{\mathsf{ISD}}(w)\big(1 - p_{\mathsf{ISD}}(w)\big).$$

Notice that we can sum the variances of each $\widehat{N}_i$ since each call to ISD is independent and uncorrelated from the others. With this new notation, we rewrite Eq. (3.20) as $|X - \mu| < \mu\varepsilon$. From Chebyshev's inequality, we get

$$\begin{aligned}\mathbb{P}\left[|X-\mu| \geq \mu\varepsilon\right] &\leq \frac{\mathrm{Var}(X)}{\mu^2\varepsilon^2} \\ &= \frac{mN_C(w)p_{\mathsf{ISD}}(w)\big(1-p_{\mathsf{ISD}}(w)\big)}{\big(mN_C(w)p_{\mathsf{ISD}}(w)\big)^2\varepsilon^2} = \frac{\big(1-p_{\mathsf{ISD}}(w)\big)}{m\varepsilon^2 N_C(w)p_{\mathsf{ISD}}(w)}.\end{aligned}$$

We finally obtain

$$\begin{aligned}\mathbb{P}\left[|X-\mu| < \mu\varepsilon\right] &= 1 - \mathbb{P}\left[|X-\mu| \geq \mu\varepsilon\right] \\ &\geq 1 - \frac{1-p_{\mathsf{ISD}}(w)}{m\varepsilon^2 N_C(w)p_{\mathsf{ISD}}(w)}.\end{aligned}$$

□

We now took into consideration the Estimator II. Notice that each codeword will be found, after m ISD calls, with probability $1 - \big(1 - p_{\mathsf{ISD}}(w)\big)^m$. As a consequence, $|A|$ is the sum of $N_C(w)$ independent and equally distributed Bernoulli variables and has expected value and variance given by

$$\begin{aligned}\mathbb{E}[|A|] &= N_C(w)\Big(1 - \big(1 - p_{\mathsf{ISD}}(w)\big)^m\Big), \\ \mathrm{Var}[|A|] &= N_C(w)\Big(1 - \big(1 - p_{\mathsf{ISD}}(w)\big)^m\Big)\big(1 - p_{\mathsf{ISD}}(w)\big)^m.\end{aligned}$$

Given these observations, the following result is immediate and its proof proceeds analogously to that of the previous theorem.

Theorem 3.3.5 (Success probability of estimator II)**.** *Let us consider estimator II with input parameters w and m. For any $\varepsilon \geq 0$, the error in the estimate is lower than ε with probability at least*

$$1 - \frac{\big(1 - p_{\mathsf{ISD}}(w)\big)^m}{\varepsilon^2 N_C(w)\Big(1 - \big(1 - p_{\mathsf{ISD}}(w)\big)^m\Big)}.$$

The above theorems allow us to derive the number of ISD calls we need to achieve a fixed relative error in the estimate.

Corollary 3.3.6 (Number of ISD calls for Approximate WEP)**.** *Let $\varepsilon \geq 0$ and $\zeta \in [0,1]$, and denote by m_I and m_{II} the minimum number of ISD calls that we need, respectively,*

so that estimators I and II achieve a relative error lower than ε with probability at least $1-\zeta$. Then, we have

$$m_I \geq \frac{(1-p_{\mathsf{ISD}}(w))}{\varepsilon^2 \zeta N_C(w) p_{\mathsf{ISD}}(w)},$$

$$m_{II} \geq \frac{\log\left(1+\zeta\varepsilon^2 N_C(w)\right) - \log\left(\zeta\varepsilon^2 N_C(w)\right)}{\log\left(1-p_{\mathsf{ISD}}(w)\right)}.$$

Interestingly, we can also derive the number of ISD calls we need, in order to derive an exact estimate for $N_C(w)$. To this end, consider the following result.

Corollary 3.3.7 (Number of ISD calls for exact counting)**.** *Consider estimators I and II, for weight w and m calls to ISD. Then to have that $\lfloor N_C^*(w) \rceil$ is equal to $N_C(w)$ with probability at least $1-\zeta$ (with $\zeta \geq 0$ and constant), we need*

$$m_I \geq \frac{4N_C(w)(1-p_{\mathsf{ISD}}(w))}{\zeta p_{\mathsf{ISD}}(w)},$$

$$m_{II} \geq \frac{\log\left(1+\zeta/4N_C(w)\right) - \log\left(\zeta/4N_C(w)\right)}{\log\left(1-p_{\mathsf{ISD}}(w)\right)}.$$

Proof. We observe that $\lfloor N_C^*(w) \rceil = N_C(w)$ if

$$N_C(w) - \frac{1}{2} \leq N_C^*(w) \leq N_C(w) + \frac{1}{2}.$$

Since we are expressing the maximum error term, in absolute terms, as $\pm\varepsilon N_C(w)$, we set

$$\varepsilon = \frac{1}{2 \cdot N_c(w)}.$$

Substituting this into the value resulting from Cor. 3.3.6, we obtain the thesis. □

Putting everything together, and considering the running time of ISD algorithms, we can derive the resulting time complexity for our estimator, as a function of only (i) the desired error term, (ii) the confidence in the estimate, and (iii) the performances of the considered ISD algorithm.

Proposition 3.3.8 (Overall cost)**.** *Estimator I and II, using a ISD subroutine with cost $t_{\mathsf{ISD}}(w)$, on input w and desiring a confidence interval $\varepsilon \geq 0$ with probability at least $1-\zeta$, have average time complexity respectively*

$$m_I \cdot t_{\mathsf{ISD}}(w) = \frac{(1-p_{\mathsf{ISD}}(w))}{\varepsilon^2 \zeta N_C(w) p_{\mathsf{ISD}}(w)} \cdot t_{\mathsf{ISD}}(w),$$

$$m_{II} \cdot t_{\mathsf{ISD}}(w) = \frac{\log\left(1+\zeta\varepsilon^2 N_C(w)\right) - \log\left(\zeta\varepsilon^2 N_C(w)\right)}{\log\left(1-p_{\mathsf{ISD}}(w)\right)} \cdot t_{\mathsf{ISD}}(w).$$

The above formulas can be instantiated with the desired ISD variant. In the following, we describe how Estimator I complexity behaves if Lee&Brickell and Stern ISD are used. Computing the associated complexity for Estimator II is straightforward.

Corollary 3.3.9. *Estimator I, using Lee & Brickell subroutine, on input w and desiring a confidence interval $\varepsilon \geq 0$ with probability at least $1-\zeta$, has average time complexity*

$$\frac{\left(1-\binom{n-k}{w-p}\binom{k}{p}\binom{n}{w}^{-1}\right)}{\varepsilon^2 \zeta N_C(w)} \cdot \frac{\left(n(n-k)^2/p_{\mathsf{inv}} + \binom{k}{p}(q-1)^p\right)}{\binom{n-k}{w-p}\binom{k}{p}\binom{n}{w}^{-1}}.$$

Corollary 3.3.10. *Estimator I, using Stern subroutine, on input w and desiring a confidence interval $\varepsilon \geq 0$ with probability at least $1-\zeta$, has average time complexity*

$$\frac{(1-\binom{\frac{k+\ell}{2}}{p}^2\binom{n-k-\ell}{w-2p}\binom{n}{w}^{-1}}{\varepsilon^2\zeta N_C(w)} \cdot \frac{\left(n(n-k)^2/p_{\mathsf{inv}} + \frac{L^2}{q^\ell} + L\right)}{\binom{\frac{k+\ell}{2}}{p}^2\binom{n-k-\ell}{w-2p}\binom{n}{w}^{-1}}.$$

3.3.4 Numerical Results

Table 3.9 shows the numerical results for a known sparse code obtained by the proposed methods. The code a $(504, 252)$ LDPC code taken from [MC09] and it has been chosen since it is one of the code tested in [HMM05]. To achieve these results we followed the approaches discussed in the previous subsections, using Stern's variant with parameters $p = 2$ and $\ell = 14$. We employed an Intel Xeon Gold (2.30 GHz) processor equipped with 512 Gb RAM, stopping the algorithm after 300 iterations. It took about 143 hours to complete all computations. Results are reported in Table 3.9.

w	Approach in [HMM05]	Estimator I	Estimator II
20	0	0	0
22	38	38	40
24	171	170	170
26	372	371	373
28	1788	1788	1789
30	24460	24002	24008
32	66681	65788	65792

Table 3.9: Estimated weight distribution of the $(504, 252)$ LDPC code from [MC09].

Similar estimates can be made for the $(495, 433)$ LDPC code from [MC09]. In this case, for each tested value of w, 1500 attempts were made. The ISD algorithm used for all three estimators is Stern with parameters $p = 2$ and $\ell = 20$. The computation takes approximately 60 hours on our hardware. Results as reported in Table 3.10.

w	Approach in [HMM05]	Estimator I	Estimator II
4	–	59	60
6	–	4436	4433
8	–	598134	596643
10	–	97205574	95509493
12	–	16914867181	17332975706
14	–	3052490293505	3126061305891
16	–	593031553106376	645685022171317

Table 3.10: Estimated weight distribution of the $(495, 433)$ LDPC code from [MC09].

Notice that for the values of w which have been tested, the output results of the estimator [HMM05] do not produce anything useful. This is not a surprise, according to what has been shown in the previous subsections. In fact, even for the smallest case, $w = 4$, we can use the rough estimate given by estimators I and II and say that

$$p^*_{\mathsf{ISD}}(4) \approx 1 - (1 - p_{\mathsf{ISD}}(4))^{60} = 0,9999998677.$$

This says that running ISD on that particular code, with that specific weight, would be unsuccessful less than once over a million attempts. As a consequence, to obtain a reasonable estimate even with the method described in [HMM05], we would have to repeat the experiment many more times than we did, and this was not an option, given the computational power at our disposal.

Notice that [HMM05] provides actual estimates for the weight distribution of the code described in Table 3.10. This difference can be explained both by the different hardware and, more importantly, the difference in the number of attempts, which in Hirotomo, Mohri and Morii's case can be easily obtained by reversing Equation 3.18, obtaining, for $w = 4$, $m \approx 99885664$. In this case the estimated number of codewords for $w \in \{4, 6, 8, 10\}$ is respectively given by $60, 4436, 564458$ and 16992863. No results are provided for larger values of w.

A SageMath proof-of-concept implementation of these results can be found at `https://github.com/triki96/WeightDistribution`.

Chapter 4

Theoretical Foundations and PoKs

"Since the appearance of public-key cryptography in the seminal Diffie-Hellman paper, many new schemes have been proposed and many have been broken. Thus, the simple fact that a cryptographic algorithm withstands cryptanalytic attacks for several years is often considered as a kind of validation procedure. A much more convincing line of research has tried to provide "provable" security for cryptographic protocols."

D. Pointcheval and J. Stern, 1997

Interactive proofs are a cornerstone of modern cryptography and, as such, used in many areas, from digital signatures to multy-party computation. A standard method of making them non-interactive and produce provably secure proofs is to apply the Fiat-Shamir transform. In this context, often the knowledge error κ of the underlying interactive proof is not small enough, and thus needs to be reduced. This is usually achieved by repeating the interactive proof in parallel t times. Recently, it was shown that the t-fold parallel repetition of any $(k_1, \ldots, k_\mu)$-special-sound multi-round public-coin interactive proof reduces the knowledge error from κ to κ^t, which is optimal. Parallel repetition of knowledge-sound interactive proofs improves security at the price of bigger transcripts. A few generic techniques have been proposed to mitigate this issue, such as the seed tree, multiple public key, and fixed-weight optimization. In this chapter, we focus on the latter and analyze it under a security standpoint. In particular, in Section 4.2 we will analyze the formal security of digital signatures obtained by modifying the Fiat-Shamir transform by applying the fixed-weight optimization, while in Section 4.3 we focus on online-extractable transforms, building a new one which is fixed-weight by design.

4.1 The Fixed-Weight Optimization

There are a variety of techniques aimed at improving the performance and efficiency of the underlying interactive proof. Informally, these techniques can be explained as transforming one interactive protocol into another. Some of them (like the *Multiple Public Key* optimization) aim at limiting the number t of repetitions, obtaining an improvement in both execution time and transcript size at the cost, for example, of bigger public keys. Other techniques , on the other hand, aim at reducing the transcript size at the cost of a slight increase in execution time. This is particularly desired when storage or transmission latency are the main concern. Among the latter techniques, one of the most common optimizations is using *fixed-weight challenge vectors*. The challenges are the random coins

(a single value in the case of a 3-round interactive proof, a list of μ values in the case of a $(2\mu+1)$-round interactive proof) sent by the verifier to the prover. Here, by fixed-weight challenge vectors, we mean vectors of challenges having a constant number of entries (for which the last component is) equal to a fixed value. This optimization has proven to be particularly helpful when different challenges have drastically-different response sizes [Bal+23b; GPV24; Bar+21a; Cho+23b; RST23; Beu+23; BKP20].

Notation. In the following, for a finite set X, we write $|X|$ for the cardinality of X and by $|x|$ we denote the number of bits necessary to represent an element $x \in X$. When s is a list or a vector, we write $(s)_i$ to denote the i-th element of s. If S is a set whose elements are lists or vectors, we define $(S)_i := \{x : \exists s \in S : (s)_i = x\}$. Given μ finite sets $\mathsf{Ch}^{[1]}, \ldots, \mathsf{Ch}^{[\mu]}$ and $\mathbf{c} \in \mathsf{Ch}^{[1]} \times \ldots \times \mathsf{Ch}^{[\mu]}$, we will write $\mathbf{c} = (c^{[1]}, \ldots, c^{[\mu]})$ where $c^{[i]} \in \mathsf{Ch}^{[i]}$ for all $i \in \{1, \ldots, \mu\}$. Furthermore, given $t \in \mathbb{N}^*$ and $\mathbf{c} \in (\mathsf{Ch}^{[1]} \times \ldots \times \mathsf{Ch}^{[\mu]})^t$, we will write $\mathbf{c} = ((\mathbf{c})_1, \ldots, (\mathbf{c})_t)$ where $(\mathbf{c})_j \in \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$ and $(\mathbf{c})_j^{[i]} \in \mathsf{Ch}^{[i]}$ for all $j \in \{1, \ldots, t\}, i \in \{1, \ldots, \mu\}$.

Definition 4.1.1 (Weight)**.** Let Ch be a finite set, $t \in \mathbb{N}^*$ and $\tilde{c} \in \mathsf{Ch}$. For an element $c = ((c)_1, \ldots, (c)_t) \in \mathsf{Ch}^t$, we define the *weight of* c with respect to $\tilde{c}$ as

$$\mathsf{wt}_{\tilde{c}}(c) := |\{j \in \{1, ..., t\} : (c)_j = \tilde{c}\}|.$$

Definition 4.1.2. Let $t, \mathrm{w}, \mu \in \mathbb{N}^*$ such that $t \geq \mathrm{w}$, let $\mathsf{Ch}^{[1]}, \ldots, \mathsf{Ch}^{[\mu]}$ be finite sets and let $\tilde{c} \in \mathsf{Ch}^{[\mu]}$. Given $\mathsf{Ch} = \mathsf{Ch}^{[1]} \times \ldots \times \mathsf{Ch}^{[\mu]}$, we denote by $\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ the set of elements $\mathbf{c} \in \mathsf{Ch}^t$ for which $\mathsf{wt}_{\tilde{c}}\left((\mathbf{c})_1^{[\mu]}, \ldots, (\mathbf{c})_t^{[\mu]}\right) = \mathrm{w}$, i.e.

$$\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}} := \left\{\mathbf{c} \in \mathsf{Ch}^t : \mathsf{wt}_{\tilde{c}}\left(\left((\mathbf{c})_1^{[\mu]}, \ldots, (\mathbf{c})_t^{[\mu]}\right)\right) = \mathrm{w}\right\}.$$

When $\tilde{c}$ is clear from the context, we will simplify the notation and write $\mathsf{Ch}^{t,\mathrm{w}}$ instead of $\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$. Furthermore, when Ch is not a Cartesian product but a simple set (i.e., $\mu = 1$ and so $\mathsf{Ch} = \mathsf{Ch}^{[\mu]}$), we will simply denote by $\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ the set

$$\left\{c = ((c)_1, \ldots, (c)_t) \in \mathsf{Ch}^t : \mathsf{wt}_{\tilde{c}}(c) = \mathrm{w}\right\}.$$

Definition 4.1.3 (Fixed-weight Repetition)**.** Let $k_1, \ldots, k_\mu, N_1, \ldots, N_\mu \in \mathbb{N}^*$, $R \subseteq X \times Y$ be a binary relation and (P, V) be a $(2\mu+1)-$round public-coin interactive proof for R, where V samples $i-$th challenges ($i \in \{1, \ldots, \mu\}$) from a set $\mathsf{Ch}^{[i]}$ of cardinality $N_i \geq k_i$. Therefore, the challenge set of (P, V) is $\mathsf{Ch} = \prod_{i=1}^{\mu} \mathsf{Ch}^{[i]}$. Let $\tilde{c}$ be a given element of $\mathsf{Ch}^{[\mu]}$. A (t, w)-*fixed-weight parallel repetition* of (P, V) with respect to $\tilde{c}$, which we denote by $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$, is a t-fold parallel repetition of (P, V) whose challenge set is $\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$.

Throughout this work, we will consider fixed-weight repetitions only for $(2\mu+1)-$round public-coin interactive proofs for which there exists a unique element $\tilde{c} \in \mathsf{Ch}^{[\mu]}$ such that, for every possible $\mathbf{c} = (c^{[1]}, \ldots, c^{[\mu]}) \in \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$, the response size when $c^{[\mu]} = \tilde{c}$ is significantly higher than when $c^{[\mu]} \neq \tilde{c}$. Under this assumption, a fixed-weight repetition can lead to a more compact protocol compared to a plain parallel repetition, as it was shown in [GPV24; Bar+21a; Bal+23b; Cho+23b; RST23].

Remark 4.1.4. In Definition 4.1.3, we choose to consider the fixed element $\tilde{c}$ as an element of $\mathsf{Ch}^{[\mu]}$ rather than of the challenge set of previous rounds or a Cartesian product of (a subset of) them. This is consistent with the concrete instances of the fixed-weight technique that have appeared so far (see the list above).

4.2 Security of Fixed-Weight Repetitions of Special-Sound Multi-Round Proofs

The security of the fixed-weight optimization is well understood in the case of a 3-round, public-coin, 2-special-sound interactive proof, i.e. $\mu = 1$ and $k_1 = 2$. In fact, in this case the 2-special soundness of the base interactive proof is preserved by the fixed-weight repetition of the interactive proof. However, the picture becomes fuzzy when $\mu = 1$ and $k_1 > 2$, and even more when $\mu > 1$. In particular, already in the case of $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round public-coin interactive proofs with $k_1 > 2$ it is not clear whether the fixed-weight t-fold parallel repetition satisfies any useful notion of special soundness. In light of this and the implications it would have on the provable security of existing signatures like CROSS [Bal+23b] or [GPV24] and future protocols, the following research question naturally arises:

Does a fixed-weight repetition of a $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round public-coin interactive proof enjoy knowledge soundness?

Building on the results from [AF22a] we positively answer the question above by explicitly building a knowledge extractor and precisely bounding the knowledge error. More precisely, we prove that the t-fold repetition with fixed weight w of a $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round interactive proof is knowledge sound. We also provide an explicit expression for the knowledge error of the parallel repetition of the interactive proof. Such knowledge error coincides with the maximum cheating probability of a dishonest prover, which is directly derived from the maximum size of the set of challenges to which the prover can answer without actually knowing a witness. This shows that our result is optimal and, in addition, it allows to formally prove the security of the interactive proofs underlying some recent post-quantum signatures, such as the code-based signature CROSS [Bal+23b] and the SIDH-based signature of [GPV24].

Technical Overview. One of the main results of [AF22a] is a knowledge extractor Ext for k-special-sound interactive proofs, whose success probability when applied to a dishonest prover P^* can be expressed in terms of a novel characterization of the power of P^*. Specifically, the ability of P^* to correctly answer to a random challenge is measured by $\delta_k(\mathsf{P}^*)$, its worst-case success probability when $k - 1$ challenges are removed from the challenge space. This new framework is particularly convenient when moving to the parallel repetition of the interactive proof. In fact, starting from a dishonest prover P^* against the t-parallel repetition of a k-special-sound interactive proof, it is possible to build t provers $\mathsf{P}_1^*, \ldots, \mathsf{P}_t^*$ against the single instance of the interactive proof. By applying the previous extractor in parallel to the provers of the single instance, the probability of witness extraction can be expressed via $\delta_k(\mathsf{P}_1^*) + \ldots + \delta_k(\mathsf{P}_t^*)$. From this, an optimal bound on the knowledge error of the parallel repetition is obtained.

The extraction algorithm Ext of [AF22a] queries the dishonest prover on uniformly-sampled challenges. Instead, when we consider fixed-weight repetitions, challenges must be sampled according to a different distribution. At the core of our result is therefore a generalisation of the knowledge extractor from [AF22a] which allows for the sampling of challenges according to an arbitrary distribution $\mathscr{D}$ over the challenge space. More in detail, we show that the extraction probability is given by $\delta_k(\mathsf{P}^*, \mathscr{D})/k$ for a k-special-sound interactive proof, where the probability space is defined by the challenges being sampled according to $\mathscr{D}$. For fixed-weight repetitions, we can then apply a similar approach as

before: starting from a dishonest prover for the fixed-weight t-fold repetition of the interactive proof, we build t dishonest provers on the single instance of the interactive proof. By applying the generalised extractor in parallel, we obtain a bound on the knowledge error of the parallel repetition of the interactive proof (Subsection 4.2.2). A similar strategy is then applied to prove knowledge soundness of fixed-weight repetitions of generic $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round interactive proofs. In particular, we first generalise the multi-round extractor from [AF22a] over arbitrary distributions and then apply it to fixed-weight repetitions (Subsection 4.2.3).

Although the resulting bounds cannot be expressed directly in terms of the knowledge error of a single instance, the obtained knowledge errors coincide with the maximum cheating probability of a dishonest prover, meaning that our result is optimal. As already observed, the maximum cheating probability of a dishonest prover is derived from the maximum size of the set of challenges to which the prover can answer without actually knowing a witness. We have translated the problem of computing these sizes into finding upper bounds for the cardinality of particular subsets of the Cartesian product of finite sets which satisfy some conditions on the components. In particular, we compute the maximum size of a set of sequences when we limit the number of different values that may appear in any single component (Subsection 4.2.1). These combinatorial results, from which a bound on the knowledge error of fixed-weight repetitions is deduced, may be of independent interest and find application in independent scenarios, and so we have formulated and proved them in full generality.

Organization. In Subsection 4.2.1 we discuss some combinatorial results that, while interesting on their own, will be essential for Subsections 4.2.2 and 4.2.3, which contain the core cryptographic results of our work. In particular, Subsection 4.2.2 deals with the easier case of 3-round interactive proofs, while Subsection 4.2.3 deals with the general multi-round case. Lastly, in Subsection 4.2.4 we identify some applications and future research directions.

4.2.1 Combinatorial Bounds

To prove that a fixed-weight repetition of a special-sound interactive proof still enjoys knowledge soundness, we will rely on some combinatorial bounds. These bounds appear to be of independent interest, and for this reason we will state and prove them in full generality in this subsection. Nevertheless, as they will find a natural application in Subsections 4.2.2 and 4.2.3, we will try to use the same notation that will be used there.

High-Level Overview

Here we provide a high-level overview of the subsection. By reading this introduction, a reader not interested in engaging with every formal step necessary to rigorously prove the combinatorial results can have a summary of the most significant outcomes.

The main problem addressed in this subsection arises from the need to estimate, for the fixed-weight repetition of a given special-sound interactive proof, the maximum success probability of a dishonest prover. This is derived from the maximum cardinality of the sets of transcripts/challenges from which it is not possible to extract a witness.

In Proposition 4.2.1 of Subsection 4.2.1, we quantify the maximum cardinality of the sets $S \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ such that $|(S)_i| \leq k$ for all $i \in \{1, \ldots, t\}$, where Ch is the finite challenge space of a k-special-sound 3-round interactive proof, $\tilde{c} \in \mathsf{Ch}$ and $w \leq t$ are two positive

integers. In particular, it holds that:

$$|S| \leq \binom{\mathrm{w}(k-1)}{\mathrm{w}} (k-2)^{\mathrm{w}(k-2)} (k-1)^{t-\mathrm{w}(k-1)}$$

if $t \geq \mathrm{w}(k-1)$, and:

$$|S| \leq \binom{t}{\mathrm{w}} (k-2)^{t-\mathrm{w}}$$

otherwise.

Subsection 4.2.1 aims at giving a bound for multi-round interactive proofs, where the situation is more involved. For this reason, we give a formal definition of the sets of transcripts/challenges from which it is not possible to extract a witness. For convenience, we refer to such sets as *acceptable sets* (Definition 4.2.4). To simplify the computation, we resort to Lemma 4.2.8, which skims these sets, and reduce the considered problem to finding the maximum over a specific classes of acceptable sets, which we call *saturated acceptable sets*. Following Definition 4.2.9, we can characterize each saturated acceptable set via a sequence $(d_{i,b_i})_{i\in\{1,\dots,t\}}$ that basically represents the cardinality of some specific subsets. More precisely, Lemma 4.2.10 shows that the cardinality of a saturated acceptable set can be expressed by a formula which depends on its sequence $(d_{i,b_i})_{i\in\{1,\dots,t\}}$, and that two other important properties are satisfied by $(d_{i,b_i})_{i\in\{1,\dots,t\}}$ (Equation (4.2) and Equation (4.3)). Then, by relying on a technical lemma on sequences of non-negative integers (Lemma 4.2.11), we prove in Proposition 4.2.12 an upper bound for the cardinalities of acceptable sets.

Bound for the 3-round case

Let us consider a finite set Ch with a fixed element $\tilde{c} \in \mathsf{Ch}$, and three positive integers k, t, w such that $|\mathsf{Ch}| \geq k \geq 2$ and $t \geq \mathrm{w}$. Our first result is an upper bound of the cardinality of particular sets $S \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ such that $|(S)_i| \leq k$ for all $i \in \{1,\dots,t\}$. This mathematical question arises in the context of a k-special-sound 3-round interactive proof for which we consider a (t,w)-fixed-weight parallel repetition. The maximum cheating probability of a dishonest prover in this case can be estimated by means of the above bound (dividing it by the total cardinality of $\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$).

Proposition 4.2.1. *Given $S \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ such that $|(S)_i| < k$ for all $i \in \{1,\dots,t\}$, we have that, if $t \geq \mathrm{w}(k-1)$:*

$$|S| \leq \binom{\mathrm{w}(k-1)}{\mathrm{w}} (k-2)^{\mathrm{w}(k-2)} (k-1)^{t-\mathrm{w}(k-1)},$$

and otherwise:

$$|S| \leq \binom{t}{\mathrm{w}} (k-2)^{t-\mathrm{w}}.$$

Proof. Since $k = 2$ trivially implies $|S| \leq 1 = \binom{\mathrm{w}}{\mathrm{w}} 0^0 1^{t-\mathrm{w}}$, in the remainder we suppose $k \geq 3$[1].

For $s \in S$, let us define $Z(s)$ as the set of indices $i \in \{1,\dots,t\}$ such that the i-th entry of s is equal to $\tilde{c}$. The set $Z(S) := \bigcup_{s\in S} Z(s)$ is therefore formed by all indices i for which at least one element of S has $\tilde{c}$ as i-th entry. We denote $|Z(S)|$ by h_S, for which

[1]Here we used $0^0 = 1$, which is a common convention in discrete mathematics, as it comes handy in many situations (e.g. the binomial theorem). For more details on this convention, see [Knu92].

holds $h_S \geq \mathrm{w}$ by definition of $\mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$. As our goal is providing an upper bound for the cardinality of S, we can assume that, for each $i \in \{1,\dots,t\}$, it holds that $|(S)_i| = k-1$, with $\tilde{c} \in (S)_i$ if and only if $i \in Z(S)$. Every element s of S can be thought as constructed by the following strategy. Choose a subset R of cardinality w from $Z(S)$ and, for every $i \in R$, set the i-th entry of s to $\tilde{c}$; for every $i \in Z(S) \setminus R$ choose a value in $(S)_i \setminus \{\tilde{c}\}$; finally, for every $i \in \{1,\dots,t\} \setminus Z(S)$, choose a value in $(S)_i$. This means that $|S|$ is bounded above by:

$$f(h_S) := \binom{h_S}{\mathrm{w}}(k-2)^{h_S-\mathrm{w}}(k-1)^{t-h_S}. \tag{4.1}$$

We note that $\binom{h_S}{\mathrm{w}}(k-2)^{h_S-\mathrm{w}}$ is monotonically increasing with ratio:

$$\frac{\binom{h_S+1}{\mathrm{w}}(k-2)^{h_S+1-\mathrm{w}}}{\binom{h_S}{\mathrm{w}}(k-2)^{h_S-\mathrm{w}}} = \frac{h_S+1}{h_S+1-\mathrm{w}}(k-2),$$

while $(k-1)^{t-h_S}$ is monotonically decreasing with ratio:

$$\frac{(k-1)^{t-h_S}}{(k-1)^{t-(h_S+1)}} = k-1.$$

This means that f is increasing as long as

$$\frac{h_S+1}{h_S+1-\mathrm{w}}(k-2) > k-1 \quad \Longleftrightarrow \quad h_S < \mathrm{w}(k-1)-1.$$

In conclusion, since $f(\mathrm{w}(k-1)-1) = f(\mathrm{w}(k-1))$, if $t \geq \mathrm{w}(k-1)$ we have

$$|S| \leq f\big(\mathrm{w}(k-1)\big) = \binom{\mathrm{w}(k-1)}{\mathrm{w}}(k-2)^{\mathrm{w}(k-2)}(k-1)^{t-\mathrm{w}(k-1)}.$$

On the other hand, if $t < \mathrm{w}(k-1)$, then f is increasing up to t, so:

$$|S| \leq f(t) = \binom{t}{\mathrm{w}}(k-2)^{t-\mathrm{w}}. \qquad \square$$

We now want to generalise the above result to a setting where the set Ch is replaced by the Cartesian product of μ finite sets $\mathsf{Ch}^{[1]},\dots,\mathsf{Ch}^{[\mu]}$, i.e. $\mathsf{Ch} := \prod_{\ell=1}^{\mu}\mathsf{Ch}^{[\ell]}$, and we fix an element $\tilde{c}$ in $\mathsf{Ch}^{[\mu]}$. We formalise such generalisation in the following definitions, by introducing the concept of acceptable sets.

Bound for the $(2\mu+1)$-round case

In the remainder of this subsection $k_1,\dots,k_\mu,N_1,\dots,N_\mu$ will denote positive integers while $\mathsf{Ch}^{[1]},\dots,\mathsf{Ch}^{[\mu]}$ finite sets such that $|\mathsf{Ch}^{[\ell]}| = N_\ell \geq k_\ell \geq 2\ \forall \ell \in \{1,\dots,\mu\}$. We set $\mathsf{Ch} = \prod_{\ell=1}^{\mu}\mathsf{Ch}^{[\ell]}$.

Definition 4.2.2. For any $S \subseteq \mathsf{Ch}$, $\ell \in \{2,\dots,\mu\}$ and $(s_1,\dots,s_{\ell-1}) \in \prod_{j=1}^{\ell-1}\mathsf{Ch}^{[j]}$, we define:

$$S_\ell(s_1,\dots,s_{\ell-1}) := \{s \in \mathsf{Ch}^{[\ell]} : \exists\,(s_1,\dots,s_{\ell-1},s,r_{\ell+1},\dots,r_\mu) \in S\}.$$

Informally, $S_\ell(s_1,\dots,s_{\ell-1})$ denotes the set of ℓ-th entries of the elements of S with the first $\ell-1$ entries equal to $(s_1,\dots,s_{\ell-1})$.

Definition 4.2.3 (Acceptability Predicate)**.** Given $S \subseteq \mathsf{Ch}$, for any $(a_1, \ldots, a_\mu) \in \mathsf{Ch}$ we define the predicate $P_{S,\mu}$ as follows:

$$P_{S,\mu}((a_1, \ldots, a_\mu)) \iff (a_1, \ldots, a_\mu) \in S.$$

For $\ell \in \{1, \ldots, \mu - 1\}$, the predicate $P_{S,\ell}((a_1, \ldots, a_\ell))$ is true if and only if:

$$|\{(a_1, \ldots, a_\ell, a_{\ell+1}) \colon a_{\ell+1} \in S_{\ell+1}(a_1, \ldots, a_\ell) \wedge P_{S,\ell+1}(a_1, \ldots, a_{\ell+1})\}| \geq k_{\ell+1}.$$

Analogously to the previous subsection, we aim at finding an upper bound of the cardinality of particular sets $S \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ that do not define a $(k_1, \ldots, k_\mu)$-tree of accepting transcripts in the context of Definition 1.1.15. We call these sets *acceptable sets*. The mentioned bound determines the maximum cheating probability of a dishonest prover in the case of a $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round interactive proof for which we consider a (t, w)-fixed-weight parallel repetition.

Definition 4.2.4 (Acceptable Set)**.** Given $A \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$, we say that it is a $(k_1, \ldots, k_\mu)$-*out-of*-$(N_1, \ldots, N_\mu)$ *acceptable set* (or, when clear from the context, simply *acceptable set*) if the following condition holds:

$$|\{a_1 \colon a_1 \in (A)_i^{[1]} \wedge P_{(A)_i,1}(a_1)\}| < k_1 \qquad \forall i \in \{1, \ldots, t\}.$$

In this work a subfamily of acceptable sets are of particular importance. We name them *saturated acceptable sets*. Before formally defining them, we provide some introductive definitions.

Definition 4.2.5 (Saturating Choice)**.** Given μ finite sets $\mathsf{Ch}^{[1]}, \ldots, \mathsf{Ch}^{[\mu]}$, and $\forall \ell \in \{1, \ldots, \mu\}$, $k_\ell \leq |\mathsf{Ch}^{[\ell]}|$, a *saturating choice* is a pair $(\mathcal{S}, \bar{B})$ such that:

- $\mathcal{S} \subseteq \bigcup_{\ell=0}^{\mu-1} \prod_{j=1}^{\ell} \mathsf{Ch}^{[j]}$ is a set of sequences;
- $\bar{B} : \mathcal{S} \to \bigcup_{\ell=1}^{\mu} \mathcal{P}(\mathsf{Ch}^{[\ell]})$ is a function such that for every $s \in \mathcal{S}$ with $|s| = \ell$, associates to s a set $\bar{B}(s) \subseteq \mathsf{Ch}^{[\ell+1]}$, where $\mathcal{P}(\mathsf{Ch}^{[\ell]})$ denotes the power set of $\mathsf{Ch}^{[\ell]}$.

Furthermore, the pair $(\mathcal{S}, \bar{B})$ satisfies the following properties:

- $\varnothing \in \mathcal{S}$, where $\varnothing$ denotes the empty sequence;
- $\bar{B}(\varnothing) \subseteq \mathsf{Ch}^{[1]}$ is such that $|\bar{B}(\varnothing)| = k_1 - 1$;
- $\forall a_1 \in \mathsf{Ch}^{[1]} \setminus \bar{B}(\varnothing)$, $(a_1) \in \mathcal{S}$;
- $\forall (a_1) \in \mathcal{S}$, $\bar{B}((a_1)) \subseteq \mathsf{Ch}^{[2]}$ is a set such that $|\bar{B}((a_1))| = k_2 - 1$;

and, inductively for $\ell \in \{2, \ldots, \mu - 1\}$:

- $\forall (a_1, \ldots, a_\ell) \in \prod_{j=1}^{\ell} \left(\mathsf{Ch}^{[j]} \setminus \bar{B}((a_1, \ldots, a_{j-1}))\right)$, $(a_1, \ldots, a_\ell) \in \mathcal{S}$;
- $\forall (a_1, \ldots, a_\ell) \in \mathcal{S}$, $\bar{B}((a_1, \ldots, a_\ell)) \subseteq \mathsf{Ch}^{[\ell+1]}$ satisfies $|\bar{B}((a_1, \ldots, a_\ell))| = k_{\ell+1} - 1$.

Definition 4.2.6. Given μ finite sets $\mathsf{Ch}^{[1]}, \ldots, \mathsf{Ch}^{[\mu]}$, $\ell < \mu$, $k_{\ell+1} \leq |\mathsf{Ch}^{[\ell+1]}|$, a sequence $(a_1, \ldots, a_\ell) \in \prod_{j=1}^{\ell} \mathsf{Ch}^{[j]}$, and a set $\bar{B} \subseteq \mathsf{Ch}^{[\ell+1]}$ such that $|\bar{B}| = k_{\ell+1} - 1$, we can define the set:

$$\mathrm{sat}_i((a_1, \ldots, a_\ell), \bar{B}) := \Big\{(a_1, \ldots, a_\ell, a_{\ell+1}, a_{\ell+2}, \ldots, a_\mu) \colon a_{\ell+1} \in \bar{B} \wedge (a_{\ell+2}, \ldots, a_\mu) \in \prod_{j=\ell+2}^{\mu} \mathsf{Ch}^{[j]}\Big\}.$$

Definition 4.2.7 (Saturated Acceptable Set)**.** Given $\bar{A} \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$, we say that it is a $(k_1,\ldots,k_\mu)$-*out-of*-$(N_1,\ldots,N_\mu)$ *saturated acceptable set* (or, when clear from the context, simply *saturated acceptable set*) if $\forall i \in \{1,\ldots,t\}$, there is a saturating choice $(\mathcal{S}_i, \bar{B}_i)$ such that $(\bar{A})_i = \bigsqcup_{s\in\mathcal{S}_i} \mathrm{sat}_i(s, \bar{B}_i(s))$ (see Definitions 4.2.5 and 4.2.6). Moreover we have that $\forall((y)_1,\ldots,(y)_t) \in \prod_{i=1}^{t}(\bar{A})_i \bigcap \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$, $((y)_1,\ldots,(y)_t) \in \bar{A}$.

Note that sets that satisfy Definition 4.2.7 are indeed acceptable sets because, by construction, we have that:

$$|\{a_1 : a_1 \in (A)_i^{[1]} \wedge P_{(A)_i,1}(a_1)\}| = k_1 - 1, \qquad \forall i \in \{1,\ldots,t\}.$$

In fact, the saturating choices $(\mathcal{S}_i, \bar{B}_i)$ have the property that, $\forall \ell \in \{1,\ldots,\mu\}$:

$$|\{s \in \mathcal{S}_i : (s, a_{\ell+1},\ldots,a_\mu) \in (A)_i \wedge P_{(A)_i,\ell}(s)\}| = k_\ell - 1.$$

This gives the intuition that there is no margin left to add more elements to saturated acceptable sets while retaining their acceptability. More formally, the following lemma proves that every maximal acceptable set is a saturated acceptable set.

Lemma 4.2.8 (Maximality of Saturated Acceptable Sets)**.** *For every $(k_1,\ldots,k_\mu)$-out-of-$(N_1,\ldots,N_\mu)$ acceptable set $A \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ there exists a saturated acceptable set $\bar{A} \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ such that $\bar{A} \supseteq A$.*

Proof. Let $A \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ be a $(k_1,\ldots,k_\mu)$-out-of $(N_1,\ldots,N_\mu)$ acceptable set. To construct a saturated acceptable set $\bar{A} \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ which contains A, we will build an appropriate saturating choice (see Definition 4.2.5) $(\mathcal{S}_i, \bar{B}_i)$ for all $i \in t$. We initially set $\mathcal{S}_i := \{\varnothing\}$ and then iteratively add elements to it. Let us define $B_i(\varnothing) := \{a_1 : a_1 \in (A)_i^{[1]} \wedge P_{(A)_i,1}(a_1)\}$. By definition of acceptable set, we have that $|B_i(\varnothing)| \leq k_1 - 1$, so we can define $\bar{B}_i(\varnothing) \subseteq \mathsf{Ch}^{[1]}$ to be a set such that $B_i(\varnothing) \subseteq \bar{B}_i(\varnothing)$ and $|\bar{B}_i(\varnothing)| = k_1 - 1$.

For each $a_1 \in \mathsf{Ch}^{[1]} \setminus \bar{B}_i(\varnothing)$, we add (a_1) to $\mathcal{S}_i$, and we define the set

$$B_i((a_1)) := \left\{a_2 : a_2 \in (A)_i^{[2]} \wedge P_{(A)_i,2}((a_1,a_2))\right\}.$$

Again, by definition of acceptable set, we have that $|B_i((a_1))| \leq k_2 - 1$, so we can define $\bar{B}_i((a_1)) \subseteq \mathsf{Ch}^{[2]}$ to be a set such that $B_i((a_1)) \subseteq \bar{B}_i((a_1))$ and $|\bar{B}_i((a_1))| = k_2 - 1$.

In general, for $\ell \in \{2,\ldots,\mu-1\}$ and for any $a_\ell \in \mathsf{Ch}^{[\ell]} \setminus \bar{B}_i((a_1,\ldots,a_{\ell-1}))$, we add $(a_1,\ldots,a_\ell)$ to $\mathcal{S}_i$ and define:

$$B_i((a_1,\ldots,a_\ell)) := \left\{a_{\ell+1} : a_{\ell+1} \in (A)_i^{[\ell+1]} \wedge P_{(A)_i,\ell+1}((a_1,\ldots,a_\ell,a_{\ell+1})\right\}.$$

As before, we have that $|B_i((a_1,\ldots,a_\ell))| \leq k_{\ell+1}-1$, so it is possible to define $\bar{B}_i((a_1,\ldots,a_\ell)) \subseteq \mathsf{Ch}^{[\ell+1]}$ as a set such that:

$$B_i((a_1,\ldots,a_\ell)) \subseteq \bar{B}_i((a_1,\ldots,a_\ell)) \wedge |\bar{B}_i((a_1,\ldots,a_\ell))| = k_{\ell+1} - 1.$$

By construction, the saturated acceptable set $\bar{A}$ associated to these saturating choices $(\mathcal{S}_i, \bar{B}_i)_{i\in\{1,\ldots,t\}}$ contains $(A)_i$. □

Definition 4.2.9. Given a saturated acceptable set $\bar{A} \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$, it is possible to associate to $\bar{A}$ a set of t-sequences $\left\{\left(d(\bar{A})_{1,b_1},\ldots,d(\bar{A})_{t,b_t}\right) : (b_1,\ldots,b_t) \in (\mathsf{Ch}^{[\mu]})_{\tilde{c}}^{t,\mathrm{w}}\right\}$, where we define:

$$d(\bar{A})_{i,b_i} := |\left\{(a)_i \in (\bar{A})_i : (a)_i^{[\mu]} = b_i\right\}| \qquad \forall i \in \{1,\ldots,t\}.$$

The result below provides a first upper bound on the cardinality of a saturated acceptable set by building on the t-sequences defined above.

Lemma 4.2.10. *Let $\bar{A} \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ be a $(k_1, \ldots, k_\mu)$-out-of-$(N_1, \ldots, N_\mu)$ saturated acceptable set. Then, for every $i \in \{1, \ldots, t\}$:*

$$\sum_{x \in \mathsf{Ch}^{[\mu]}} d(\bar{A})_{i,x} = \sum_{\ell=1}^{\mu} \left(\prod_{j=\ell+1}^{\mu} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right); \tag{4.2}$$

$$d(\bar{A})_{i,\tilde{c}} \geq \sum_{\ell=1}^{\mu-1} \left(\prod_{j=\ell+1}^{\mu-1} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right); \tag{4.3}$$

and $|\bar{A}| = \sum_{b \in (\mathsf{Ch}^{[\mu]})_{\tilde{c}}^{t,\mathrm{w}}} \prod_{i=1}^{t} d(\bar{A})_{i,(b)_i}$.

Proof. Let $\bar{A} \subseteq \mathsf{Ch}_{\tilde{c}}^{t,\mathrm{w}}$ be a $(k_1, \ldots, k_\mu)$-out-of $(N_1, \ldots, N_\mu)$ saturated acceptable set. By Definition 4.2.5 and 4.2.6, we have that, for all $i \in \{1, \ldots, t\}$, $\ell \in \{0, \ldots, \mu - 1\}$, and $s \in \mathcal{S}_i$ such that $|s| = \ell$:

$$|\mathrm{sat}_i(s, \bar{B}_i(s))| = (k_{\ell+1} - 1) \cdot \prod_{j=\ell+2}^{\mu} N_j.$$

Moreover, note that for $\ell > 0$ there are $N_\ell - k_\ell + 1$ choices for the last element of s if we fix the first $\ell - 1$, so in total we have that:

$$|\{s \in \mathcal{S}_i : |s| = \ell\}| = \prod_{j=1}^{\ell} (N_j - k_j + 1).$$

Putting everything together, setting to 1 the empty product, we have that:

$$\left| \bigsqcup_{s \in \mathcal{S}_i \,:\, |s| = \ell} \mathrm{sat}_i(s, \bar{B}_i(s)) \right| = \left(\prod_{j=1}^{\ell} (N_j - k_j + 1) \right) (k_{\ell+1} - 1) \left(\prod_{j=\ell+2}^{\mu} N_j \right),$$

so, by Definition 4.2.7 we have the following relation:

$$|(\bar{A})_i| = \sum_{\ell=1}^{\mu} \left(\prod_{j=\ell+1}^{\mu} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right). \tag{4.4}$$

Thanks to the second condition of Definition 4.2.7, we have that:

$$(\bar{A})_i = \bigsqcup_{x \in \mathsf{Ch}^{[\mu]}} \left\{ (a)_i \in (\bar{A})_i : (a)_i^{[\mu]} = x \right\},$$

as for every $(y)_i \in (\bar{A})_i$ there exist $(y)_1, \ldots, (y)_{i-1}, (y)_i, \ldots, (y)_t$ such that $((y)_1, \ldots, (y)_t) \in \bar{A}$. So by Definition 4.2.9 we have proved Equation (4.2). Another direct consequence of this is that $|\bar{A}| = \sum_{b \in \{\mathsf{Ch}^{[\mu]}\}_{\tilde{c}}^{t,w}} \prod_{i=1}^{t} d(\bar{A})_{i,(b)_i}$.

Finally, for any $i \in \{1, \ldots, t\}$ we have that:

$$d(\bar{A})_{i,\tilde{c}} \geq \sum_{\ell=1}^{\mu-1} \left(\prod_{j=\ell+1}^{\mu-1} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right),$$

as the right-hand side corresponds to the number of saturated branches in $(\bar{A})_i$, i.e. the number of elements $((y)_i^{[1]}, \ldots, (y)_i^{[\mu-1]}) \in \prod_{i=1}^{\mu-1} \mathsf{Ch}^{[i]}$ such that, for every x in $\mathsf{Ch}^{[\mu]}$, the element $((y)_i^{[1]}, \ldots, (y)_i^{[\mu-1]}, x)$ belongs to $(\bar{A})_i$. □

The result we present below is a technical lemma regarding the property of a particular sequence of sequences It will be a key step in proving the maximum cardinality among the acceptable sets.

Lemma 4.2.11. *Let* $\mathrm{w} \leq t$, N, Z_0, Z_1, Z_2 *be positive integers and consider* t *sequences of non-negative integers* $(d_{i,0}, \ldots, d_{i,N})_{i\in\{1,\ldots,t\}}$ *such that, for every* $i \in \{1, \ldots, t\}$, *it holds that:*

$$d_{i,0} \in \{Z_0, Z_2\}, \qquad \sum_{j=0}^{N} d_{i,j} = Z_1.$$

Let $\alpha := |\{i : d_{i,0} = Z_0\}|$, $\ell_0 := \max(0, \mathrm{w} - t + \alpha)$, *and* $\ell_1 := \min(\mathrm{w}, \alpha)$. *Then:*

$$\sum_{b\in S_{t,\mathrm{w}}} \prod_{i=1}^{t} d_{i,b_i} = \sum_{\ell=\ell_0}^{\ell_1} \binom{\alpha}{\ell}\binom{t-\alpha}{\mathrm{w}-\ell} Z_0^{\ell}\,(Z_1 - Z_0)^{\alpha-\ell}\,(Z_2)^{\mathrm{w}-\ell}(Z_1 - Z_2)^{t-\alpha-\mathrm{w}+\ell}, \tag{4.5}$$

where $S_{t,\mathrm{w}} := \{0, \ldots, N\}_0^{t,\mathrm{w}}$ *(see Definition 4.1.2).*

Proof. Given $b \in S_{t,\mathrm{w}}$, let us define four support sets:

$$\begin{aligned}
B_{1,b} &:= \{i \in \{1, \ldots, t\} : b_i = 0 \wedge d_{i,0} = Z_0\},\\
B_{2,b} &:= \{i \in \{1, \ldots, t\} : b_i = 0 \wedge d_{i,0} = Z_2\},\\
B_{3,b} &:= \{i \in \{1, \ldots, t\} : b_i \neq 0 \wedge d_{i,0} = Z_0\},\\
B_{4,b} &:= \{i \in \{1, \ldots, t\} : b_i \neq 0 \wedge d_{i,0} = Z_2\}.
\end{aligned}$$

It clearly holds that $t = |\bigsqcup_{j=1}^{4} B_{j,b}|$, $\alpha = |B_{1,b} \bigsqcup B_{3,b}|$, $\mathrm{w} = |B_{1,b} \bigsqcup B_{2,b}|$. Therefore, we have:

$$\begin{aligned}
\sum_{b\in S_{t,\mathrm{w}}} \prod_{i=1}^{t} d_{i,b_i} &= \sum_{b\in S_{t,\mathrm{w}}} \prod_{j=1}^{4} \left(\prod_{i\in B_{j,b}} d_{i,b_i} \right)\\
&= \sum_{b\in S_{t,\mathrm{w}}} \prod_{i\in B_{1,b}} d_{i,0} \prod_{i\in B_{2,b}} d_{i,0} \prod_{i\in B_{3,b}} d_{i,b_i} \prod_{i\in B_{4,b}} d_{i,b_i}\\
&= \sum_{b\in S_{t,\mathrm{w}}} \prod_{i\in B_{1,b}} Z_0 \prod_{i\in B_{2,b}} Z_2 \prod_{i\in B_{3,b}} d_{i,b_i} \prod_{i\in B_{4,b}} d_{i,b_i}.
\end{aligned}$$

Now let us consider two disjunct sets $B_1, B_2 \subseteq \{1, \ldots, t\}$, with $\ell := |B_1|$. In order for the set $B' := \{b \in S_{t,\mathrm{w}} : B_{1,b} = B_1 \wedge B_{2,b} = B_2\}$ to be non-empty, we have that $\ell \leq \min(\mathrm{w}, \alpha)$, and also $|B_2| = \mathrm{w} - \ell \leq t - \alpha$, i.e., $\ell \geq \max(0, \mathrm{w} - t + \alpha)$. Now, we have that:

$$\begin{aligned}
\sum_{b\in B'} \prod_{i=1}^{t} d_{i,b_i} &= \sum_{b\in B'} \prod_{i\in B_{1,b}} Z_0 \prod_{i\in B_{2,b}} Z_2 \prod_{i\in B_{3,b}} d_{i,b_i} \prod_{i\in B_{4,b}} d_{i,b_i}\\
&= \sum_{b\in B'} (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell} \prod_{i\in B_{3,b}} d_{i,b_i} \prod_{i\in B_{4,b}} d_{i,b_i}.
\end{aligned}$$

It is straightforward to see that, for any $i' \in \{1, \ldots, t\}$, the equality below holds:

$$S_{t,\mathrm{w}} = \left(\bigsqcup_{b' \in S_{t-1,\mathrm{w}}} \bigsqcup_{j=1}^{N} \left\{ (b'_1, \ldots, b'_{i'-1}, j, b'_{i'}, \ldots, b'_{t-1}) \right\} \right) \bigsqcup \left(\bigsqcup_{b' \in S_{t-1,\mathrm{w}-1}} \left\{ (b'_1, \ldots, b'_{i'-1}, 0, b'_{i'}, \ldots, b'_{t-1}) \right\} \right).$$

Now we want to adapt this partition to our set $B' \subseteq S_{t,\mathrm{w}}$ by considering an index i' in $\{1, \ldots, t\} \setminus (B_1 \bigsqcup B_2)$. Let us define an index-translation function $f_{i'} : \{1, \ldots, t-1\} \longrightarrow \{1, \ldots, t\} \setminus \{i'\}$:

$$f_{i'}(i) := \begin{cases} i & \text{if } i < i'; \\ i+1 & \text{if } i \geq i'. \end{cases}$$

Then, we can define the following subsets of $\{1, \ldots, t-1\}$:

$$B_1(i') := \{i : f_{i'}(i) \in B_1\}, \qquad B_2(i') := \{i : f_{i'}(i) \in B_2\},$$

and introduce the set:

$$B'(i') := \{b \in S_{t-1,\mathrm{w}} : \{i \in \{1, \ldots, t-1\} : b_i = 0 \wedge d_{f_{i'}(i),0} = Z_0\} = B_1(i') \wedge \\ \wedge \{i \in \{1, \ldots, t-1\} : b_i = 0 \wedge d_{f_{i'}(i),0} = Z_2\} = B_2(i')\}.$$

We have that:

$$B' = \bigsqcup_{b' \in B'(i')} \bigsqcup_{j=1}^{N} \left\{ (b'_1, \ldots, b'_{i'-1}, j, b'_{i'}, \ldots, b'_{t-1}) \right\}.$$

Let us define also:

$$B_3(i') := \{i \in \{1, \ldots, t-1\} : f_{i'}(i) \notin B_1 \bigsqcup B_2 \wedge d_{f_{i'}(i),0} = Z_0\};$$
$$B_4(i') := \{i \in \{1, \ldots, t-1\} : f_{i'}(i) \notin B_1 \bigsqcup B_2 \wedge d_{f_{i'}(i),0} = Z_2\}.$$

We can use this partition in our sum. We first assume $d_{i',0} = Z_0$, which leads to:

$$\begin{aligned}
\sum_{b\in B'}\prod_{i=1}^{t} d_{i,b_i} &= (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}\sum_{b\in B'}\left(\prod_{i\in B_{3,b}} d_{i,b_i}\prod_{i\in B_{4,b}} d_{i,b_i}\right)\\
&= (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}\sum_{b'\in B'(i')}\sum_{j=1}^{N}\left(d_{i',j}\prod_{i\in B_3(i')} d_{f_{i'}(i),b'_i}\prod_{i\in B_4(i')} d_{f_{i'}(i),b'_i}\right)\\
&= (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}\sum_{b'\in B'(i')}\left(\sum_{j=1}^{N} d_{i',j}\right)\left(\prod_{i\in B_3(i')} d_{f_{i'}(i),b'_i}\prod_{i\in B_4(i')} d_{f_{i'}(i),b'_i}\right)\\
&= (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}\sum_{b'\in B'(i')}\left(\left(\sum_{j=0}^{N} d_{i',j}\right)-d_{i',0}\right)\left(\prod_{i\in B_3(i')} d_{f_{i'}(i),b'_i}\prod_{i\in B_4(i')} d_{f_{i'}(i),b'_i}\right)\\
&= (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}\sum_{b'\in B'(i')}(Z_1-Z_0)\left(\prod_{i\in B_3(i')} d_{f_{i'}(i),b'_i}\prod_{i\in B_4(i')} d_{f_{i'}(i),b'_i}\right)\\
&= (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}(Z_1-Z_0)\sum_{b'\in B'(i')}\left(\prod_{i\in B_3(i')} d_{f_{i'}(i),b'_i}\prod_{i\in B_4(i')} d_{f_{i'}(i),b'_i}\right).
\end{aligned}$$

Note that in this way we have effectively extracted the factor corresponding to the index i' without affecting the other indices, and note also that in the case where $d_{i',0} = Z_2$, the only difference is that we can factor out (Z_1-Z_2) instead of (Z_1-Z_0). If we repeat this technique starting from $B'(i')$ (whose elements have length $t-1$) instead of B' (whose elements have length t), we can factor out another index. This means that, by repeating the same partition and factoring technique for every $i' \in \{1,\dots,t\}\setminus(B_1 \bigsqcup B_2)$, and remembering that for any $b \in B'$ we have $|B_{3,b}| = \alpha-\ell$ and $|B_{4,b}| = t-\alpha-\mathrm{w}+\ell$, we obtain:

$$\sum_{b\in B'}\prod_{i=1}^{t} d_{i,b_i} = (Z_0)^{\ell}(Z_2)^{\mathrm{w}-\ell}(Z_1-Z_0)^{\alpha-\ell}(Z_1-Z_2)^{t-\alpha-\mathrm{w}+\ell}.$$

To conclude, note that for ℓ fixed there are $\binom{\alpha}{\ell}$ possible choices for B_1 and $\binom{t-\alpha}{\mathrm{w}-\ell}$ possible choices for B_2, and that by varying ℓ, the set of all the possible B' forms a partition of $S_{t,\mathrm{w}}$. □

The following result provides an upper bound for the cardinalities of (saturated) acceptable sets.

Proposition 4.2.12. *Denote by $n_{t,\mathrm{w}}$ the maximum cardinality of a $(k_1,\dots,k_\mu)$-out-of-$(N_1,\dots,N_\mu)$ acceptable set. Then, $n_{t,\mathrm{w}}$ is the maximum of the expression:*

$$\sum_{\ell=\max(0,\mathrm{w}-t+\alpha)}^{\min(\mathrm{w},\alpha)}\binom{\alpha}{\ell}\binom{t-\alpha}{\mathrm{w}-\ell}Z_0^{\ell}\,(Z_1-Z_0)^{\alpha-\ell}\,(Z_2)^{\mathrm{w}-\ell}(Z_1-Z_2)^{t-\alpha-\mathrm{w}+\ell},$$

where the maximum is taken over $\alpha \in \{0, \dots, t\}$ *and*

$$Z_0 := \prod_{\ell=1}^{\mu-1} N_\ell; \tag{4.6}$$

$$Z_1 := \sum_{\ell=1}^{\mu} \left(\prod_{j=\ell+1}^{\mu} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right); \tag{4.7}$$

$$Z_2 := \sum_{\ell=1}^{\mu-1} \left(\prod_{j=\ell+1}^{\mu-1} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right). \tag{4.8}$$

Proof. Notice that, thanks to Lemma 4.2.8, we can always limit ourselves to consider saturated acceptable sets $\bar{A}$ for which, thanks to Lemma 4.2.10, the following conditions hold for every $i \in \{1, \dots, t\}$:

$$\sum_{x \in \mathsf{Ch}^{[\mu]}} d(\bar{A})_{i,x} = Z_1, \qquad d(\bar{A})_{i,\tilde{c}} \geq Z_2, \tag{4.9}$$

with Z_1, Z_2 as in Eq. (4.7) and (4.8), respectively. In addition,

$$n_{t,\mathrm{w}} := \max_{A} \{|A|\} = \max_{\bar{A}} \{|\bar{A}|\} = \max_{\bar{A}} \left\{ \sum_{b \in (\mathsf{Ch}^{[\mu]})^{t,\mathrm{w}}_{\tilde{c}}} \prod_{i=1}^{t} d(\bar{A})_{i,(b)_i} \right\}. \tag{4.10}$$

For simplicity of notation we remove below the dependence from $\bar{A}$, meaning that we replace $d(\bar{A})_{i,x}$ with $d_{i,x}$, with $x \in \{0, \dots, N_\mu - 1\}$. Furthermore, we biject the elements of $\mathsf{Ch}^{[\mu]}$ with the set $\{0, \dots, N_\mu - 1\}$ mapping $\tilde{c}$ to 0, and again we define $S_{t,\mathrm{w}} := \{0, \dots, N_\mu - 1\}^{t,\mathrm{w}}_0$. According to this, we will also denote by $\{(\bar{d}_{i,0}, \dots, \bar{d}_{i,N_\mu-1})\}_{i \in \{1,\dots,t\}}$ a generic set that respects the conditions imposed by Equation (4.9). We would therefore like to find a valid assignment of $(\bar{d}_{i,j})_{i,j}$ that maximizes the expression above. We show that it is sufficient to consider sets such that, for each index $i \in \{1, \dots, t\}$,

$$\left(\bar{d}_{i,0}, \sum_{j=1}^{N_\mu - 1} \bar{d}_{i,j} \right) \in \{(Z_0, Z_1 - Z_0), (Z_2, Z_1 - Z_2)\}. \tag{4.11}$$

Consider the case $i = 1$ (the extension to the generic case $i \neq 1$ is immediate) and notice that:

$$n_{t,\mathrm{w}} = \max \left\{ \bar{d}_{1,0} \sum_{(b_2,\dots,b_t) \in S_{t-1,\mathrm{w}-1}} \prod_{i=2}^{t} \bar{d}_{i,b_i} + \left(\sum_{j=1}^{N_\mu - 1} \bar{d}_{1,j} \right) \sum_{(b_2,\dots,b_t) \in S_{t-1,\mathrm{w}}} \prod_{i=2}^{t} \bar{d}_{i,b_i} \right\}.$$

Let us consider the two sums in the expression above, namely

$$\sum_{(b_2,\dots,b_t) \in S_{t-1,\mathrm{w}-1}} \prod_{i=2}^{t} \bar{d}_{i,b_i} \quad \text{and} \quad \sum_{(b_2,\dots,b_t) \in S_{t-1,\mathrm{w}}} \prod_{i=2}^{t} \bar{d}_{i,b_i}.$$

Depending on whether the first sum is larger or smaller than the second, we can maximize the entire expression by maximizing the factor that multiplies the largest sum. According to this, we can maximize $n_{t,\mathrm{w}}$ by maximizing either $\bar{d}_{1,0}$ or $\sum_{j=1}^{N_\mu - 1} \bar{d}_{1,j}$. In particular, if we have that:

$$\sum_{(b_2,\dots,b_t) \in S_{t-1,\mathrm{w}-1}} \prod_{i=2}^{t} \bar{d}_{i,b_i} \geq \sum_{(b_2,\dots,b_t) \in S_{t-1,\mathrm{w}}} \prod_{i=2}^{t} \bar{d}_{i,b_i},$$

then the maximum is obtained when we take $\bar{d}_{1,0} = \prod_{\ell=1}^{\mu-1} N_\ell = Z_0$ (and consequently $\sum_{j=1}^{N_\mu - 1} \bar{d}_{1,j} = Z_1 - Z_0$). Otherwise, since $\bar{d}_{1,0}$ is always greater or equal than Z_2, we can maximize the expression by choosing $\sum_{j=1}^{N_\mu - 1} \bar{d}_{1,j} = Z_1 - Z_2$ and $\bar{d}_{1,0} = Z_2$.

Since we can consider only sets of this type when maximizing Eq. 4.10, Lemma 4.2.11 applies with Z_0, Z_1 and Z_2 respectively as in Eq. (4.6), (4.7) and (4.8), $N := N_\mu - 1$. This concludes the proof. □

4.2.2 Fixed-Weight Repetition of a k-Special-Sound Sigma Protocol

Let (P, V) be a k-special-sound Sigma protocol, with challenge space Ch of cardinality N. We make a two-fold assumption on the protocol:

- the knowledge error $(k-1)/N$ is not negligible in the security parameter;
- the response size for a specific challenge $\tilde{c} \in \mathsf{Ch}$ significantly exceeds the response sizes for the other challenges.

To reduce the knowledge error while limiting the increase in the overall response size, a common technique is to repeat the interactive proof in parallel t times, with exactly w repetitions using the unfavourable challenge $\tilde{c}$. We denote by $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ the resulting protocol. In this section, we want to prove that such scheme is knowledge sound. To this end, we first slightly generalize the notation and results in [AF22a] by, at times, replacing the uniform distribution with an arbitrary one.

A dishonest prover against the Sigma protocol (P, V) can be described as an arbitrary (possibly probabilistic) algorithm $\mathcal{A}\colon \mathsf{Ch} \to \{0,1\}^*$. Let $\mathsf{Vf}\colon \mathsf{Ch} \times \{0,1\}^* \to \{0,1\}$ be a verification function. Throughout this section $\mathscr{D}$ will denote a probability distribution over Ch with support Ch. We define the *$\mathscr{D}$-success probability* of $\mathcal{A}$ as

$$\varepsilon^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D}) = \Pr[\mathsf{Vf}(C, \mathcal{A}(C)) = 1],$$

where the probability space is defined by C being sampled from Ch according to the probability distribution $\mathscr{D}$ and by the randomness of $\mathcal{A}$. When $\mathscr{D}$ is the uniform distribution over Ch, we simply write $\varepsilon^{\mathsf{Vf}}(\mathcal{A})$. Similarly, we adapt the worst-case success probability of $\mathcal{A}$ for a random challenge when $k-1$ challenges are removed from Ch, as follows:

$$\delta_k^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D}) = \min_{S \subset \mathsf{Ch} : |S| = k-1} \Pr[\mathsf{Vf}(C, \mathcal{A}(C)) = 1 \mid C \notin S].$$

It is easily seen that $\delta_1^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D}) = \varepsilon^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D})$. Moreover, in the following lemma we prove that $\delta_k^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D})$ is a decreasing function in k for any choice of $\mathscr{D}$.

Lemma 4.2.13. *Let $\mathscr{D}$ be a probability distribution over Ch. Then, for all $k \in \mathbb{N}^*$,*

$$\delta_{k+1}^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D}) \leq \delta_k^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D}).$$

Proof. Let C be a random variable distributed as $\mathscr{D}$ and let $S \subseteq \mathsf{Ch}$ be such that it minimizes $\delta_k^{\mathsf{Vf}}(\mathcal{A}, \mathscr{D})$. Moreover, let $\bar{S} = \mathsf{Ch} \setminus S$ and $\bar{S}' = \{c \in \bar{S} \mid \mathsf{Vf}(c, \mathcal{A}(c)) = 1\}$. Then,

for any $c' \in \bar{S}'$, we have

$$\begin{aligned}\delta^{\mathsf{Vf}}_{k+1}(\mathcal{A},\mathscr{D}) &\leq \Pr\left[\mathsf{Vf}(C,\mathcal{A}(C))=1 \;\middle|\; C \notin S \cup \{c'\}\right] \\ &= \frac{\left(\sum_{c\in\bar{S}'}\Pr[C=c]\right)-\Pr[C=c']}{\left(\sum_{c\in\bar{S}}\Pr[C=c]\right)-\Pr[C=c']} \\ &\leq \frac{\sum_{c\in\bar{S}'}\Pr[C=c]}{\sum_{c\in\bar{S}}\Pr[C=c]} = \delta^{\mathsf{Vf}}_{k}(\mathcal{A},\mathscr{D}).\end{aligned}$$

where the second inequality follows by observing that $\bar{S}' \subseteq \bar{S}$, and $\sum_{c\in\bar{S}'}\Pr[C=c] \leq \sum_{c\in\bar{S}}\Pr[C=c]$. □

In the following, we also consider the restriction $\mathscr{D}|_S$ of $\mathscr{D}$ to a subset $S \subseteq \mathsf{Ch}$.

Definition 4.2.14 (Distribution Restriction)**.** Let $\mathscr{D}$ be a probability distribution over Ch and let $X \sim \mathscr{D}$. For any subset $S \subseteq \mathsf{Ch}$, the *restriction* $\mathscr{D}|_S$ of $\mathscr{D}$ to S is defined by the following density function

$$\Pr[X|_S = x] = \frac{\Pr[X=x]}{\sum_{x'\in S}\Pr[X=x']}, \qquad \text{for all } x \in S.$$

Input: $k \in \mathbb{N}^*$, Ch a finite set with $|\mathsf{Ch}| = N \geq k$ and $S \subseteq \mathsf{Ch}$ with $|S| \geq k$.
Oracle access: algorithm $\mathcal{A}\colon \mathsf{Ch} \to \{0,1\}^*$ and verification function $\mathsf{Vf}\colon \mathsf{Ch}\times\{0,1\}^* \to \{0,1\}$.
Output: if successful, $(c_1,y_1),\ldots,(c_k,y_k) \in \mathsf{Ch}\times\{0,1\}^*$ with $\mathsf{Vf}(c_i,y_i)=1$ for all i and $c_i \neq c_j$ for $i \neq j$, otherwise $\perp$.

Sample $c_1 \in S$ according to $\mathscr{D}|_S$ and obtain $y_1 \leftarrow \mathcal{A}(c_1)$
if $\mathsf{Vf}(c_1,y_1)=0$ **then** abort and output $\perp$
end if
if $\mathsf{Vf}(c_1,y_1)=1$ and $k=1$ **then** output $(c_1,y_1) \in \mathsf{Ch}\times\{0,1\}^*$
else
  repeat
    set $S' = S \setminus \{c_1\}$ and run $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D}|_{S'})$
    set $\mathsf{coin} \leftarrow \mathsf{Vf}(d,\mathcal{A}(d))$ with $d \in S$ sampled according to $\mathscr{D}|_S$
  until $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D}|_{S'})$ outputs $(c_2,y_2),\ldots,(c_k,y_k)$ or $\mathsf{coin}=1$
end if
if $\mathsf{coin}=1$ **then return** $\perp$
else return $(c_1,y_1),\ldots,(c_k,y_k)$
end if

Figure 4.1: Extractor $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D}|_S)$

A simple adaptation of [AF22a, Lemma 2] proves the existence of an extraction algorithm $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D})$ - with oracle access to $\mathcal{A}$ and that samples challenges from Ch following the distribution $\mathscr{D}$ - which runs in expected polynomial time and succeeds with probability at least $\delta^{\mathsf{Vf}}_k(\mathcal{A},\mathscr{D})/k$. The extraction algorithm is described in Figure 4.1.

Lemma 4.2.15. *Let $k \in \mathbb{N}^*$, Ch be a finite set with cardinality $N \geq k$, $\mathsf{Vf}\colon \mathsf{Ch}\times\{0,1\}^* \to \{0,1\}$ an arbitrary function and $\mathscr{D}$ an arbitrary probability distribution over Ch. Then there exists an algorithm $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D})$ so that, given oracle access to any (probabilistic) algorithm $\mathcal{A}\colon \mathsf{Ch} \to \{0,1\}^*$, $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D})$ requires an expected number of at most $2k-1$ queries to $\mathcal{A}$ and, with probability at least $\delta^{\mathsf{Vf}}_k(\mathcal{A},\mathscr{D})/k$, it outputs k pairs $(c_1,y_1),(c_2,y_2),\ldots,(c_k,y_k) \in \mathsf{Ch}\times\{0,1\}^*$ with $\mathsf{Vf}(c_i,y_i)=1$ for all i and $c_i \neq c_j$ for all $i \neq j$.*

Proof. The proof is similar to the proof of [AF22a, Lemma 2]. In the original proof, the extractor samples the challenges uniformly from a subset $S \subseteq \mathsf{Ch}$. Here, we need to consider the natural restriction of $\mathscr{D}$ to S as per Def. 4.2.14. Since we are assuming that $\mathscr{D}$ has support equal to Ch, the restriction is well-defined. Then Lemma 4.2.13 is enough to adapt the proof of [AF22a, Lemma 2] and obtain the claim. □

Knowledge Soundness of Fixed-Weight Repetitions

We now consider the (t, w)-fixed-weight repetition $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ of a k-out-of-N special-sound Sigma protocol (P, V). With reference to Definition 4.1.3 and 4.1.1, we assume that the challenge space for (P, V) is $\mathsf{Ch} = \{0, \dots, N-1\}$ and the unfavourable challenge is $\tilde{c} = 0$, and we write $\mathsf{wt}(c)$ in place of $\mathsf{wt}_0(c)$.

The uniform distribution on $\mathsf{Ch}^{t,\mathrm{w}} = \mathsf{Ch}_0^{t,\mathrm{w}}$ induces t probability distributions $\mathscr{D}_i$ on Ch, obtained by taking the i-th component of a challenge uniformly sampled from $\mathsf{Ch}^{t,\mathrm{w}}$.

Definition 4.2.16. For every $i \in \{1, \dots, t\}$, we define the probability distribution $\mathscr{D}_i$ over Ch as the probability distribution having the following density function:

$$\Pr[X_i = a] = \frac{|\{c \in \mathsf{Ch}^{t,\mathrm{w}} \mid (c)_i = a\}|}{|\mathsf{Ch}^{t,\mathrm{w}}|}, \qquad \text{for all } a \in \mathsf{Ch}.$$

An adversary against $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ is described by a (possibly-probabilistic) algorithm $\mathcal{A} \colon \mathsf{Ch}^{t,\mathrm{w}} \to \{0,1\}^*$. The success probability of $\mathcal{A}$ is defined as

$$\varepsilon^V(\mathcal{A}) = \Pr[\mathsf{Vf}(C, \mathcal{A}(C)) = 1],$$

for some verification algorithm $\mathsf{Vf} \colon \mathsf{Ch}^{t,\mathrm{w}} \times \{0,1\}^* \to \{0,1\}$, where C is a random variable uniformly distributed over $\mathsf{Ch}^{t,\mathrm{w}}$.

From $\mathcal{A}$, we can construct t algorithms $\mathcal{A}_1, \dots, \mathcal{A}_t$, considering only a single invocation of (P, V). In particular, each $\mathcal{A}_i$ takes as input a challenge $c_i \in \mathsf{Ch}$ and runs $y \leftarrow \mathcal{A}(c = (c_i, \bar{c}))$, where $\bar{c}$ is sampled uniformly at random from $\mathsf{Ch}^{t-1,\mathrm{w}-1}$ if $c_i = 0$ or from $\mathsf{Ch}^{t-1,\mathrm{w}}$ otherwise, and $\mathcal{A}$ appropriately reorder its input so that c_i is the i-th component of c (i.e. $(c)_i = c_i$). Finally, $\mathcal{A}_i$ returns y along with $\bar{c}$.

Notice that, when the input challenge c_i for $\mathcal{A}_i$ is sampled according to the probability distribution $\mathscr{D}_i$ over Ch (see Def. 4.2.16), then the inputs passed to $\mathcal{A}$ are uniformly distributed over $\mathsf{Ch}^{t,\mathrm{w}}$. In this case, for each $\mathcal{A}_i$, we can run the extractor $\mathsf{Ext}^{\mathcal{A}_i}(\mathscr{D}_i)$ of Fig. 4.1. From Lemma 4.2.15, the extraction succeeds with probability at least $\delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i)/k$, where

$$\delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) = \min_{S_i \subset \mathsf{Ch} : |S_i| = k-1} \Pr[\mathsf{Vf}(D_i, \mathcal{A}(D_i)) = 1 \mid D_i \notin S_i],$$

D_i is distributed as $\mathscr{D}_i$ and Vf appropriately reorder its input[2].

In the following lemma we show that, when executed in parallel, at least one of the extractors $\mathsf{Ext}^{\mathcal{A}_i}(\mathscr{D}_i)$ succeeds with high probability in producing k challenge-response pairs that verify Vf and such that the i-th components of the challenges are all distinct.

Lemma 4.2.17. *Let $k, t \in \mathbb{N}^*$, $1 \leq \mathrm{w} \leq t$ and Ch a finite set with cardinality $N \geq k$. Let $\mathsf{Vf} \colon \mathsf{Ch}^{t,\mathrm{w}} \times \{0,1\}^* \to \{0,1\}$ and let $\mathcal{A}$ be a (probabilistic) algorithm that takes as input $c \in \mathsf{Ch}^{t,\mathrm{w}}$ and returns a string $y \in \{0,1\}^*$. Then*

[2]The verification function for $\mathcal{A}_i$ is the same Vf considered for $\mathcal{A}$, but seen as a function of the form $\mathsf{Ch} \times (\mathsf{Ch}^{t-1} \times \{0,1\}^*)$.

$$\sum_{i=1}^{t} \delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) \geq \frac{\varepsilon^V(\mathcal{A}) - \kappa_{t,\mathrm{w}}}{1 - \bar{\kappa}},$$

where

- $\bar{\kappa} = \min\left\{\frac{\mathrm{w}}{t} + (k-2)\frac{t-\mathrm{w}}{t(N-1)}, (k-1)\frac{t-\mathrm{w}}{t(N-1)}\right\}$;
- $\kappa_{t,\mathrm{w}} = \binom{t}{\mathrm{w}}^{-1} \frac{\eta_{t,\mathrm{w}}}{(N-1)^{t-\mathrm{w}}}$, *with*

$$\eta_{t,\mathrm{w}} = \begin{cases} \binom{\mathrm{w}(k-1)}{\mathrm{w}}(k-2)^{\mathrm{w}(k-2)}(k-1)^{t-\mathrm{w}(k-1)} & \text{if } t \geq \mathrm{w}(k-1) \\ \binom{t}{\mathrm{w}}(k-2)^{t-\mathrm{w}} & \text{otherwise} \end{cases}.$$

Proof. Let $(C)_i$ be the i-th component of the random variable C uniformly distributed over $\mathsf{Ch}^{t,\mathrm{w}}$ and let Λ denote the event $\mathsf{Vf}(C, \mathcal{A}(C)) = 1$. Therefore $\Pr[\Lambda] = \varepsilon^{\mathsf{Vf}}(\mathcal{A})$. For $i \in \{1, \ldots, t\}$, let $S_i \subset \mathsf{Ch}$ be such that it minimizes $\delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i)$. Then,

$$\sum_{i=1}^{t} \delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) = \sum_{i=1}^{t} \Pr[\mathsf{Vf}(D_i, \mathcal{A}_i(D_i)) = 1 \mid D_i \notin S_i] = \sum_{i=1}^{t} \Pr[\Lambda \mid (C)_i \notin S_i]$$

as, for any $i \in \{1, \ldots, t\}$, D_i and $(C)_i$ are identically distributed and it holds that

$$\begin{aligned} \Pr[\mathsf{Vf}(D_i, \mathcal{A}_i(D_i)) = 1 \mid D_i \notin S_i] &= \Pr[\mathsf{Vf}((C)_i, \mathcal{A}_i((C)_i)) = 1 \mid (C)_i \notin S_i] \\ &= \Pr[\Lambda \mid (C)_i \notin S_i]. \end{aligned}$$

From elementary probability, it follows that

$$\begin{aligned} \sum_{i=1}^{t} \Pr[\Lambda \mid (C)_i \notin S_i] &= \sum_{i=1}^{t} \frac{\Pr[\Lambda \wedge (C)_i \notin S_i]}{\Pr[(C)_i \notin S_i]} \\ &= \sum_{i=1}^{t} \frac{\Pr[\Lambda \wedge (C)_i \notin S_i]}{1 - \Pr[(C)_i \in S_i]} \\ &\geq \frac{\Pr[\Lambda \wedge \bigcup_i (C)_i \notin S_i]}{1 - \bar{\kappa}} \\ &\geq \frac{\Pr[\Lambda] - \Pr[\bigcap_i (C)_i \in S_i]}{1 - \bar{\kappa}}, \end{aligned}$$

where in the first inequality we can take $1/(1-\bar{\kappa})$ out of the sum by observing that, for any $i \in \{1, \ldots, t\}$,

$$\Pr[(C)_i \in S_i] = \begin{cases} \frac{\mathrm{w}}{t} + (k-2)\frac{t-\mathrm{w}}{t(N-1)} & \text{if } 0 \in S_i \\ (k-1)\frac{t-\mathrm{w}}{t(N-1)} & \text{otherwise} \end{cases}.$$

Since $\bar{\kappa}$ is defined as the minimum of the previous expression, it holds that $\Pr[(C)_i \in S_i] \geq \bar{\kappa}$.

Moreover, let us define

$$\kappa_{t,\mathrm{w}} = \max_{S_1,\ldots,S_t} \Pr[(C)_1 \in S_1 \wedge (C)_2 \in S_2 \wedge \ldots \wedge (C)_t \in S_t],$$

where the maximum is over all sets $S_i \subset \mathsf{Ch}$ with $|S_i| = k-1$. Notice that, equivalently, $\kappa_{t,\mathrm{w}} = \max_S \Pr[C \in S]$, where $S \subset \mathsf{Ch}^{t,\mathrm{w}}$ such that $|(S)_i| < k$ for all $i \in \{1,\dots,t\}$. The maximal size $\eta_{t,\mathrm{w}}$ of S is computed in Prop. 4.2.1 as

$$\eta_{t,\mathrm{w}} = \begin{cases} \binom{\mathrm{w}(k-1)}{\mathrm{w}}(k-2)^{\mathrm{w}(k-2)}(k-1)^{t-\mathrm{w}(k-1)} & \text{if } t \geq \mathrm{w}(k-1) \\ \binom{t}{\mathrm{w}}(k-2)^{t-\mathrm{w}} & \text{otherwise} \end{cases}.$$

Therefore

$$\kappa_{t,\mathrm{w}} = \Pr[C \in S] \leq \frac{\eta_{t,\mathrm{w}}}{|\mathsf{Ch}^{t,\mathrm{w}}|} = \binom{t}{\mathrm{w}}^{-1} \frac{\eta_{t,\mathrm{w}}}{(N-1)^{t-\mathrm{w}}},$$

which completes the proof. □

In light of Lemma 4.2.17, we can bound the probability that at least one extractor $\mathsf{Ext}^{\mathcal{A}_i}(\mathscr{D}_i)$ is successful as follows:

$$\max_{1\leq i\leq t} \delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) \geq \frac{1}{t}\sum_{i=1}^{t} \delta_k^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) \geq \frac{\varepsilon^{\mathsf{Vf}}(\mathcal{A}) - \kappa_{t,\mathrm{w}}}{t(1-\bar{\kappa})}.$$

As a consequence, the (t,w)-fixed-weight repetition $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ of a k-special-sound Sigma protocol (P,V) is knowledge sound with knowledge error $\kappa_{t,\mathrm{w}}$, which corresponds to the maximum cheating probability of a dishonest prover.

Theorem 4.2.18 (Fixed-Weight Repetition of a k-Special-Sound Sigma Protocol)**.** *Let (P,V) be a k-out-of-N special-sound Sigma protocol. Let $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ be the (t,w)-fixed-weight repetition of (P,V), where $k,t \in \mathbb{N}^*$ and $1 \leq \mathrm{w} \leq t$. Then $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ is knowledge sound with knowledge error $\kappa^{t,\mathrm{w}}$, where*

$$\kappa_{t,\mathrm{w}} = \binom{t}{\mathrm{w}}^{-1} \frac{\eta_{t,\mathrm{w}}}{(N-1)^{t-\mathrm{w}}},$$

with

$$\eta_{t,\mathrm{w}} = \begin{cases} \binom{\mathrm{w}(k-1)}{\mathrm{w}}(k-2)^{\mathrm{w}(k-2)}(k-1)^{t-\mathrm{w}(k-1)} & \textit{if } t \geq \mathrm{w}(k-1) \\ \binom{t}{\mathrm{w}}(k-2)^{t-\mathrm{w}} & \textit{otherwise} \end{cases}.$$

Remark 4.2.19. Recently, [AFR23] considered a further generalisation, named Γ-special soundness, of the notion of k-special soundness. Within such generalisation, the subsets of challenges from which it is possible to extract a witness are determined by an arbitrary access structure Γ, i.e. a monotone set of subsets of the challenge space. The authors proved that, for any Γ-special-sound Sigma protocol, it is possible to build an extractor that has knowledge error κ_Γ and an expected running time that scales with t_Γ, where κ_Γ, t_Γ are positive integers determined by Γ. Then, if t_Γ is polynomial, Γ-special soundness implies knowledge soundness. Moreover, they showed that both a k-special-sound Sigma protocol and its t-fold parallel repetition are Γ-special sound for a suitable access structure Γ, which led them to re-discover the results of [AF22a].
The (t,w)-fixed-weight repetition of a k-special-sound 3-round interactive proof can also be described within this framework, and the results of Theorem 4.2.18 can be obtained by techniques similar to that of [AFR23]. Unfortunately, it is not possible to find an access structure that suitably describes the (t,w)-fixed-weight repetition of a $(k_1,\dots,k_\mu)$-special-sound $(2\mu+1)$-round interactive proof. Therefore, with the goal of paving the way for the analysis of this second case, we have made the description of the extractor for Sigma protocols explicit, building on the techniques of [AF22a] rather than those of [AFR23].

4.2.3 Fixed-Weight Repetition of a $(k_1, \ldots, k_\mu)$-Special Sound Interactive Proof

In the following, we extend the result of Subsection 4.2.2 to multi-round interactive proofs. Let (P, V) be a $(k_1, \ldots, k_\mu)$-special-sound $(2\mu+1)$-round interactive proof with challenge space $\mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$. We define $K = \prod_{i=1}^{\mu} k_i$ and write $\mathbf{c} = (c^{[1]}, \ldots, c^{[\mu]})$ for an element $\mathbf{c} \in \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$.

Similarly to the case of 3-round interactive proofs, with the aim of reducing the knowledge error while limiting the increase in the overall response size, we consider the t-fold parallel repetition of the protocol (P, V), where exactly w repetitions use the unfavourable challenge $\tilde{c} \in \mathsf{Ch}^{[\mu]}$ in the last round. $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ is the resulting protocol. To show that this interactive proof is knowledge sound, we again start by slightly generalising the notation and results in [AF22a].

A dishonest prover against (P, V) can be described as an arbitrary (probabilistic) algorithm $\mathcal{A}\colon \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]} \to \{0,1\}^*$. Let $\mathsf{Vf}\colon \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]} \times \{0,1\}^* \to \{0,1\}$ be a verification function and $\mathscr{D} = (\mathscr{D}^{[1]}, \ldots, \mathscr{D}^{[\mu]})$ a collection of probability distributions, where $\mathscr{D}^{[i]}$ is over $D^{[i]} \subseteq \mathsf{Ch}^{[i]}$ with $|D^{[i]}| \geq k_i$. We define the *$\mathscr{D}$-success probability* of $\mathcal{A}$ as

$$\varepsilon^V(\mathcal{A}, \mathscr{D}) = \Pr[\mathsf{Vf}(C, \mathcal{A}(C)) = 1],$$

where $C = (C^{[1]}, \ldots, C^{[\mu]})$ is a random variable, with $C^{[i]}$ being distributed as $\mathscr{D}^{[i]}$. If C is uniformly distributed over $\mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$, we write $\varepsilon^V(\mathcal{A})$. Similarly, we adapt the punctured success probability of $\mathcal{A}$ as

$$\delta^{\mathsf{Vf}}_{\mathbf{k}}(\mathcal{A}, \mathscr{D}) = \min_{S^{[1]}, S^{[2]}(\cdot), \ldots, S^{[\mu]}(\cdot)} \Pr\left[\mathsf{Vf}(C, \mathcal{A}(C)) = 1 \;\middle|\; \begin{smallmatrix} C^{[1]} \notin S^{[1]} \wedge C^{[2]} \notin S^{[2]}(C^{[1]}) \wedge \cdots \\ \cdots \wedge C^{[\mu]} \notin S^{[\mu]}(C^{[1]}, \ldots, C^{[\mu-1]}) \end{smallmatrix}\right],$$

where the minimum is over all sets $S^{[1]} \in \mathsf{Ch}^{[1]}|_{k_1 - 1}$, and over all functions $S^{[i]}\colon \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[i-1]} \to \mathsf{Ch}^{[i]}|_{k_i - 1}$, with $i = 2, \ldots, \mu$. Here, for any $i \in \{1, \ldots, \mu\}$, $\mathsf{Ch}^{[i]}|_{k_i - 1}$ denotes the set of subsets of $\mathsf{Ch}^{[i]}$ with cardinality $k_i - 1$.

Next, we define an extraction algorithm $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D})$ with oracle access to $\mathcal{A}$ that samples the challenges according to the distribution $\mathscr{D}$. Building on [AF22a, Lemma 4], it is possible to show that $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D})$ runs in expected polynomial time and succeeds with probability at least $\delta^{\mathsf{Vf}}_{k}(\mathcal{A}, \mathscr{D})/K$.

Lemma 4.2.20. *Let $k_1, \ldots, k_\mu \in \mathbb{N}^*$, $K = \prod_{i=1}^{\mu} k_i$, $\mathsf{Ch}^{[1]}, \ldots, \mathsf{Ch}^{[\mu]}$ be finite sets with $\mathsf{Ch}^{[j]}$ having cardinality $N_j \geq k_j$, $\mathsf{Vf}\colon \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]} \times \{0,1\}^* \to \{0,1\}$ an arbitrary function and $\mathscr{D} = (\mathscr{D}^{[1]}, \ldots, \mathscr{D}^{[\mu]})$ a collection of probability distributions $\mathscr{D}^{[j]}$ with support equal to $\mathsf{Ch}^{[j]}$. Then, there exists an algorithm $\mathsf{Ext}^{\mathcal{A}}(\mathscr{D})$ that, given oracle access to a (probabilistic) algorithm $\mathcal{A}\colon \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]} \to \{0,1\}^*$, with an expected number of at most $2^\mu K$ queries to $\mathcal{A}$ and with probability at least $\delta^{\mathsf{Vf}}_{\mathbf{k}}(\mathcal{A}, \mathscr{D})/K$, it outputs K pairs $(\mathbf{c}_1, y_1), \ldots, (\mathbf{c}_K, y_K) \in \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]} \times \{0,1\}^*$ with $\mathsf{Vf}(\mathbf{c}_i, y_i) = 1$ for all $i \in \{1, \ldots, K\}$ and such that the vectors $\mathbf{c}_i$ form a $(k_1, \ldots, k_\mu)$-tree of transcripts.*

Proof. The proof resembles that of [AF22a, Lemma 4], with the only difference that the single-instance extractor used internally is an instantiation of the one described in Fig. 4.1. □

Knowledge-Soundness of Fixed-Weight Repetitions

Let $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ be the (t, w)-fixed-weight repetition – with respect to an unfavourable challenge $\tilde{c} \in \mathsf{Ch}^{[\mu]}$ – of a $(k_1, \ldots, k_\mu)$-special-sound $(2\mu+1)$-round interactive proof (P, V) with challenge space $\mathsf{Ch} = \mathsf{Ch}^{[1]} \times \cdots \mathsf{Ch}^{[\mu]}$.

For ease of notation, in the following we assume that the challenge space for the i-th round of (P,V) is $\mathsf{Ch}^{[i]} = \{0,\dots,N_i-1\}$ while the unfavourable challenge for the last round is $\tilde{c}=0$.

Definition 4.2.21. For each $j \in \{1,\dots,\mu-1\}$, let $\mathscr{U}^{[j]}$ be the uniform distribution over $\mathsf{Ch}^{[j]}$. For every $i \in \{1,\dots,t\}$, let $\mathscr{D}_i^{[\mu]}$ be the probability distribution having the following density function:

$$\Pr[X_i = a] = \frac{|\{c \in (\mathsf{Ch}^{[\mu]})_0^{t,\mathrm{w}} \mid (c)_i = a\}|}{|(\mathsf{Ch}^{[\mu]})_0^{t,\mathrm{w}}|}, \qquad \text{for all } a \in \mathsf{Ch}^{[\mu]}.$$

Finally, let $\mathscr{D}_i = (\mathscr{U}^{[1]},\dots,\mathscr{U}^{[\mu-1]},\mathscr{D}_i^{[\mu]})$.

An adversary against $(\mathsf{P}^{t,\mathrm{w}},\mathsf{V}^{t,\mathrm{w}})$ is described as a (possibly-probabilistic) algorithm which, on input a row $\mathbf{c} = ((\mathbf{c})_1,\dots,(\mathbf{c})_t)$ of columns $(\mathbf{c})_i = ((c)_i^{[1]},\dots,(c)_i^{[\mu]}) \in \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$ of challenges such that $\mathsf{wt}_0((c)_1^{[\mu]},\dots,(c)_t^{[\mu]}) = \mathrm{w}$, outputs a string $y \in \{0,1\}^*$. The success probability of $\mathcal{A}$ is defined as

$$\varepsilon^V(\mathcal{A}) = \Pr[\mathsf{Vf}(C,\mathcal{A}(C)) = 1],$$

for some verification algorithm $\mathsf{Vf}\colon \mathsf{Ch}_0^{t,\mathrm{w}} \times \{0,1\}^* \to \{0,1\}$, with C being a random variable uniformly distributed over $\mathsf{Ch}_0^{t,\mathrm{w}}$.

Such an algorithm $\mathcal{A}$ induces t algorithms $\mathcal{A}_1,\dots,\mathcal{A}_t$, analogous to those considered in the context of a single instance of (P,V). Each $\mathcal{A}_i$ takes as input a column $(\mathbf{c})_i \in \mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$. Then $\mathcal{A}_i$ runs $y \leftarrow \mathcal{A}(\mathbf{c} = ((\mathbf{c})_i, \bar{\mathbf{c}}))$, where $\bar{\mathbf{c}}$ is sampled uniformly at random from $\mathsf{Ch}_0^{t-1,\mathrm{w}-1}$ if $(c)_i^{[\mu]} = 0$ or from $\mathsf{Ch}_0^{t-1,\mathrm{w}}$ otherwise, and $\mathcal{A}$ is understood to appropriately reorder its input so that $(\mathbf{c})_i$ is the i-th component of $\mathbf{c}$. Finally $\mathcal{A}_i$ returns y along with $\bar{\mathbf{c}}$.

Notice that, when the input challenge for $\mathcal{A}_i$ is sampled according to the probability distribution $\mathscr{D}_i = (\mathscr{U}^{[1]},\dots,\mathscr{U}^{[\mu-1]},\mathscr{D}_i^{[\mu]})$ over $\mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$, then the inputs passed to $\mathcal{A}$ are uniformly distributed over $\mathsf{Ch}_0^{t,\mathrm{w}}$. Hence, for each $\mathcal{A}_i$, we can consider the extractor $\mathsf{Ext}^{\mathcal{A}_i}(\mathscr{D}_i)$ of Lemma 4.2.20, which succeeds with probability at least $\delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i,\mathscr{D}_i)/K$, where

$$\delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i,\mathscr{D}_i) = \min_{S_i^{[1]},S_i^{[2]}(\cdot),\dots,S_i^{[\mu]}(\cdot)} \Pr\left[\mathsf{Vf}(D_i,\mathcal{A}_i(D_i)) = 1 \;\middle|\; \begin{smallmatrix} D_i^{[1]} \notin S_i^{[1]} \wedge D_i^{[2]} \notin S_i^{[2]}(D_i^{[1]}) \wedge \cdots \\ \cdots \wedge D_i^{[\mu]} \notin S_i^{[\mu]}(D_i^{[1]},\dots,D_i^{[\mu-1]}) \end{smallmatrix}\right],$$

and D_i is distributed as $\mathscr{D}_i$.

In the following lemma we show that, when executed in parallel, at least one of the extractors $\mathsf{Ext}^{\mathcal{A}_i}(\mathscr{D}_i)$ succeeds with high probability in producing $\prod_{i=1}^{\mu} k_i$ challenge-response pairs that verify Vf and such that the challenges form a $(k_1,\dots,k_\mu)$-tree of transcripts.

Lemma 4.2.22. *Let $k_1,\dots,k_\mu,t,w \in \mathbb{N}^*$ such that $1 \le \mathrm{w} \le t$ and let $\mathsf{Ch}^{[1]},\dots,\mathsf{Ch}^{[\mu]}$ be finite sets with $\mathsf{Ch}^{[j]}$ having cardinality $N_j \ge k_j$. Let $\mathsf{Vf}\colon \mathsf{Ch}_0^{t,\mathrm{w}} \times \{0,1\}^* \to \{0,1\}$ and let $\mathcal{A}$ be a (probabilistic) algorithm that takes as input an element $\mathbf{c}$ of $\mathsf{Ch}_0^{t,\mathrm{w}}$ and outputs a string $y \in \{0,1\}^*$. Then*

$$\sum_{i=1}^{t} \delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i,\mathscr{D}_i) \ge \frac{\varepsilon^V(\mathcal{A}) - \kappa_{t,\mathrm{w}}}{1-\bar{\kappa}},$$

where $\kappa_{t,\mathrm{w}}$ is the maximum, taken over $\alpha \in \{0,\dots,t\}$, of the expression:

$$\frac{\sum_{\ell=\max(0,\mathrm{w}-t+\alpha)}^{\min(\mathrm{w},\alpha)} \binom{\alpha}{\ell}\binom{t-\alpha}{\mathrm{w}-\ell} Z_0^{\ell}\,(Z_1-Z_0)^{\alpha-\ell}\,(Z_2)^{\mathrm{w}-\ell}(Z_1-Z_2)^{t-\alpha-\mathrm{w}+\ell}}{|\mathsf{Ch}_0^{t,\mathrm{w}}|},$$

with $|\mathsf{Ch}_0^{t,\mathrm{w}}| = \binom{t}{w}(N_\mu - 1)^{t-w}(\prod_{i=1}^{\mu-1} N_i)^t$ *and*

$$
\begin{aligned}
Z_0 &:= \prod_{\ell=1}^{\mu-1} N_\ell, \\
Z_1 &:= \sum_{\ell=1}^{\mu} \left(\prod_{j=\ell+1}^{\mu} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right), \\
Z_2 &:= \sum_{\ell=1}^{\mu-1} \left(\prod_{j=\ell+1}^{\mu-1} N_j \right) (k_\ell - 1) \left(\prod_{j=1}^{\ell-1} (N_j - k_j + 1) \right).
\end{aligned}
$$

Moreover, it holds that

$$
\bar{\kappa} \geq 1 - \left(\prod_{j-1}^{\mu-1} \frac{N_j - k_j + 1}{N_j} \right) \left(\frac{N_\mu - k_\mu + 1}{N_\mu - 1} \frac{t-w}{w} + \frac{w}{t} \right).
$$

Proof. Let $(C)_i$ be the i-th component of the random variable C uniformly distributed over $\mathsf{Ch}_0^{t,\mathrm{w}}$ and let Λ be the event $\mathsf{Vf}(C, \mathcal{A}(C)) = 1$. Therefore $\Pr[\Lambda] = \varepsilon^V(\mathcal{A})$. For $i \in \{1, \ldots, t\}$, let $S_i^{[1]}$ and $S_i^{[2]}(\cdot), \ldots, S_i^{[\mu]}(\cdot)$ be such that they minimize $\delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i)$. Moreover, we denote by Γ_i the event

$$
(C)_i^{[1]} \notin S_i^{[1]} \wedge (C)_i^{[2]} \notin S_i^{[2]}((C)_i^{[1]}) \wedge \cdots \wedge (C)_i^{[\mu-1]} \notin S_i^{[\mu-1]}((C)_i^{[1]}, \ldots, (C)_i^{[\mu-2]})
$$

and by Ω_i the event $(C)_i^{[\mu]} \notin S_i^{[\mu]}((C)_i^{[1]}, \ldots, (C)_i^{[\mu-1]})$. Furthermore, we consider the probability distribution D_i which is distributed as $\mathscr{D}_i = (\mathscr{U}^{[1]}, \ldots, \mathscr{U}^{[\mu-1]}, \mathscr{D}_i^{[\mu]})$ over $\mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$. Therefore, similarly to the proof of Lemma 4.2.17, D_i and $(C)_i$ are identically distributed for any $i \in \{1, \ldots, t\}$ and, by construction of the $\mathcal{A}_i$'s, it holds that

$$
\begin{aligned}
\sum_{i=1}^{t} \delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) &= \sum_{i=1}^{t} \Pr\left[\mathsf{Vf}(D_i, \mathcal{A}_i(D_i)) = 1 \;\middle|\; \begin{smallmatrix} D_i^{[1]} \notin S_i^{[1]} \wedge D_i^{[2]} \notin S_i^{[2]}(D_i^{[1]}) \wedge \cdots \\ \cdots \wedge D_i^{[\mu]} \notin S_i^{[\mu]}(D_i^{[1]}, \ldots, D_i^{[\mu-1]}) \end{smallmatrix} \right] \\
&= \sum_{i=1}^{t} \Pr[\Lambda \mid \Gamma_i \cap \Omega_i].
\end{aligned}
$$

From elementary probability, it follows that

$$
\begin{aligned}
\sum_{i=1}^{t} \Pr[\Lambda \mid \Gamma_i \cap \Omega_i] &= \sum_{i=1}^{t} \frac{\Pr[\Lambda \wedge (\Gamma_i \cap \Omega_i)]}{\Pr[\Gamma_i \cap \Omega_i]} = \sum_{i=1}^{t} \frac{\Pr[\Lambda \wedge (\Gamma_i \cap \Omega_i)]}{\Pr[\Gamma_1 \cap \Omega_1]} \\
&\geq \frac{\Pr[\Lambda \wedge \bigcup_i (\Gamma_i \cap \Omega_i)]}{1 - \bar{\kappa}} \geq \frac{\Pr[\Lambda] - \Pr\left[\bigcap_i \overline{(\Gamma_i \cap \Omega_i)}\right]}{1 - \bar{\kappa}},
\end{aligned}
$$

where in the first inequality we can take $1/(1-\bar{\kappa})$ out of the sum by defining $\bar{\kappa} = 1 - \Pr[\Gamma_1 \cap \Omega_1]$ and observing that $\Pr[\Gamma_1 \cap \Omega_1] = \cdots = \Pr[\Gamma_t \cap \Omega_t]$. To obtain the expression of $\bar{\kappa}$, first write $\Pr[\Gamma_1 \cap \Omega_1] = \Pr[\Gamma_1] \cdot \Pr[\Omega_1 \mid \Gamma_1]$ and notice that

$$
\Pr[\Gamma_1] = \prod_{j=1}^{\mu-1} \frac{N_j - k_j + 1}{N_j}.
$$

Moreover, observe that $\Pr[\Omega_1 \mid \Gamma_1] = \Pr\left[(C)_1^{[\mu]} \notin S_1^{[\mu]}\right]$ for some set $S_1^{[\mu]} \subset \mathsf{Ch}^{[\mu]}$ with $|S_1^{[\mu]}| = k_\mu - 1$. Now, let $\bar{S}_1^{[\mu]} = \mathsf{Ch}^{[\mu]} \setminus S_1^{[\mu]}$, then

$$\Pr\left[(C)_1^{[\mu]} \notin S_1^{[\mu]}\right] = \Pr\left[(C)_1^{[\mu]} \in \bar{S}_1^{[\mu]}\right] = \frac{\bar{\eta}}{|(\mathsf{Ch}^{[\mu]})_0^{t,\mathrm{w}}|},$$

where the value of $\bar{\eta}$ is given by the size of a maximal $\bar{S} \subseteq (\mathsf{Ch}^{[\mu]})_0^{t,\mathrm{w}}$ such that $|(\bar{S})_1| \leq N_\mu - k_\mu + 1$. Following the same techniques of Prop. 4.2.1, we can explicitly compute $\bar{\eta}$ depending on whether $0 \in \bar{S}_1^{[\mu]}$. It holds that

$$\begin{aligned}
\bar{\eta} &= \max\left\{\begin{matrix} \binom{t-1}{\mathrm{w}}(N_\mu - k_\mu + 1)(N_\mu - 1)^{t-\mathrm{w}-1} & \text{If } 0 \notin \bar{S}_1^{[\mu]} \\ \binom{t-1}{\mathrm{w}}(N_\mu - k_\mu)(N_\mu - 1)^{t-\mathrm{w}-1} + \binom{t-1}{\mathrm{w}-1}(N_\mu - 1)^{t-\mathrm{w}} & \text{If } 0 \in \bar{S}_1^{[\mu]} \end{matrix}\right\} \\
&\leq \binom{t-1}{\mathrm{w}}(N_\mu - k_\mu + 1)(N_\mu - 1)^{t-\mathrm{w}-1} + \binom{t-1}{\mathrm{w}-1}(N_\mu - 1)^{t-\mathrm{w}} \\
&= \binom{t}{w}(N_\mu - 1)^{t-w}\left(\frac{N_\mu - k_\mu + 1}{N_\mu - 1}\frac{t-w}{w} + \frac{w}{t}\right).
\end{aligned}$$

Since $|(\mathsf{Ch}^{[\mu]})_0^{t,\mathrm{w}}| = \binom{t}{\mathrm{w}}(N_\mu - 1)^{t-\mathrm{w}}$, we obtain

$$\Pr[\Gamma_1 \cap \Omega_1] \leq \left(\prod_{j=1}^{\mu-1}\frac{N_j - k_j + 1}{N_j}\right)\left(\frac{N_\mu - k_\mu + 1}{N_\mu - 1}\frac{t-w}{w} + \frac{w}{t}\right).$$

Now, let $\kappa_{t,\mathrm{w}} = \Pr\left[\bigcap_{i=1}^{t}\overline{(\Gamma_i \cap \Omega_i)}\right]$. For any $i \in \{1, \ldots, t\}$, the set $S_i^{[1]}$ and the maps $S_i^{[2]}(\cdot), \ldots, S_i^{[\mu]}(\cdot)$ identify a subset $(\bar{A})_i$ of $\mathsf{Ch}^{[1]} \times \cdots \times \mathsf{Ch}^{[\mu]}$ defined in the following way. The set $S_i^{[1]}$ dictates that $(\bar{A})_i$ contains

$$\left\{(c^{[1]}, c^{[2]}, \ldots, c^{[\mu]}) : c^{[1]} \in S_i^{[1]} \wedge (c^{[2]}, \ldots, c^{[\mu]}) \in \prod_{j=2}^{\mu}\mathsf{Ch}^{[j]}\right\}.$$

Furthermore, the set $S_i^{[1]}$ and the map $S_i^{[2]}(\cdot)$ dictate that $(\bar{A})_i$ also contains

$$\left\{(c^{[1]}, c^{[2]}, \ldots, c^{[\mu]}) : c^{[1]} \notin S_i^{[1]} \wedge c^{[2]} \in S_i^{[2]}(c^{[1]}) \wedge (c^{[3]}, \ldots, c^{[\mu]}) \in \prod_{j=3}^{\mu}\mathsf{Ch}^{[j]}\right\}.$$

By iterating this argument, we deduce that the sets $(\bar{A})_i$ have a form identical to the sets in the proof of Lemma 4.2.8 which have the same names, the only difference being that within that proof the sets $(\bar{A})_i$ were determined by a starting acceptable set A, while here they are determined by $S_i^{[1]}$ and $S_i^{[2]}(\cdot), \ldots, S_i^{[\mu]}(\cdot)$. As a consequence, $\kappa_{t,\mathrm{w}}$ corresponds to the probability of belonging to the set $\bar{A}$, which is defined as the set containing every element $((y)_1, \ldots, (y)_t) \in \prod_{i=1}^{t}(\bar{A})_i$ which respects the first condition of Definition 4.2.4, i.e. $((y)_1^{[\mu]}, \ldots, (y)_t^{[\mu]}) \in (\mathsf{Ch}^{[\mu]})_0^{t,\mathrm{w}}$. In other words, $\bar{A}$ is a saturated acceptable set, as defined in Definition 4.2.7, and then we can apply Proposition 4.2.12 to conclude that the probability $\kappa_{t,\mathrm{w}}$ is exactly the one in the claim.

□

As for the fixed-weight repetition of a Sigma protocol, following Lemma 4.2.22 we can bound the probability that at least one extractor $\mathsf{Ext}^{\mathcal{A}_i}(\mathscr{D}_i)$ is successful:

$$\max_{1\leq i\leq t} \delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) \geq \frac{1}{t}\sum_{i=1}^{t} \delta_{\mathbf{k}}^{\mathsf{Vf}}(\mathcal{A}_i, \mathscr{D}_i) \geq \frac{\varepsilon^{\mathsf{Vf}}(\mathcal{A}) - \kappa_{t,\mathrm{w}}}{t\cdot(1-\bar{\kappa})}.$$

It follows that the (t,w)-fixed-weight repetition $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ of a $(k_1,\ldots,k_\mu)$-special-sound $(2\mu+1)$-round interactive proof (P,V) is knowledge sound with knowledge error $\kappa_{t,\mathrm{w}}$, which again corresponds to the maximum cheating probability of a dishonest prover

Theorem 4.2.23 (Fixed-Weight Repetition of a $(k_1,\ldots,k_\mu)$-Special-Sound Multi-Round Interactive Proof)**.** *Let (P,V) be a $(k_1,\ldots,k_\mu)$-special-sound $(2\mu+1)$-round interactive proof and $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ be the (t,w)-fixed-weight repetition of (P,V), where $w,k,t\in\mathbb{N}^*$ and $1\leq \mathrm{w}\leq t$. Then $(\mathsf{P}^{t,\mathrm{w}}, \mathsf{V}^{t,\mathrm{w}})$ is knowledge sound with knowledge error $\kappa_{t,\mathrm{w}}$, where $\kappa_{t,\mathrm{w}}$ is the maximum, taken over $\alpha\in\{0,\ldots,t\}$, of the expression*

$$\frac{\sum_{\ell=\max(0,\mathrm{w}-t+\alpha)}^{\min(\mathrm{w},\alpha)} \binom{\alpha}{\ell}\binom{t-\alpha}{\mathrm{w}-\ell} Z_0^{\ell}\,(Z_1-Z_0)^{\alpha-\ell}\,(Z_2)^{\mathrm{w}-\ell}(Z_1-Z_2)^{t-\alpha-\mathrm{w}+\ell}}{\binom{t}{w}(N_\mu-1)^{t-w}(\prod_{i=1}^{\mu-1}N_i)^t},$$

where Z_0, Z_1, Z_2 are defined as in Lemma 4.2.22.

4.2.4 Applications and Conclusions

In this section, we have established a positive result about the security of fixed-weight parallel repetitions of special-sound (multi-round) interactive proofs. We have given an in-depth description of the optimal strategy of a dishonest prover attacking a fixed-weight repetition of an interactive proof. In particular, we have provided an explicit expression of the maximum adversary's cheating probability, for both the 3-round and multi-round cases. Next, we have generalized the knowledge extractor from [AF22a], applying it to the (t,w)-fixed-weight repetition of a $(k_1,\ldots,k_\mu)$-special-sound $(2\mu+1)$-round interactive proof. We have obtained a strong result on the knowledge soundness of the fixed-weight optimization, proving that the knowledge error of the protocol matches the maximum cheating probability of a dishonest prover. To the best of our knowledge, this is the first time the security of this standard optimization has been analyzed, beyond 2-special-sound Sigma protocols.

Our work gives direct, tight results on the security of the interactive proofs underlying many recent signatures. For instance, they provide an explicit knowledge error for the fixed-weight repetition of q2-identification schemes [Che+16], such as the 5-round interactive proof underlying CROSS [Bal+23b]. Similarly, they can be applied to k-special-sound Sigma protocols, with $k>2$, such as the recent SIDH-based signature of [GPV24].

Future Works. When dealing with multi-round interactive proofs, our results cover the fixed-weight optimization in the case where a fixed challenge in the last round is repeated a prescribed number of times. This is a seemingly arbitrary choice, as we might consider fixed-weight challenges in intermediate rounds or in a generic subset of them, but it is closely tailored to concrete applications of interactive proofs for building digital signatures. Indeed, the fixed-weight optimization is motivated by the presence of challenges with larger response sizes, while intermediate rounds have typically constant-size responses. However, an extension of our results to a "generalized" fixed-weight optimization might be of interest for future protocols with different approaches from the current ones.

On the negative side, our result does not directly translate to the non-interactive case and thus we cannot use the bound of Subsections 4.2.2 and 4.2.3 to determine parameters of signature schemes. Indeed, when considering signature schemes, it is necessary to also take into consideration the security loss caused by the Fiat-Shamir Transform. As shown in [AFK22b], the Fiat-Shamir transform of a t-fold parallel repetition of a $(k_1, \ldots, k_\mu)$-special-sound interactive proof incurs in a security loss that is exponential in the number of rounds. While the attack of [AFK22b] does not specifically target fixed-weight repetitions, the same heuristic could be applied to our scenario. Adapting the attack to find precise bounds for the security loss is an interesting open problem for future research, since they are crucial to determine provable-secure parameters of multi-round-based signatures, such as CROSS [Bal+23b].

4.3 Straight-Line Extraction from Group Action

In this section, we focus on Sigma protocols. We have already seen that it is possible to turn them in non-interactive proof systems using general transforms (or compilers), of which the most noteworthy is the one proposed by Fiat and Shamir [FS87a; PS96; Abd+02], but other ways are also possible, like the ones proposed by Pass [Pas03], Fischlin [Fis05; KS22; CL24] and Unruh [Unr15]. Each compiler for Sigma protocols describes how to build a NIZKP, and how it is possible to define an algorithm, called extractor, that interacting with a prover can extract a valid witness for the statement in input to the prover and the verifier. The transformation introduced by Fiat and Shamir [FS87a; PS96; Abd+02] is proven secure in [PS96] describing an extractor with the capability to rewind the prover algorithm to reconstruct a witness for the public statement. This capability is not required by the extractors described by Pass, Fischlin, and Unruh. For this reason the NIZKP obtained using the latter transforms meet the definition of online (or straight line) extractable NIZKP, since they are associated to extractors who do not need to rewind the prover to extract a witness.

Despite Fiat-Shamir [FS87a; PS96; Abd+02] being the most common and practical way to compile interactive proofs in non-interactive proofs, the fact that the extraction of the witness (needed to prove the proof is knowledge sound) requires rewinding [PS96] limits its applicability in contexts where rewinding is not an option, for example while executing some multiparty protocols that require the parties to create NIZKP of knowledge of witnesses. For those applications, the security proof requires the use of NIZKP with online extractors (Definition 4.3.3). Another downside of using Fiat-Shamir, which is mostly omitted when it comes the time to instantiate the protocols, is the loss of tightness in the security proof which does not occur when using online extractable NIZKP. We analyze the special case constituted by the online extractable NIZKP obtained from group actions, asking ourselves the following question:

> *Is it possible to improve the standard techniques to turn a Sigma protocol from cryptographic group actions into an online-extractable NIZKP?*

In the following, we propose a new online extractable transform which improves, for some relevant set of parameters, the results obtained by using Fischlin, Pass or Unruh.

4.3.1 Sigma Protocols and Online-Extractable Transforms

We have already had the opportunity to discuss interactive protocols in their most general sense. In this section, given a binary relation $\mathcal{R} \subseteq X \times Y$, we will focus on Sigma protocols

which are complete, 2-special sound and Honest-Verifier Zero-Knowledge. With a little abuse of notation, we will simply call these objects "Sigma protocols", always referring to the following definition.

Definition 4.3.1 (Sigma Protocol [Dam02])**.** Let $\mathcal{R} \subseteq X \times Y$ be a binary relation. A *Sigma protocol* Σ for $\mathcal{R}$ is a pair $\Sigma = (\mathsf{P}, \mathsf{V})$ where P is an algorithm taking as input $(x, y) \in \mathcal{R}$, and V is an algorithm taking as input a statement $x \in X$. The interaction between P and V is structured as follows: P computes a message f, called *first message*, and sends it to V. V samples a random challenge from a finite challenge space $\mathsf{ch} \xleftarrow{\$} \mathsf{Ch}$, and sends ch to $\mathsf{P}(x, y)$. Finally, P computes a response r and sends it to V who outputs `Accept` or `Reject` as a deterministic function of $x, (f, \mathsf{ch}, r)$. We require that this protocol satisfies the *completeness*, *2-special soundness* and the *honest-verifier zero-knowledge (HVZK)* properties, as defined below:

1. *Completeness:* We say that $\Sigma = (\mathsf{P}, \mathsf{V})$ is complete if for all $(x, y) \in \mathcal{R}$, when P and V interact with each other following the protocol, V always outputs `Accept`.

2. *2-Special Soundness:* We say that $\Sigma = (\mathsf{P}, \mathsf{V})$ is 2-special sound if there is an efficient deterministic algorithm Ext (called a witness extractor) with the following property: given as input a statement $x \in X$, along with two accepting conversations (f, ch, r) and (f, ch', r') for x, with $\mathsf{ch} \neq \mathsf{ch}'$, Ext always outputs $y \in Y$ such that $(x, y) \in \mathcal{R}$ (i.e., y is a witness for x).

3. *Honest-Verifier Zero-Knowledge:* We say that $\Sigma = (\mathsf{P}, \mathsf{V})$ is HVZK if there exists a probabilistic polynomial time (PPT) algorithm Sim (called a simulator) that takes as input $(x, \mathsf{ch}) \in X \times \mathsf{Ch}$ and outputs a pair (f, r) such that (f, ch, r) is an accepting conversation for x with the same distribution as conversations between honest P and V with output x.

When dealing with Sigma protocols and the Fiat-Shamir transform [Sch91; CL04; Bia+20; Tan+22; DFG19; Ste93], in order to achieve some notion of security, the interactive protocol is required to satisfy some additional properties. We provide a description of the required properties in the following and call *effective* a Sigma protocol that satisfies these requirements.

Definition 4.3.2 (Effective Sigma protocol)**.** We say that a Sigma protocol is effective if it also satisfies the following properties.

1. *First Message Entropy:* Being λ the security parameter, the min-entropy of $f \xleftarrow{\$} \mathsf{P}(x, y)$ is super-logarithmic in λ.

2. *Public Coin:* For any λ, for any $(x, y) \in \mathcal{R}$, and for any $f \xleftarrow{\$} \mathsf{P}(x, y)$, $\mathsf{ch} \xleftarrow{\$} \mathsf{Ch}$ is uniform in $\{0, 1\}^{\ell(\lambda)}$.

3. *Unique Responses:* For any PPT algorithm A taking as input the security parameter λ, A produces a tuple $(x, f, \mathsf{ch}, r, r') \xleftarrow{\$} A(\lambda)$ satisfying $\mathsf{V}(x, f, \mathsf{ch}, r) = \mathsf{V}(x, f, \mathsf{ch}, r') =$ `Accept` with no more than negligible probability in λ.

Parallel composition of Sigma protocols. Given an effective Sigma protocol Σ, it is possible to define another effective Sigma protocol Σ^ℓ, given by ℓ parallel repetitions of Σ, as follows: P computes a first message $f = (f_1, \ldots, f_\ell)$ of Σ^ℓ, where f_i is a first message

of Σ and sends it to V. V samples a random challenge $\mathsf{ch} = (\mathsf{ch}_1, \ldots, \mathsf{ch}_\ell)$ of Σ^ℓ, given by the concatenation of ℓ challenges ch_i of Σ, and sends ch to $\mathsf{P}(x, y)$. Finally, P computes a response $r = (r_1, \ldots, r_\ell)$ given by the concatenation of the responses of Σ to the challenge ch_i and the first message f_i for $i \in [\ell]$, and sends it to V who outputs `Accept` or Reject as a deterministic function of $x, (f, \mathsf{ch}, r)$. If Σ is a Sigma protocol, Σ^ℓ is also a Sigma protocol [Dam02, Lemma 1]. It is easy to see that if the properties required in the definition of an effective Sigma protocol are satisfied by Σ, then the same properties are inherited by Σ^ℓ, therefore also Σ^ℓ is an effective Sigma protocol.

We define the concept of non-interactive zero-knowledge proof (NIZKP) of knowledge for relation $\mathcal{R}$ with online (or straight-line) extractors in the random oracle model.

Definition 4.3.3 (NIZKP of Knowledge with online extractor [Fis05])**.** A NIZKP of knowledge for relation $\mathcal{R}$ with an online extractor is a pair of PPT algorithms (P, V) satisfying the following properties:

1. *Completeness*: for any oracle H, any $(x, y) \in \mathcal{R}$, for any $\pi \xleftarrow{\$} \mathsf{P}^H(x, y)$, we have $\mathsf{V}^H(y, \pi) = \texttt{Accept}$ with overwhelming probability.

2. *Zero-Knowledge*: there exist a pair of probabilistic polynomial time (PPT) algorithms $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ (called simulator) such that, for any pair of PPT algorithms $D = (D_0, D_1)$ (called distinguisher) the following experiments are indistinguishable:

   - let H be a random oracle, $(x, y, \delta) \xleftarrow{\$} D_0^H(\lambda)$ and $\pi \xleftarrow{\$} \mathsf{P}^H(x, y)$ if $(x, y) \in \mathcal{R}$, otherwise $\pi \leftarrow \perp$. Output $D_1^H(\pi, \delta)$;
   - let $(H_0, \sigma) \xleftarrow{\$} \mathsf{Sim}_0(\lambda)$, $(x, y, \delta) \xleftarrow{\$} D_0^{H_0}(\lambda)$, and $(H_1, \pi) \xleftarrow{\$} \mathsf{Sim}_1(\sigma, x, 1)$ if $(x, y) \in \mathcal{R}$, and $(H_1, \pi) \xleftarrow{\$} \mathsf{Sim}_1(\sigma, x, 0)$ otherwise. Output $D_1^{H_1}(\pi, \delta)$.

   where δ is the state information which is passed between D_0 and D_1 and σ the state information passed between Sim_0 and Sim_1.

3. *Online Extractor*: There exist a PPT algorithm Ext (called online extractor) such that the following holds for any algorithm A. Let H be a random oracle, $(x, \pi) \xleftarrow{\$} A^H(\lambda)$, and $Q_H(A)$ be the set of queries made by A to H, and H's answers. Denote with $y \xleftarrow{\$} \mathsf{Ext}(x, \pi, Q_H(A))$, then the probability that $\mathsf{V}^H(x, \pi) = 1$ and $(x, y) \notin R$ is negligible in λ.

Note that Sim in the second experiment of the zero-knowledge property programs the random oracle giving access to the random oracle H_0 to D who can send queries to H_0 before it generates the statement-witness pair $(x, y) \in \mathcal{R}$ and sends it to Sim. Then, Sim can program the random oracle according to the statement x while keeping it consistent with H_0 because D_1, who has received the state information δ from D_0, will be allowed to perform random oracle queries to H_1 before producing its output.

In some NIZKP (see for example [CL24]) the probabilistic algorithm P taking in input a statement-witness pair in $\mathcal{R}$ does not produce a valid proof π with probability 1, in this case P can be modified so that if the proof is not valid (i.e. $\pi = \perp$), P starts over using a different randomness until it produces a valid proof $\pi \neq \perp$. of course this modification allows to reduce the failure probability to a negligible value but, on the other hand, the proof generation time grows with the number of times the prover has to start from the beginning. To study these cases, it is useful to define the concept of completeness error.

Definition 4.3.4 (Completeness Error)**.** We define the completeness error ϵ_c^{P} of a NIZKP of knowledge (P, V) for the relation $\mathcal{R}$ as the probability that a prover, taking as input a statement-witness pair $(x, y) \in \mathcal{R}$, generates an invalid proof $\pi = \perp$ for x

$$\epsilon_c^{\mathsf{P}} = \Pr\Big[\perp \xleftarrow{\$} \mathsf{P}(x, y) | (x, y) \in \mathcal{R}\Big].$$

We also define the concept of soundness error associated to an online extractor as the probability that the algorithm A, who is given in input a statement-witness pair $(x, y) \in \mathcal{R}$, can output a valid proof π for x, but the extractor Ext, who is given access to $\pi, Q_H(A)$ can not extract a witness for the statement x.

Definition 4.3.5 (Soundness Error w.r.t. an Online Extractor)**.** We define the soundness error $\epsilon_s^{\mathsf{P},\mathsf{Ext}}$ of a NIZKP (P, V) associated to an online extractor Ext as the probability that an algorithm A can generate a valid proof π for a statement x of its choice performing the set of random oracle queries $Q_H(A)$ without having $\mathsf{Ext}(x, \pi, Q_H(A))$ being able to extract a witness for x.

$$\epsilon_s^{\mathsf{P},\mathsf{Ext}} = \Pr\Big[(x, \mathsf{Ext}(x, \pi, Q_H(A))) \notin R \mid (x, \pi, Q_H(A)) \xleftarrow{\$} A^H(\lambda) \wedge \mathsf{V}^H(x, \pi) = 1\Big]$$

To prove the soundness of a NIZKP it is crucial to design a PPT algorithm Ext which is able to extract the witness with overwhelming probability, because this implies that the algorithm A who has generated the proof actually knows the witness for x (or given the queries it has made, it would be able to reconstruct a witness for x). In general we want that the soundness error for a NIZKP is below $2^{-\lambda}$. Below we describe some noteworthy techniques used to generate online-extractable NIZKP from Sigma protocols.

Fischlin Transform

We recap the transform proposed by Fischlin in [Fis05] and recently revisited by Chen and Lindell in [CL24] where the authors simplify the description of the transform reducing the number of parameters used to define it, also simplifying the relations among the parameters that must be satisfied in order to obtain secure NIZKPs. The Fischlin transform, in its simplified version described in [CL24], instructs the prover taking in input a witness-statement pair $(x, y) \in \mathcal{R}$ and perform the following steps:

- generate ρ first messages $(f_1, \ldots, f_\rho)$ and set $\overline{f} := (f_1, \ldots, f_\rho)$
- $\forall i \in [\rho]$ search for $\mathsf{ch}_i \in \mathsf{Ch}$ such that:
  - $(f_i, \mathsf{ch}_i, r_i)$ is an accepting transcript;
  - $H_b(x, \overline{f}, i, \mathsf{ch}_i, r_i) = 0^b$.
- output as a proof for x: $\pi \leftarrow (\overline{f}, \{\mathsf{ch}_i\}_{i\in\rho}, \{r_i\}_{i\in\rho})$.

To verify a proof π for x, the verifier checks that $\forall i \in [\rho]$ the transcripts $(f_i, \mathsf{ch}_i, r_i)$ are valid for x, and that $H_b(\overline{f}, \mathsf{ch}_i, r_i) = 0^b$.

Note that the challenge space Ch must be sufficiently large to allow the prover to eventually find a challenge for each first message. If $|\mathsf{Ch}|$ is too small, it can be easily enlarged by executing in parallel several instances of the Sigma protocol until the challenge space becomes sufficiently large [Dam02, Lemma 2]. Being $(\mathsf{P}_{\mathtt{Fi}}, \mathsf{V}_{\mathtt{Fi}})$ the NIZKP obtained

by applying the Fischlin transform to an effective Sigma protocol $\Sigma = (\mathsf{P}, \mathsf{V})$. In [CL24] the authors describe that, in order to guarantee a completeness error $\epsilon_c^{\mathsf{P_{Fi}}} \leq 2^{-40}$, the number of challenges that the prover might need to try is

$$|\mathsf{Ch}| = 2^{b+5} \text{ if } \rho < 64 \quad \text{and} \quad |\mathsf{Ch}| = 2^{b+6} \text{ otherwise.} \tag{4.12}$$

Similarly, in order to guarantee a soundness error $\epsilon_s^{\mathsf{P_{Fi}},\mathsf{Ext_{Fi}}} = 2^{\lambda}$ we need

$$b\rho = \lambda; \tag{4.13}$$

where $\mathsf{Ext_{Fi}}$ is the online extractor used to prove the knowledge soundness of the NIZKP obtained using the Fischlin compiler [Fis05; CL24].

Pass Transform

Given a Sigma protocol Σ which is 2-special sound and honest-verifier zero-knowledge, the transform proposed by Pass in [Pas03] first consists in deriving another Sigma protocol Σ' described as follows:

- the prover generates:
  1. a first message f according to Σ;
  2. 2 distinct challenges $\mathsf{ch}_0, \mathsf{ch}_1 \in \mathcal{C}$, the challenge space of Σ;
  3. r_0, r_1, the responses corresponding to the challenges $\mathsf{ch}_0, \mathsf{ch}_1$ w.r.t. f and computes two commitments C_0, C_1 to respectively r_1 and r_2;

  and sends $(f, \mathsf{ch}_0, \mathsf{ch}_1, C_0, C_1)$ to the verifier;
- the verifier generates a random bit $b \in \{0, 1\}$;
- the prover sends the opening r_b of the commitment C_b to the verifier;
- the verifier checks that
  1. the transcript (f, ch_b, r_b) is a valid transcript of Σ;
  2. C_b actually opens to r_b.

Now it is possible to derive a non-interactive proof by applying the Fiat-Shamir transform to λ parallel executions of Σ' obtaining the following non-interactive proof:

- the prover performs the following operations:
  1. generates λ first messages of Σ' and $\mathbf{f}' \leftarrow (f'_1, \ldots, f'_\lambda)$;
  2. for every $i \in \{1, \ldots, \lambda\}$, generate two challenges $\mathsf{ch}_{(i,0)}, \mathsf{ch}_{(i,1)}$, where $\mathsf{ch}_{(i,0)} \neq \mathsf{ch}_{(i,1)}$. Define $\mathbf{b}' := ((\mathsf{ch}_{(i,0)}, \mathsf{ch}_{(i,1)}))_{i \in \{1,\ldots,\lambda\}}$;
  3. for every $i \in \{1, \ldots, \lambda\}$, compute the responses $r_{(i,0)}, r_{(i,1)}$ corresponding to the challenges $\mathsf{ch}_{(i,0)}, \mathsf{ch}_{(i,1)}$ w.r.t. f_i, and computes two commitments $C_{(i,0)}, C_{(i,1)}$ to r_1 and r_2, respectively. Define $\mathbf{c}' := (C_{(i,0)}, C_{(i,1)})_{i \in \{1,\lambda\}}$;
  4. generates the challenge $\mathsf{ch} = H(\mathbf{f}', \mathbf{b}', \mathbf{c}', m) \in \{0, 1\}^\lambda$, where m is the message to sign;
  5. computes the responses $\mathbf{r}' \leftarrow (r'_1, \ldots, r'_\lambda)$ of Σ' where for each $i \in [\lambda]$ the challenge corresponding to f'_i is ch_i.

the resulting proof of m is $\sigma \leftarrow (\mathbf{f}', \mathbf{b}', \mathbf{c}', \mathbf{r}')$.

- the verifier computes $\mathsf{ch} = H(\mathbf{f}', \mathbf{b}', \mathbf{c}', m)$, checks that the transcripts associated to $(\mathbf{f}', \mathsf{ch}, \mathbf{r}')$ are λ valid transcripts, and checks that $\mathbf{c}'_{(i,\mathsf{ch}_i)}$ opens to $\mathbf{r}'_i$.

For what concerns the completeness of the proof, note that a honest prover can always produce a valid proof because it generates valid commitments to the responses of the Sigma protocol Σ. For what concerns the soundness of the digital proof scheme, the way the extractor can learn the witness is from the special soundness of the underlying Sigma protocol and the random oracle queries providing information about two valid transcripts for the same first message and two distinct challenges. A prover must always be able to open the commitment C_{i,ch_i} (which is part of f'_i) associated to the challenge ch_i. To do so, it must have sent a random oracle query for $(r_{i,\mathsf{ch}_i}, s_{i,\mathsf{ch}_i})$ where r_{i,ch_i} is the right response associated to the first message f_i and challenge ch_i.

This means that the only way a prover can create a valid proof *without* giving to the random oracle two transcripts for the same first message f_i and distinct challenges is that the prover guesses in advance the λ bit challenge ch and only computes the commitments associated to the responses it has guessed.

Unruh Transform

The transform proposed by Pass [Pas03] has been generalized by Unruh [Unr15] to be secure against polynomial-time quantum adversaries. More specifically, when proving security in both Pass and Fischlin transforms, we design an extractor which can extract a valid witness by (1) rewinding the adversary and (2) accessing the input queries to the random oracle. In the quantum random oracle setting this is a problem because rewinding is difficult, as the input queries can not be measured without compromising the state, and a key step in these security proofs is the ability to reprogram the random oracle to an input that has already been asked for by the adversary. In the classical ROM this is not a problem, but in the QROM the concept of "input that has already been requested by the opponent" is not well defined. This is because the input queries could be in superposition, and we could not measure the input without altering the state of the qbits. The crucial issue is therefore to build an oracle for which we can observe the input queries made by the adversary, having access only to the outputs that the adversary sends us. Unruh's proposed idea is to use a random oracle G that is also a permutation. Clearly, for each oracle input x that we want to know, we must have access to $G(x)$, and therefore we will have to include $G(x)$ in the output of the protocol.

Informally, given a Sigma protocol Σ which is 2-special sound and honest-verifier zero-knowledge, the transform proposed by Unruh in [Unr15] works as follows. The prover is instructed to generate $m = 2^\ell$ different proofs for Σ with the same first message. The prover computes the responses and commits to the transcripts building a Merkle tree. This basic unit is repeated $r = \frac{\lambda}{\ell}$ times to amplify the soundness to a security level λ. Upon giving input to a hash function the r Merkle roots, the prover receives a challenge that specifies for each of the r basic units a transcript that must be revealed. So, the prover opens the transcripts corresponding to the specified challenge for each first message, and the verifier checks that the opening of the commitments are correct and that the transcripts are valid. Intuitively, a prover who is able to open the commitment showing the transcript must have committed to two transcripts for the same first message and different challenge with probability $\left(2^{-\ell}\right)^r$, from which the parameters condition $\ell \cdot r \geq \lambda$.

- the prover performs the following operations:
  1. generates $t \cdot m$ proofs $(f_i, \mathsf{ch}_{i,j}, r_{i,j})$ for Σ;
  2. for every $i \in \{1, \dots, t\}$, for every $j \in \{1, \dots, m\}$, uses the permutation function and compute the commitments $h_{i,j} \leftarrow G(r_{i,j})$.
  3. generates the challenge $J_1 || \dots || J_t = H(x, (f_i)_i, (\mathsf{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j})$;

  the resulting proof of m is $(x, (f_i)_i, (\mathsf{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j}, (r_{i,J_i})_i)$.
- the verifier computes $J_1 || \dots || J_t = H(x, (f_i)_i, (\mathsf{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j},)$, checks that, $\forall i$, $\mathsf{ch}_{i,1}, \dots, \mathsf{ch}_{i,m}$ are pairwise distinct, the transcripts associated to $(x, (f_i)_i, (\mathsf{ch}_{i,j})_{i,J_i}, r_i)$ are t valid transcripts, and checks that $h_{i,J-i} = G(r_i)$.

A malicious prover who succeeds in this will have to include valid responses in at least a large fraction of the $G(r_{i,j})$. Thus by inverting G, we can find two valid triples (f, ch, r) and (f, ch_0, r_0) if the malicious prover's proof passes verification. In [KS22, Section 1.1] the authors observe that only the opening of the commitments incurs in a $\log(2^\ell) \cdot r \cdot \lambda = \lambda^2$ bits (i.e. approximately 2 KBytes for $\lambda = 128$) of overhead in the size of the proof π.

4.3.2 Cryptographic Group Actions

Definition 4.3.6 (Group Action)**.** A group $(G, \cdot, e)$ is said to act on a set X if there exists a map

$$\begin{aligned} \star \colon G \times X &\longrightarrow X \\ (g, x) &\longmapsto g \star x \end{aligned}$$

such that $e \star x = x$ for every $x \in X$ and $g \star (h \star x) = (g \cdot h) \star x$ for every $x \in X$ and all $g, h \in G$. In this case, we say that the triple $(G, X, \star)$ is a *group action*.

In order to build cryptographic schemes using group actions, we need instances of group actions $(G, X, \star)$ for which it is possible to efficiently perform some operations such as (1) compute the group operation and the inverses in the group, (2) perform random sampling in X and G, (3) decide the equality and validity of a representation of the elements in X and (4) compute the action $\star$ of a group element over a set element. This kind of group actions are called *effective group actions* [Ala+20].

For a group action to be useful in cryptography, it must not only be effective but also be based on one-way functions, or equivalently hard problems, for which the prover is the only one who knows the associated solution. An effective group action that satisfies this further requirement is called *cryptographic group action*. A (typically hard) family of problems for effective group actions is the Group Action Inversion Problem (GAIP) defined below.

Definition 4.3.7 (GAIP)**.** The GAIP on a group action $(G, X, \star)$ is defined as follows. Given as input a pair (x_0, x_1) of elements in X, output a group element $g \in G$ such that $x_1 = g \star x_0$.

Another well-known problem is the Stabilizer Computation Problem (SCP) defined below.

Definition 4.3.8 (SCP)**.** The SCP over a group action $(G, X, \star)$ is defined as follows. On input $x_0 \in X$, find an element $g \in G, g \neq e$ such that $g \star x_0 = x_0$.

Examples of effective group actions for which these assumptions are believed to hold are based on isogenies [Cas+18], tensors [Ji+19; Tan+22] and linear codes [Cho+23a; Bia+20]. The Sigma protocol $\Sigma_\star = (\mathsf{P}, \mathsf{V})$ for the relation $\mathcal{R} \subseteq \{((x_0, x_1), g) \in G \times X^2 \mid x_1 = g \star x_0)\}$ is defined in Protocol 4.2.

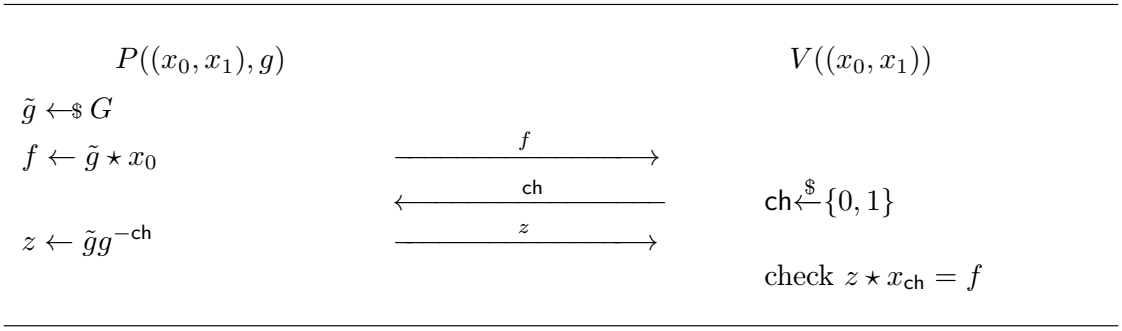

Figure 4.2: Sigma Protocol Based on Group Action $\Sigma_\star$.

It can be easily proved that the Sigma protocol satisfies the completeness, 2-special soundness and (special) honest verifier zero knowledge properties defined in Definition 4.3.1. The simulator Sim, used to prove the HVZK property, takes in input a statement (x_0, x_1) and a challenge $\mathsf{ch} \in \{0,1\}$ samples uniformly at random a group element $z \in G$, computes the first message $f = z \star x_{\mathsf{ch}}$ and returns the transcript (f, ch, z). The protocol is 2-special sound since, given two transcripts $(f, 0, z_0)$ and $(f, 1, z_1)$, the extractor can exhibits $z_1^{-1} z_0$ as a witness for (x_0, x_1). A single interaction of this basic protocol is represented in Figure 4.3.

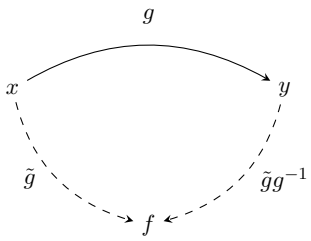

Figure 4.3: Group action graph.

Remark 4.3.9. When we instantiate the Sigma protocol described in Protocol 4.2 using a group action for which SCP is hard, then the Sigma protocol has quasi unique responses, because given a first message f, and a challenge ch, the prover who knows the witness g can compute a single valid response. In this case, since the Sigma protocol has unique responses (as per Definition 4.3.2), it is public coin, and if instantiated using a group action $(G, X, \star)$ with sufficiently large group G and set X, then also the min-entropy of the first message of the Sigma protocol is super-logarithmic in λ, then $\Sigma_\star$ is an effective Sigma protocol according to Definition 4.3.2.

The Sigma protocol $\Sigma_\star$ described in protocol 4.2 has a binary challenge space $\mathsf{Ch} = \{0,1\}$. The challenge space can be increased by considering the parallel composition of ℓ instances of $\Sigma_\star$ that defines a Sigma protocol $\Sigma_\star^\ell$ that has, as its challenge space, $\mathsf{Ch} = \{0,1\}^\ell$, whose size is 2^ℓ.

Design Principles. When designing NIZKP starting from the Sigma protocol from cryptographic group actions, one should take into consideration *at least* the following two principles, which we state informally. The first regards the computational costs of computing the group action operation.

Principle 1. Computing the action of a group element on a set element is computationally expensive. We want to keep this number low to reduce the creation or verification time of NIZKP.

The second principle is a consequence of the use of the following standard technique, at least when describing schemes based on cryptographic group actions. Since responses to challenge 0 are random group elements, these values can be derived starting from λ-bits long seeds [Bor+23]. The size of seeds in general is much smaller than the size of the group elements; therefore, the transcripts with challenge 0 are much lighter than the transcripts with challenge 1.

Principle 2. In the NIZKP generation process, we want to keep the weight of the challenges low to reduce the size of the proof.

Remark 4.3.10. Since we focus on the application of straight line transforms to $\Sigma_\star$, it is reasonable to identify a target weight that we want the challenges in our proof to match. For example, if we consider the group action used in LESS [Bia+20], the group elements have a size of 237 bytes for the security parameter $\lambda = 128$. This means that a proof associated with a challenge of weight in the range of 16 and 43 results in a weight *greater* of 3.7 KBytes and 10.1 KBytes, respectively.

4.3.3 A Group Action Oriented Transform

We propose a transform for the creation of NIZKP which satisfies the online-extractability property and is fixed-weight by design, meaning that the weight of the challenge is a parameter of the transform and is not probabilistic, as it happens for Fischlin, Unruh and Pass transforms. In more detail, in the following we first propose a transform that has completeness error 0. In the next subsection, we will exploit this initial structure to outline a new transform, balancing between the completeness error and the size of the proof. We will see that, for some relevant security parameters, this transformation improves the performances obtained by applying the other ones, yielding lighter challenges for the same query complexity and the same number of group action computations.

General Construction

The bigger picture of our construction is as follows. Suppose we are given a 2-special sound Sigma protocol and a true statement $(x, y) \in \mathcal{R}$. Given L first messages $f_1, \ldots, f_L$ for the basic Sigma protocol, denote with $\overline{f} = (f_1, \ldots, f_L)$ and let the prover search through challenges ch_i and responses z_i to find ρ tuples $(x, \overline{f}, \mathsf{ch}_i = 1, z_i)$ whose b least significant bits of the hash collide, for a small b. For the sake of simplicity, in the following we will assume that H only has b output bits. The prover outputs the vector $(f_i, \mathsf{ch}_i, z_i)_{i \in L}$. No further hash values need to be included, and the verifier checks that all executions are valid and, for the ρ executions for which $\mathsf{ch}_i = 1$, their hash collide. The parameters will be set up in such a way that the honest prover is able to find a valid proof efficiently. Informally, a honest prover is able to carry out a proof without querying the random oracle twice on the same first message only if it guesses in advance the a ρ-collision for a given hash function. Otherwise, the knowledge extractor can find them in the list of hash queries and compute the witness. The details are as follows.

We make use of effective Sigma protocols with logarithmic challenge length ℓ. Given $d \in \{0,1\}^b$, let $\mathcal{T}(d)$ be the set of *d-target components* associated to the digest d. The sets $\mathcal{T}(d)$ are initialized to the empty set. The transform can be described as follows:

Construction 1. Let $\Sigma = (\mathsf{P}, \mathsf{V})$ be an effective Sigma protocol for a relation $\mathcal{R}$. We define a non-interactive proof system $(\mathsf{P}_\blacktriangle, \mathsf{V}_\blacktriangle)$ for relation $\mathcal{R}$ in the random oracle model which is defined by the following parameters. Let L be the number of "parallel repetitions" conducted, τ the weight of the challenge included in the proof, and b the output size in bits of the random oracle. Then the algorithms $(\mathsf{P}_\blacktriangle, \mathsf{V}_\blacktriangle)$ are defined as follows.

- *Prover.*
  1. The Prover ${\mathsf{P}_\blacktriangle}^H(x, y)$ runs the prover $\mathsf{P}_\blacktriangle(x, y)$ in L independent repetitions to obtain L first messages $f_1, \ldots, f_L$ and sets $\overline{f} = (f_1, \ldots, f_L)$;
  2. for each $i \in [L]$, P computes the response z_i associated to the first message f_i and the challenge 1;
  3. Given $i \in [L]$ and $d := H(x, \overline{f}, i, 1, z_i) \in \{0, \ldots, 2^b - 1\}$, add i to $\mathcal{T}(d)$;
  4. Being $D = \{d : |\mathcal{T}(d)| \geq \tau\}$
     - if $D = \emptyset$, then P outputs $\perp$.
     - Otherwise, sample $\bar{d}$ from D and sample a subset $\mathcal{T}'(\bar{d})$ of $\mathcal{T}(\bar{d})$ with $|\mathcal{T}'(\bar{d})| = \tau$. P generates $\mathsf{ch} \in \{0,1\}^L$ setting, $\forall i \notin \mathcal{T}'(\bar{d})$ $\mathsf{ch}_i = 0\}$, and $\mathsf{ch}_i = 1$ otherwise. Finally, for every $i \in [L]$, the prover computes the response z_i associated to f_i and ch_i, and outputs $\pi = (f_i, \mathsf{ch}_i, z_i)_{i \in [L]}$ [3].
- *Verifier.* The verifier ${\mathsf{V}_\blacktriangle}^H(x)$ on input $\pi = (f_i, \mathsf{ch}_i, z_i)_{i \in [L]}$ accepts if and only if the following three conditions hold:
  1. $\mathsf{V}(x, f_i, \mathsf{ch}_i, z_i) = 1$ for every $i \in [L]$;
  2. $\mathsf{ch}_i = 1$ on exactly τ indexes $i_1, \ldots, i_\tau$, otherwise $\mathsf{ch}_i = 0$;
  3. $H(\overline{f}, i_1, 1, z_{i_1}) = \cdots = H(\overline{f}, i_\tau, 1, z_{i_\tau})$.

We recall that the responses to challenge 0 are lighter because they can be encoded in seeds, according to the technique we have introduced before the description of Principle 2.

Security Analysis

In the following we prove that Construction 1 is a NIZKP with online extractor as defined in Definition 4.3.3.

Theorem 4.3.11. *Let (P, V) be an effective Sigma protocol for relation $\mathcal{R}$. Define $(\mathsf{P}_\blacktriangle, \mathsf{V}_\blacktriangle)$ as the non-interactive proof system obtained by applying Construction 1 to (P, V). Assuming $b(\tau - 1) = \lambda$ and $L = 2^b(\tau - 1) + 1$, then $(\mathsf{P}_\blacktriangle, \mathsf{V}_\blacktriangle)$ is a non-interactive zero-knowledge proof of knowledge for relation $\mathcal{R}$ (in the random oracle model) with an online extractor, completeness error $\epsilon_c^{\mathsf{P}_\blacktriangle} = 0$, and soundness error $\epsilon_s^{\mathsf{P}_\blacktriangle, \mathsf{Ext}_\blacktriangle} = 2^{-\lambda}$.*

Proof. We prove that Construction 1 satisfies the completeness, zero-knowledge and knowledge extraction properties.

Completeness. Recall that an honest prover is able to produce a valid proof when he finds τ indices $i_1, \ldots, i_\tau$ such that $H(\overline{f}, i_1, 1, z_{i_1}) = \cdots = H(\overline{f}, i_\tau, 1, z_{i_\tau})$. We observe that it is possible to obtain completeness 1 simply by setting $L = 2^b(\tau - 1) + 1$. In this case there always exists a digest $\bar{d}$ such that $|\mathcal{T}(\bar{d})| \geq \tau$ by a straightforward variant of the pigeonhole principle. This means that for such a value of L the $\epsilon_c^{\mathsf{P}} = 0$.

[3] Or just $\pi = (\mathsf{ch}_i, z_i)_{i \in [L]}$ if the Sigma protocol is first message retrievable.

Zero-Knowledge. A possible zero-knowledge simulator $\mathsf{Sim}_{\blacktriangle} = (\mathsf{Sim}_{\blacktriangle 0}, \mathsf{Sim}_{\blacktriangle 1})$ works as follows. $\mathsf{Sim}_{\blacktriangle 0}$ defines H_0 as an empty hash table which it programs to simulate a random oracle. $\mathsf{Sim}_{\blacktriangle}$ gives access to it to $D_0^{H_0}$, then, $D_0^{H_0}$ can send $\mathsf{Sim}_{\blacktriangle 0}$ random oracle queries for input q. $\mathsf{Sim}_{\blacktriangle}$ samples uniformly at random $d(q) \xleftarrow{\$} \{0,1\}^b$ and adds to the hash table the pair $(q, d(q))$. D_0 in turn sends to $\mathsf{Sim}_{\blacktriangle}$ a triple (x, w, δ). When $(x, y) \notin R$, the simulation is immediate, and $\mathsf{Sim}_{\blacktriangle 1}$ outputs $(\perp, H_1)$.

Otherwise, if $(x, y) \in \mathcal{R}$, for every $d \in \{0,1\}^b$, $\mathsf{Sim}_{\blacktriangle 1}$ defines $\mathcal{T}(d) = \emptyset$, after which for every $i \in [L]$, it randomly samples a string $s_i \in \{0,1\}^b$, and adds i to $\mathcal{T}(s_i)$. Since $L = 2^b(\tau - 1) + 1$, $D = \{d \in \{0,1\}^L : |\mathcal{T}(d)| \geq \tau\} \neq \emptyset$. $\mathsf{Sim}_{\blacktriangle 1}$ samples $\overline{d} \in D$, and randomly selects a subset $\mathcal{T}'(\overline{d}) \subseteq \mathcal{T}(\overline{d})$ of cardinality τ. $\mathsf{Sim}_{\blacktriangle 1}$ creates the challenge vector ch setting $\mathsf{ch}_i = 1 \quad \forall i \in \mathcal{T}'(\overline{d})$ and $\mathsf{ch}_i = 0$ otherwise. Finally it runs the special zero-knowledge simulator Sim of the underlying effective Sigma protocol L times on x and each ch_i to obtain L tuples $(f_i, \mathsf{ch}_i, z_i)$ such that $V(x, f_i, \mathsf{ch}_i, z_i) = 1$. $\mathsf{Sim}_{\blacktriangle 1}$ programs the hash table H_1 as H_0 to which are added the pairs $((\overline{f}, i, \mathsf{ch}_i, z_i), s_i)$, i.e. $H_1(\overline{f}, i, \mathsf{ch}_i, z_i) = s_i$. Finally, $\mathsf{Sim}_{\blacktriangle 1}$ outputs $(\pi, H_1) = \big(\{(f_i, \mathsf{ch}_i, z_i)\}_{i \in [L]}, H_1\big)$.

$\mathsf{Sim}_{\blacktriangle}$ produces a proof π which is indistinguishable from a real proof because the output of the hash function H_1 are sampled uniformly at random. From these, the components for which $\mathsf{ch}_i = 1$ are selected as in a real protocol execution and then, given the values $\mathsf{ch}_i, \forall i \in [L]$, also the simulation of the transcript is correct because we generate it using the simulator $\mathsf{Sim}(x, \mathsf{ch}_i)$ of the underlying Sigma protocol. The only case in which the simulation fails is when $\mathsf{Sim}_{\blacktriangle 1}$ has to program the hash table H_1 adding the queries used to generate the proof π, mapping each transcript $(\overline{f}, i, \mathsf{ch}_i, z_i)$ to the associated digest s_i, but the input $(\overline{f}, i, \mathsf{ch}_i, z_i)$ was already queried by D_0. This happens with negligible probability because the underlying Sigma protocol is effective so the min-entropy of the commitments is super-logarithmic, and D_0 can query H_0 only about a polynomial number of inputs.

Online Extraction. According to Def. 4.3.3, the scheme described in Construction 1 admits an online extractor if there exists an algorithm Ext which, on input $\pi = (x, (f_i, \mathsf{ch}_i, z_i)_{i \in [L]})$ and $Q_H(\mathcal{A})$ outputs, except with negligible probability, a witness w for x.

The extractor Ext we define loops through the list $Q_H(\mathcal{A})$ and the all the transcripts $\{(f_i, \mathsf{ch}_i, z_i)_{i \in [L]}\}$ included in π looking for two accepting transcripts $(f, \mathsf{ch}, z), (f, \mathsf{ch}', z')$ with $\mathsf{ch} \neq \mathsf{ch}'$. If the search is successful Ext calls the knowledge extractor Ext of the underlying effective Sigma protocol on these values and outputs the same extracted value. If the search is unsuccessful, Ext outputs $\perp$.

Thus, in order to compute the failure probability of Ext, it is sufficient to bound the probability that $\mathcal{A}$ does not provide Ext with two such transcripts, but still the proof π verifies.

For the sake of simplicity we can restrict our analysis to algorithms $\mathcal{A}$ who only sends "meaningful random oracle queries", which are the queries that might be made also by a verifier to checks the validity of a proof π for a statement x. These queries are defined as:

- queries of the form $(x, \overline{f}, i, \mathsf{ch}_i, z_i)$;
- queries with $\mathsf{ch}_i = 1$;
- with $V_{1,\mathtt{FS}}(x, f_i, \mathsf{ch}_i, z_i) = 1$.

Otherwise we could build an algorithm $\mathcal{A}'$ which uses $\mathcal{A}$ as a subroutine and answers to non-meaningful queries programming an "internal" random oracle (internal to $\mathcal{A}'$), and to meaningful queries by forwarding the queries to the "real" random oracle. The interaction with $\mathcal{A}'$ would be indistinguishable to $\mathcal{A}$ from the interaction with a random oracle, therefore the probability that $\mathcal{A}$ outputs a valid proof π is unchanged, and if $\mathcal{A}'$ outputs the same proof π, the success probability of $\mathcal{A}$ and $\mathcal{A}'$ is the same.

We also ask that the algorithm $\mathcal{A}$ always sends a meaningful random oracle query associated to the components of the proof π whose challenge is set to 1. If $\mathcal{A}$ does not do that, we could instruct the wrapper algorithm $\mathcal{A}'$ to send the queries associated to those transcripts just before sending to Ext the proof π.

This means that for each $i \in [L]$ such that the proof $\pi = \{(f_i, \mathsf{ch}_i, z_i)_{i\in[L]}\}$ has $\mathsf{ch}_i = 0$, $\mathcal{A}$ has never queried the random oracle for $(x, \overline{f}, i, 1, z_i)$.

Therefore $\mathcal{A}$ has sent only τ meaningful queries for the components $i \in [L]$ for prefix $(x, \overline{f})$ s.t. $\mathsf{ch}_i = 1$. In order for the proof to be verified, the output of the random oracle on such queries $d_{i_1}, \ldots, d_{i_\tau}$ has to be equal to $\overline{d} \in \{0, \ldots, 2^b - 1\}$. The random oracle queries are distinct, therefore the outputs − which are all equal to $\overline{d}$ − have been sampled uniformly at random from the random oracle. The probability that these digests are equal is

$$\mathbb{P}(d_{i_1} = \cdots = d_{i_\tau}) = \sum_{c\in\{0,1\}^b} \mathbb{P}(d_{i_1} = \cdots = d_{i_\tau} = c) = 2^b \cdot (\frac{1}{2^b})^\tau = 2^{-b(\tau-1)}.$$

As a consequence,

$$\epsilon_s^{A,\mathsf{Ext}_\blacktriangle} \leq \frac{1}{2^{b(\tau-1)}}$$

□

Parameters

Table 4.1 reports the performance of our transform, as b and τ vary. In particular, as b varies in the set $\{2, 3, 4, 5, 6, 7, 8\}$, we calculate τ so that $b(\tau - 1) \geq 128$, and consequently L as $2^b(\tau - 1) + 1$. We will therefore calculate the number of group actions necessary to produce a proof, the number of $1s$ actually present among the challenges, the query complexity of the prover, i.e. the number of queries that a prover must make to a random oracle in order to produce the proof, and the overhead, i.e. the amount of data relating to the commitment to be sent in the proof. In the case of this transform, the overhead is always zero, as the data relating to the commitment can be entirely obtained from the remaining part of the proof. However, we have decided not to remove this column from Table 4.1, so that later the comparison between this transform and the subsequent ones that we will analyze will be facilitated.

	b	τ	L	#Group Actions	#1s	Query Compl.	Overhead (Kb)
GAO	2	65	257	257	65	257	-
	3	44	345	345	44	345	-
	4	33	513	513	33	513	-
	5	27	833	833	27	833	-
	6	23	1409	1409	23	1409	-
	7	20	2433	2433	20	2433	-
	8	17	4097	4097	17	4097	-

Table 4.1: As the parameters b and τ (associated to the security parameter $\lambda = 128$) vary, the table describes the number L of parallel repetitions needed for the proof. Furthermore, for each triplets (b, τ, L), the number of group action computations, the number of challenges equal to 1, the query complexity, and the overhead are represented.

Notice that the number of group actions coincides with the number of parallel repetitions L, the number of ones equals τ, and the query complexity equals the number of group actions computed.

4.3.4 Comparison With Known Transforms

In the following, we describe the known online-extractable transformations when applied to $\Sigma_\star$, comparing them with the transform that we have introduced in Subsection 4.3.3. As we will see, the NIZKPs obtained by applying the transforms in the simplest way return immense weights; a natural way to limit this problem is to try to fix the weight of the challenges (as suggested in Fiat-Shamir), which brings with it an increase in group actions to calculate.

Pass Transform Applied to $\Sigma_\star$

Given a Sigma protocol $\Sigma_\star$, the transform proposed by Pass in [Pas03] works as follows.

Construction 2. Let H be a random oracle and $((x_0, x_1), g) \in \mathcal{R}$ be a statement-witness pair given in input to P_P, whereas V_P is given as input only (x_0, x_1). The Pass transform is as follows.

Prover algorithm. The prover $\mathsf{P}_\mathsf{P}^H((x_0, x_1), g)$ executes the following instructions:

- generates λ first messages $f_1, \dots, f_\lambda$ of $\Sigma_\star$;
- for every $i \in \{1, \dots, \lambda\}$, considers the challenges 0 and 1, computes the associated responses $r_{i,0}, r_{i,1}$ and computes the commitments to these values $C_{i,0} \leftarrow H(r_{i,0})$ and $C_{i,1} \leftarrow H(r_{i,1})$,;
- computes the vector $\mathbf{f}'$ where $f'_i \leftarrow (f_i, C_{i,0}, C_{i,1})$;
- generates the challenge $\mathsf{ch} = H(\mathbf{f}', (x_0, x_1)) \in \{0,1\}^\lambda$;
- For each $i \in \{1, \dots, \lambda\}$, compute the response r'_i corresponding to f_i and $\mathsf{ch}_i \in \{0,1\}$ setting $r'_i \leftarrow r_{i,\mathsf{ch}_i}$. Let us define $\mathbf{r}' \leftarrow (r'_1, \dots, r'_\lambda)$;
- the resulting proof for (x_0, x_1) is $\pi \leftarrow (\mathbf{f}', \mathbf{r}') = (\mathbf{f}, \{(C_{i,0}, C_{i,1})\}_{i\in[\lambda]}, \{r_{i,\mathsf{ch}_i}\}_{i\in[\lambda]})$.

Since the Sigma protocol is first-message retrievable, the proof can be lightened by including only $\pi \leftarrow (\mathsf{ch}, \{C_{i,1-\mathsf{ch}_i}\}_{i\in\{1,\cdots,\lambda\}}, \mathbf{r}')$ from which the verifier can reconstruct $\mathbf{f}'$.

Verifier algorithm. The verifier $\mathsf{V}_{\mathsf{P}}(x_0, x_1)$ receives π, computes $\mathsf{ch} = H(\mathbf{f}', (x_0, x_1))$, checks that the transcripts associated with $(\mathbf{f}', \mathsf{ch}, \mathbf{r}')$ are λ valid transcripts and that the commitments are opened correctly.

We describe the behavior of the transformation in the context of our interest, describing some optimizations and the main limitations. A direct application of the Pass transform leads to proofs obtained by repeating the basic protocol λ times, clashing with two facts. On the one hand, the prover is forced to include λ hash values in the proof (corresponding to the commitments that are not opened), for a fixed overhead of 2Kb. On the other hand, the weight of the challenge ch can be modeled as a binomial with mean $\mu = \lambda$ and probability $p = 0.5$, for an average weight of $\lambda/2 = 64$ ones. For the dimensions of the responses associated with the 1-challenges with which we will have to deal later, this value is too high and, according to Principle 2, we would like to be able to lower it. A common and natural approach to achieve this goal is by the fixed-weight optimization, which allows the weight of the challenge to be reduced as desired, at the price of increasing the number of parallel repetitions necessary to obtain the same initial security level. According to this, we weighted the hash function in order to produce digests of the same weight as those used in our transformation (see Table 4.1). This levels the weight of the challenges and allows a fair comparison between the two transforms. Table 4.2 shows the Pass parameters for these weight values.

	w	t	#Group Actions	#1s	Query Compl.	Overhead (Kb)
PASS	65	132	132	65	265	2
	44	152	152	44	305	2.4
	33	210	210	33	421	3.5
	27	306	306	27	613	5.3
	23	458	458	23	917	8.4
	20	711	711	20	1423	13.6
	17	1334	1334	17	2669	28

Table 4.2: As the fixed weight w varies, t represents the minimum number of parallel repetitions necessary to obtain security level $\lambda = 128$. Furthermore, for every choice of (w, t), table above also represents the number of group actions computed, the number of ones present in the callenge vector, the query complexity of the prover and the associated overhead necessary to produce a proof.

Note that, using the Pass transformation applied to $\Sigma_\star$, the number of group actions computed is equal to the number of parallel repetitions t, and the number of 1s present in the challenge vector is equal to the parameter w. The query complexity is always double the number of parallel repetitions, and the overhead is λ bits for each repetition of the basic protocol $\Sigma_\star$, leading to a total of $\lambda \cdot t$ bits of overhead. It is clear that following Principle 2 results in reducing the weight of the challenges, but at the same time it places a greater distance from Principle 1, because we are forced to compute a greater number of group actions. In addition to this, increasing t results in a considerable increase in overhead (see Table 4.2), making the proof impractical.

Unruh Transform Applied to $\Sigma_\star$

A straight application of this transform is as follows. The prover is instructed to commit (using the random oracle) to the transcript of 2^ℓ invocation of $\Sigma_\star^\ell$ with the same first

message (made of ℓ first messages of $\Sigma_\star$) and all the 2^ℓ possible different challenges. Notice that, regardless of the choice of parameters, by construction of the non-interactive proof, the weight of the challenges is on average always the same, and is given by $(\ell/2) \cdot r = 64$. Similarly, the overhead is always 2Kb for every parameter choice.

It would be desirable to reduce as much as possible the weight of the challenges revealed, so that the revealed transcripts are lighter. A natural attempt to reduce the number of ones is to force each of the r vector-challenges to have weight 1. According to this, for each block of $\bar{\ell}$ first messages, We could instruct the prover to build the transcripts corresponding only to the $\bar{\ell} + 1$ challenges of weight less or equal than 1. This means that instead of instantiating Unruh using, for each repetition Σ_*^ℓ with $\ell \cdot r = \lambda$, we must instantiate it using, for each repetition $\Sigma_*^{\bar{\ell}}$ where $r \cdot \log_2(\bar{\ell} + 1) = 128$. In Table 4.3 we report some parameters choices for this optimization attempt. In particular, parameters have been chosen trying to match the number of group actions or the number of ones used in our transform. In particular, as the parameter r varies, the odd (even) lines of Table 4.3 determine ℓ in order to match the number of group action computations (the number of ones present in the vector challenge) of our transform, as described in Table 4.1.

	$\bar{\ell}$	r	t	#Group Actions	$\mathbb{E}[\#1s]$	Query Compl.	Overhead (Kb)
UNRUH	3	64	192	192	48	384	2
	5	53	265	265	44	530	2
	7	43	301	301	38	588	2
	11	36	396	396	33	792	2
	15	32	480	480	30	960	2
	23	28	644	644	27	1288	2
	31	26	806	806	25	1612	2
	47	23	1081	1081	23	2162	2
	63	22	1386	1334	22	4826	2
	84	20	1680	1680	20	3360	2
	127	19	2413	2413	19	4826	2
	184	17	3128	3128	17	6256	2
	255	16	4080	4080	16	8160	2

Table 4.3: As $\bar{\ell}$ and r vary, t represents the minimum number of parallel repetitions necessary to obtain security level $\lambda = 128$. For each parameter set, the table above also represents the number of group actions computed by the prover, the number of 1s to be included in the proof, the query complexity and the overhead due to the commitments.

Regarding Table 4.3, the number of group actions computed by the prover equals the number t of parallel repetitions, while the number of ones is computed as $\bar{\ell}/(\bar{\ell} + 1) \cdot r$. Similarly, the query complexity is always double the number of group actions computed, and the overhead is $r\lambda \cdot \log(\bar{\ell}+1)$. Notice that, as the number of ones decreases, the product $\bar{\ell} \cdot r$ (which represents the number of first messages, and so the group action computed by the prover) grows rapidly, forcing the prover to try a very large number of first messages. In addition to this, differently from our transform, there always remains the overhead due to commitments, as well as the (less important) query complexity, which is now doubled if compared with Table 4.1. This makes the optimization of this transform an impractical way to meet Principle 1 and 2 satisfactorily.

Fischlin Transform Applied to $\Sigma_\star$

A simpler way to avoid the overhead that is present in both the Pass and Unruh transforms, and more generally to turn $\Sigma_\star$ into an online-extractable NIZKP, is proposed by the Fischlin transform.

Construction 3 (Fischlin transform applied to $\Sigma_\star$)**.** Let H_b be a random oracle and $(x, y) \in \mathcal{R}$ a statement-witness pair given in input to $\mathsf{P}_{\mathtt{Fi}}$, whereas V is given in input only x. We instantiate the Fischlin transform as follows. Let ρ be the number of repetitions of the Fischlin transform, b the length of the digests of H_b and $\ell = b + 5$ the number of first messages of $\Sigma_\star$ in each of the ρ repetitions.

Prover algorithm The prover $\mathsf{P}_{\mathtt{Fi}}^{H_b}((x_0, x_1), g)$ executes the following instructions:

- sample uniformly at random $\ell \cdot \rho = (b+5) \cdot \rho$ group elements $\{\tilde{g}_{i,j}\}_{i\in[\rho],j\in[\ell]}$ and compute the first messages $f_i = (\tilde{g}_{i,1} \star x_0, \ldots, \tilde{g}_{i,\ell} \star x_0), \forall i \in [\rho]$;
- compute $\overline{f} \leftarrow (f_1 || \cdots || f_\rho)$ and executes the proof of work searching for the challenges $\mathsf{ch}_i \in \{0,1\}^\ell$, for each $i \in [\rho]$, such that

$$H_b(x, \overline{f}, i, \mathsf{ch}_i, r_i) = 0^b$$

  where $r_{i,j} = \tilde{g}_{i,j}$ if $\mathsf{ch}_{i,j} = 0$ and $r_{i,j} = \tilde{g}_{i,j} g^{-1}$ if $\mathsf{ch}_{i,j} = 1$.
- output the NIZKP of knowledge π of g

$$\pi \leftarrow (\overline{f}, \{\mathsf{ch}_i\}_{i\in\rho}, \{r_i\}_{i\in\rho}).$$

Verifier algorithm The verifier $\mathsf{V}_{\mathtt{Fi}}(x_0, x_1)$ receives in input $\pi = (\overline{f}, \{\mathsf{ch}_i\}_{i\in[\rho]}, \{r_i\}_{i\in[\rho]})$ and checks that

- the transcripts $(f_i, \mathsf{ch}_i, r_i)$ are valid. To do so, $\mathsf{V}_{\mathtt{Fi}}$ computes $g_{i,j} = r_{i,j}$, checking that $g_{i,j} \star x_{\mathsf{ch}_{i,j}} = f_{i,j}$ for all the ℓ sub-transcripts;
- $\forall i \in [\rho]$, checks that $H_b(x, \overline{f}, i, \mathsf{ch}_i, r_i) = 0^b$.

Fischlin transform applied to $\Sigma_\star$ in Construction 3 does not provide any guarantee about the size of the proof, basically because there is not a fine-grained control for the weight of the challenges (which in turn determines the number of group elements that must be included in it). Moreover, to guarantee a negligible completeness error, we must enlarge the size of the challenges for each repetition by 5 bits. This means that $\forall i \in [\rho]$, each first message f_i is given by $\ell = b + 5$ first messages $f_i = (f_{i,1}, \ldots, f_{i,\ell})$ of $\Sigma_\star$, the challenge ch_i is a bit string $(\mathsf{ch}_{i,1}, \ldots, \mathsf{ch}_{i,\ell}) \in \{0,1\}^\ell$, and the response $r_i = (r_{i,1}, \ldots, r_{i,\ell})$ is the response to the parallel instances of the Sigma protocol from cryptographic group actions corresponding to the first message f_i and the challenge ch_i. Therefore, as a consequence of Equations (4.12) and (4.13), the minimal number of first messages of $\Sigma_\star$ that must be computed is

$$\ell\rho = (b+5)\rho = b\rho + 5\rho = 128 + 5\rho.$$

Notice also that, since the Sigma protocol $\Sigma_\star$ is first-message retrievable, i.e. from the challenge $\mathsf{ch}_{i,j}$ and a response $g_{i,j}$ (derived from $r_{i,j}$) it is possible to retrieve the associated first message $f_{i,j} = g_{i,j} \star x_{\mathsf{ch}_{i,j}}$, then the proof π can be defined by omitting the value $\overline{f}$, setting

$$\pi = (\{\mathsf{ch}_i\}_{i\in\rho}, \{r_i\}_{i\in\rho}).$$

In order to reduce the dimension of the proof, according to Principle 1, the smaller is the number of ρ, the smaller is the number of group action computations; however, as a downside, the smaller is ρ, the larger becomes the value b (by Equation (4.13)) which makes the query complexity (i.e. the number of random oracle queries) grow exponentially. Furthermore, we will show that the increase in the query complexity is linked to the number of challenges equal to one, and therefore, pursuing this direction contradicts Principle 2. In the following, we provide some attempts to balance between these two factors.

To reduce the weight of the challenge found while performing the Fischlin transform, an immediate approach is to instruct the prover algorithm to try the challenges in increasing order of weight: at first the challenge 0, then all the challenges of weight 1, then all the challenges of weight 2, up to the challenge of weight $\ell = b + 5$. As a result, the prover will find the lighter challenge satisfying the predicate $H_b(x, \overline{f}, i, \mathsf{ch}_i, r_i)$. Consider a proof π as above. Given λ, b, ρ such that $b \cdot \rho = \lambda$, and $\ell = b + 5$, we are interested in quantifying the expected number of challenges $\mathsf{ch}_{i,j} = 1$. Given a set of first messages $\overline{f}$, for each repetition $i \in [\rho]$, the expected weight of ch_i can be computed easily if we fix an order in which the challenges are chosen. In particular, we choose challenges in increasing weight, starting from $1 = \binom{\ell}{0}$ element of weight 0 (which is 0^ℓ), then $\ell = \binom{\ell}{1}$ challenges of weight 1, then $\binom{\ell}{2}$ challenges of weight 2, and so on. As a consequence, in the following, for every $j \in [\ell]$, we denote with $k_j = \sum_{i=0}^{j} \binom{\ell}{i}$ the number of attempts associated to a challenge with size less or equal to j. Let $Q_{\mathtt{Fi}}(b)$ be the random variable describing the number of attempts before finding a solution, and denote with $W(b, \ell)$ the random variable representing the weight of the lighter challenge satisfying $H_b(\overline{f}, i, ch_i, r_i) = 0^b$. The expected value of $W(b, \ell)$ is given by

$$\begin{aligned}
\mathbb{E}[W(b,\ell)] &= \sum_{i=1}^{\ell} i \Pr[W(b) = i] = \sum_{i=1}^{\ell} i \Pr[k_{i-1} < Q_{\mathtt{Fi}}(b) \leq k_i] \\
&= \sum_{i=1}^{\ell} i \Pr[k_{i-1} < Q_{\mathtt{Fi}}(b)] \Pr[Q_{\mathtt{Fi}}(b) \leq k_i | k_{i-1} < Q_{\mathtt{Fi}}(b)] \\
&= \sum_{i=1}^{\ell} i \Pr[k_{i-1} < Q_{\mathtt{Fi}}(b)] \Pr[Q_{\mathtt{Fi}}(b) \leq k_i - k_{i-1}] \\
&= \sum_{i=1}^{\ell} i (1 - 2^{-b})^{k_{i-1}} (1 - (1 - 2^{-b})^{k_i - k_{i-1}}).
\end{aligned} \tag{4.14}$$

This is the expected value of the weight of the challenge, and therefore the expected number of group elements that must be revealed in each repetition of the plain Fischlin transform. Note that this value is a function of b since from b we determine $\ell = b + 5$. We summarize this value for different parameter choices in Table 4.4, using $\lambda = 128$ as security parameter.

b	ρ	ℓ	#Group Actions	#1s	Query Compl.	Overhead (Kb)
2	64	7	448	54	256	-
3	43	8	344	51	344	-
4	32	9	288	48	512	-
5	26	10	260	48	832	-
6	22	11	242	48	1408	-
7	19	12	228	48	2432	-
8	16	13	208	46	4096	-

Table 4.4: Given the security level $\lambda = 128$, as parameters $b, \rho = \lceil 128/b \rceil, \ell = b+5$ vary, the table above represents the number of group action operations that must be computed by the prover and the verifier; the expected weight of the challenge ch, the query complexity to generate a proof with parameters b and ρ; the overhead, which is always zero.

Notice that the expected number of queries that the prover must perform depends on the value of b since the digests are random elements in $\{0, \dots, 2^b - 1\}$, therefore in order to find the target challenge satisfying the Fischlin's predicate the expected number of queries in 2^b. Since the number of repetitions is ρ the overall query complexity to generate a proof is $\rho \cdot 2^b$. Table 4.4 also points out that the weight of the challenges is quite high. We describe some optimization that allows us to slightly improve the size of the proof reducing the weight of the challenge.

A first attempt to address this problem could be as follows. Similarly to what we have done for the Unruh transform, we would like to always have challenges of weight 1. In this case, this means that the challenge embedded in a proof has a weight that does not exceed the number of repetitions. To ensure this, according to the observation by Chen and Lindell in [CL24], the prover should consider challenges of approximately 2^{b+5} bits (see Equation (4.12)) to guarantee to be able to try only challenges of weight ≤ 1 while seeking an inversion of H_b of 0^b. Consequently, this will necessitate the calculation of $2^{b+5}\rho$ group actions, rendering the selection of parameters highly inefficient from a computational cost perspective.

A second approach which could be helpful in the creation of a proof associated with a lighter challenge takes inspiration from the transform introduced in the previous subsection and would be to look for a collision of the hash function H_b over all ρ repetitions of the Fischlin transform. This approach allows to reduce the weight of the challenges included in the NIZKP obtained using the transform proposed by Fischlin [Fis05], while it increases the amount of random oracle queries that must be performed to output a valid NIZKP. This optimization is surpassed from every perspective by the optimization presented by Kondi and Shelat [KS22] which reduces even more the weight of the challenges, making and also reduces the query complexity of the scheme.

Optimization by Kondi and Shelat [KS22]. In [KS22], the authors propose a modification of the Fischlin transform that results in a reduced query complexity. The reduction in query complexity naturally results in a reduction in challenge weight. The idea is the following: instead of looking for a preimage of the digest 0^b, for each of the ρ repetitions, the authors instruct the prover to create $\frac{\rho}{2}$ pairs of first messages $(f_1, f_2), \cdots, (f_{\rho-1}, f_\rho)$, and for each $i \in [\frac{\rho}{2}]$ to look for challenges $\mathsf{ch}_{2i-1}, \mathsf{ch}_{2i}$ so that the corresponding digests $H_{2b}(\overline{f}, 2i-1, \mathsf{ch}_{2i-1}, r_{2i-1}) = H_{2b}(\overline{f}, 2i-1, \mathsf{ch}_{2i}, r_{2i})$, where the digest size is doubled. This

yields the same soundness error, using the same number ρ of repetitions of the Sigma protocol. The authors show in [KS22, Section 5.1] that the query complexity using this approach is reduced by the 10-15% for values of b and this increases the efficiency by reducing the number of challenges to try to find a solution. This allows to reduce the weight of the challenges if we keep the challenge space unchanged. Below we report the expected weight of the challenges when we use this optimization. Note that we needed to increase the number of repetitions for $\rho = 43$ and $\rho = 19$ to 44 and 20 respectively, to guarantee that we could create all the needed pairs.

In order to make a fair comparison with the performance analysis of the Fischlin transform, we must evaluate the expected weight of challenges when the query complexity and the number of group actions computed is the same. Given a value of b as the number of output bits of the random oracle in both Fischlin and our transform, this can be done by assigning to each of the $\rho(b)$ repetitions of the Fischlin transform a number $\ell(b)$ of first messages of $\Sigma_\star$ that is equal to the number $L(b)$ of first messages of $\Sigma_\star$ used in our transform divided by ρ, that is, $\ell(b) = L(b)/\rho(b)$, where $\rho(b) = \lceil 128/b \rceil$. Notice also that, given the size of the challenge $\ell(b)$, we can compute the expected query complexity for each of the ρ blocks and also the expected weight associated with the challenges, using Equation 4.14. Results are reported in Table 4.5.

	$2 \cdot b$	ρ	L	#Group Actions	#1s	Query Compl.	Overhead (Kb)
Fischlin-KS	4	64	320	320	72	244	-
	6	44	352	352	54	325	-
	8	32	544	544	38	464	-
	10	26	858	858	31	744	-
	12	22	1430	1430	27	1249	-
	14	20	2440	2440	25	2269	-
	16	16	4112	4112	20	3635	-

Table 4.5: Given the security level $\lambda = 128$, as parameters b, ρ and L vary, the table reports the number of group action operations that must be computed by the prover and the verifier, the expected weight of the challenge ch and the query complexity to generate a proof with parameters b and ρ.

Regarding Table 4.5, notice that the number of group action to be computed equals $L = \rho \cdot \ell$ which is always greater of equal to the corresponding value L in Table 4.1 for the same value of b. The number of ones and the expected query complexity of the challenge is computed experimentally on 1000 random proofs. More importantly, once fixed the desired challenge weight, the transform always behaves worse than the one introduced in the previous subsection and described in Table 4.1.

4.3.5 Trading Query Complexity for More Compact NIZKP

We introduce a technique which can be used to balance between the query complexity and the completeness error. We have seen that it is possible to obtain completeness 1 simply by setting $L = 2^b(\tau - 1) + 1$. The doubt that naturally arises is whether it is possible to reduce L so that ϵ_c^{P} is still reasonably low, but L is significantly lower than $2^b(\tau - 1) + 1$. To answer this question, we model the hash function as a random oracle and compute the probability that in a sequence of L elements with values in $\{0, \ldots, 2^b - 1\}$ there are at least τ occurrences of the same element. To answer this question we introduce the definition of

k-counterimaginable functions.

Definition 4.3.12 (k-Counterimaginable Function)**.** Let m, n and k be three natural numbers, and consider a function $f : [m] \longrightarrow [n]$. We say that f is k-counterimaginable if, for every $i \in [n]$ it holds that $|\{f^{-1}(i)\}| \leq k$. Furthermore, we denote by $\mathcal{F}(k, m, n)$ the number of k-counterimaginable functions $f : [m] \longrightarrow [n]$. Formally:

$$\mathcal{F}(k, m, n) := |\{f : [m] \longrightarrow [n] \ : \ f \ \text{ is } \ k\text{-counterimaginable}\}|.$$

This reflects the problem of computing the completeness error ϵ_c^{P} for the scheme described in Construction 1. The probability that P fails to produce a valid proof $\pi \neq \perp$ is

$$\epsilon_c^{\mathsf{P}} = \frac{\mathcal{F}(\tau - 1, L, 2^b)}{(2^b)^L}. \tag{4.15}$$

where $\mathcal{F}(\tau - 1, L, 2^b)$ is the number of $(\tau - 1)$-counterimaginable functions and $(2^b)^L$ is the number of functions with domain size L and codomain size 2^b. The formula for quantifying $\mathcal{F}(k, m, n)$ is given by the following result.

Proposition 4.3.13. *Let m, n and k be three natural numbers. Then,*

$$\begin{cases} \mathcal{F}(k, m, n) = \sum_{c=0}^{\lfloor m/k \rfloor} \frac{1}{c!} \Big(\prod_{j=0}^{c-1} \binom{m-kj}{k} \Big) \mathcal{F}(1, c, n) \mathcal{F}(k-1, m-kc, n-c) \\ \mathcal{F}(1, m, n) = m! \binom{n}{m}. \end{cases} \tag{4.16}$$

Proof. If $k = 1$, the number of 1-counterimaginable (or injective) functions $f : [m] \longrightarrow [n]$ can be computed considering that every element of the domain must be set by f to a different element of the codomain. We have $\binom{n}{m}$ different subsets of $[n]$ with cardinality m, and for each subset we can choose f in $m!$ different ways. If $k > 1$, $\mathcal{F}(k, m, n)$ can be computed recursively computing the number of maps $N_{k,c}$ which have c k-uples of colliding domain elements (as c varies in $\{0, \ldots, \lfloor m/k \rfloor\}$) and such that they are $(k-1)$-counterimaginable on the remaining elements, which is

$$N_{k,c} = (1/c!) \Big(\prod_{j=0}^{c-1} \binom{m-kj}{k} \Big) \mathcal{F}(1, c, n) \mathcal{F}(k-1, m-kc, n-c)$$

where

- $(1/c!) \Big(\prod_{j=0}^{c-1} \binom{m-kj}{k} \Big)$ is the number of way one can pick c subsets of k elements out of m;
- $\mathcal{F}(1, c, n)$ is the number of way the c subsets can be assigned to different values in n;
- $\mathcal{F}(k-1, m-kc, n-c)$ is the number of $(k-1)$-counterimaginable functions on the elements that ar not part of the c k-tuple.

Therefore, since the sets just described are a partition of the set of k-counterimaginable functions, the thesis follows. □

Remark 4.3.14. The number of functions that satisfy Def. 4.3.12 can be used to describe a generalization of the birthday paradox in which we are interested in quantifying the probability that, among m people, at least k of them have the same birthday.

This result allows us to quantify the completeness probability, and therefore, having fixed parameters L, τ, b, compute α so that a prover is able to sign with probability $1-2^{-\alpha}$. According to [CL24], we provide sets of parameters so that the completeness error is $\epsilon_c^{\mathsf{P}} < 2^{-40}$.

Example 4.3.15. In the following, we consider an instance of GAO where we took $b = 4$ and $\tau = 33$. Notice that, in order to have a completeness error 0, it is sufficient to choose $L = 2^b(\tau - 1) + 1 = 513$. In Table 4.4 we represent, by varying L, the completeness probability that we would obtain. In particular, to reach a completeness error $\epsilon_c^{\mathsf{P}_\blacktriangle} \leq 2^{-40}$ we need $L \geq 506$. To reach a completeness error of $\epsilon_c^{\mathsf{P}_\blacktriangle} \leq 2^{-10}$ we need $L \geq 460$.

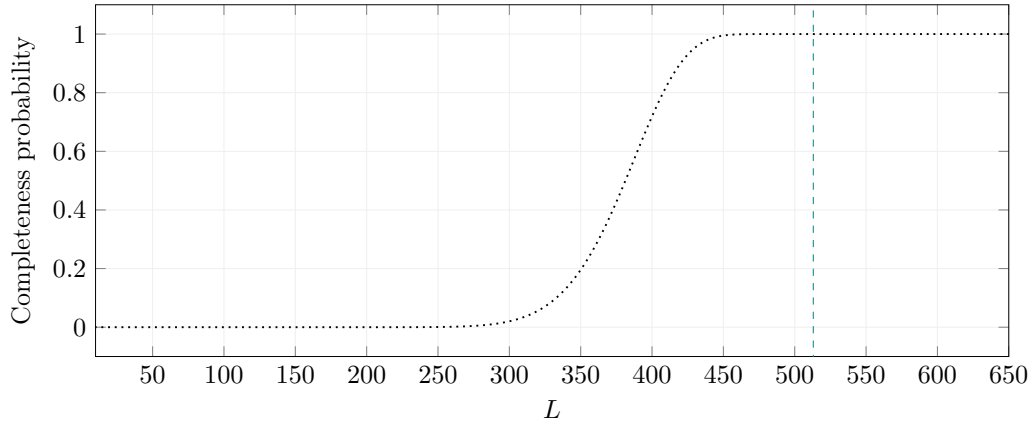

Figure 4.4: The figure reports the completeness probability $(1 - \epsilon_c^{\mathsf{P}})$ for a GAO's instance with $b = 4, \tau = 33$. The green dashed vertical line corresponds to $L = 513$ and is associated to a completeness error $\epsilon_c^{\mathsf{P}} = 0$.

General Construction

Our construction is as follows.

Construction 4. Let (P, V) be an effective Sigma protocol with challenges of ℓ bits for relation $\mathcal{R}$. We define a non-interactive proof system $(\mathsf{P}_\blacksquare, \mathsf{V}_\blacksquare)$ for relation $\mathcal{R}$ in the random oracle model which is defined by the following parameters. Let L be the total number of "parallel repetitions" conducted, τ the weight of the challenge in the proof, b the output size in bits of the random oracle, and $k_{\max}$ the maximum number of attempts in order to create a NIZKP. Then the algorithms $(\mathsf{P}_\blacksquare, \mathsf{V}_\blacksquare)$ are defined as follows.

- *Prover.*

  1. The Prover ${\mathsf{P}_\blacksquare}^H(x, y)$ initializes an integer counter $k = 1$ and runs the prover $\mathsf{P}(x, y)$ in L independent repetitions to obtain L first messages $f_1, \ldots, f_L$. Let $\overline{f} = (f_1, \ldots, f_L)$.
  2. ${\mathsf{P}_\blacksquare}^H$ sets $\mathcal{T}_k(d) = \emptyset$ for every $d \in \{0,1\}^b$. Given $i \in [L]$, ${\mathsf{P}_\blacksquare}^H(x, y)$ computes $d := H(k, x, \overline{f}, i, 1, z_i) \in \{0, \ldots, 2^b - 1\}$ (where $V(x, f_i, 1, z_i) = 1$) and adds i to $\mathcal{T}_k(d)$.
  3. Being $D = \{d : |\mathcal{T}_k(d)| \geq \tau\}$
     - if $D \neq \emptyset$, sample $\bar{d} \in D$ uniformly at random, and sample uniformly at random $\mathcal{T}'_k(\bar{d}) \subseteq \mathcal{T}_k(\bar{d})$, with $|\mathcal{T}'_k(\bar{d})| = \tau$. P generates $\mathsf{ch} \in \{0,1\}^L$ setting,

$\forall i \notin \mathcal{T}'_k(\bar{d})$ $\mathsf{ch}_i = 0$, and $\mathsf{ch}_i = 1$ otherwise. Finally, for every $i \notin \mathcal{T}'_k(\bar{d})$, the prover sets z_i to be the response associated to f_i and ch_i, and outputs $\pi = (k, f_i, \mathsf{ch}_i, z_i)_{i\in[L]}$.

– If $D = \emptyset$ and $k < k_{\max}$, then $\mathsf{P}_\blacksquare$ sets $k := k+1$ and goes to Step 2, otherwise returns $\perp$.

- *Verifier.* The verifier $\mathsf{V}_\blacksquare^H$, on input x and $\pi = (k, f_i, \mathsf{ch}_i, z_i)_{i\in[L]}$ accepts if and only if the following conditions holds:

  1. $k \leq k_{\max}$;
  2. $\mathsf{V}_1(x, f_i, \mathsf{ch}_i, z_i) = 1$ for every $i \in [L]$;
  3. $z_i = 1$ on exactly τ indexes $i_1, \ldots, i_\tau$;
  4. $H(k, \overline{f}, i_1, 1, z_{i_1}) = \cdots = H(k, \overline{f}, i_\tau, 1, z_{i_\tau})$.

This construction allows the prover and the verifier to compute a smaller number of first messages, i.e. to compute less group actions compared to Construction 1, which is desirable according to Principle 1. The cost incurred is for the prover, who may need to compute up to L hash functions as many as $k_{\max}$ times, potentially increasing the query complexity for the prover to $L \cdot k_{\max}$. In contrast, the verifier benefits from this setup by receiving the transcripts along with an index $k \in [k_{\max}]$. We prove that also Construction 4 is a non-interactive zero-knowledge proof of knowledge with online extractor as for Definition 4.3.3.

Theorem 4.3.16. *Let* (P, V) *be an effective Sigma protocol for relation* $\mathcal{R}$*. Define* $(\mathsf{P}_\blacksquare, \mathsf{V}_\blacksquare)$ *as the non-interactive proof system obtained by applying Construction 1 to* (P, V)*. Assuming*

$$1-\left(1-\frac{1}{2^{b(\tau-1)}}\right)^{k_{max}} \leq 2^{-\lambda}, \quad \tau \leq L < 2^b(\tau-1)+1, \quad \text{and} \quad k_{max} \leq \left\lceil \frac{-40}{\log_2\left(\frac{\mathcal{F}(\tau-1,L,2^b)}{(2^b)^L}\right)} \right\rceil$$

then $(\mathsf{P}_\blacksquare, \mathsf{V}_\blacksquare)$ *is a non-interactive zero-knowledge proof of knowledge for relation* $\mathcal{R}$ *(in the random oracle model) with an online extractor, completeness error* $\epsilon_c^{\mathsf{P}_\blacksquare} \leq 2^{-40}$*, and soundness error* $\epsilon_s^{\mathsf{P}_\blacksquare,\mathsf{Ext}_\blacksquare} \leq 2^{-\lambda}$*.*

Proof. **Completeness** In this construction the completeness error is given by

$$\begin{aligned}
\epsilon_c^{\mathsf{P}} &= \Pr\left[\perp \xleftarrow{\$} \mathsf{P}(w,x) | (w,x) \in \mathcal{R}\right] \\
&= \Pr[\,\forall i \in [k_{\max}] : \{d : |\mathcal{T}_i(d)| \geq \tau\} = \emptyset] \\
&= (\Pr[\{d : |\mathcal{T}_i(d)| \geq \tau\} = \emptyset])^{k_{\max}} \\
&= \left(\frac{\mathcal{F}(\tau-1, L, 2^b)}{(2^b)^L}\right)^{k_{\max}} \leq 2^{-40}.
\end{aligned}$$

and the expected number of salts k that the prover must try is $\frac{(2^b)^L}{\mathcal{F}(\tau-1,L,2^b)}$.

Zero-Knowledge We design a simulator $\mathsf{Sim}_\blacksquare$ which uses the simulator $\mathsf{Sim}_\blacktriangle$ that we have presented in the proof of Theorem 4.3.11. The way it answers to random oracle queries is the same way used by $\mathsf{Sim}_\blacktriangle$.

Once it receives from D_0 a pair $(w, x) \in \mathcal{R}$, $\mathsf{Sim}_\blacksquare$ executes the following operations:

1. $\mathsf{Sim}_{\blacksquare}$ sets $k = 1$
2. if $k > k_{\max}$, $\mathsf{Sim}_{\blacksquare}$ returns $\perp$. Otherwise $\mathsf{Sim}_{\blacksquare}$ generates L digests $d_j^{(k)}, j \in [L]$, sets $\mathcal{T}_k(d) = \{j \in [L] : d_j^{(k)} = d\}, \forall d \in \{0, \ldots, 2^b - 1\}$, and sets $D_k = \{d \in \{0, \ldots, 2^b - 1\} : \mathcal{T}_k(d) \geq \tau\}$.
3. if $D = \emptyset$ then the simulator increments k by 1 and starts again from Item 2.
4. if $D \neq \perp$ set $\overline{k} = k$, $\mathsf{Sim}_{\blacksquare}$ samples at random $\overline{d}$ in D and a random subset $\mathcal{T}'_{\overline{k}}(\overline{d}) \subseteq \mathcal{T}_{\overline{k}}(\overline{d})$ of cardinality τ. Then it computes ch setting $\mathsf{ch}_i = 1, \forall i \in \mathcal{T}'_{\overline{k}}(\overline{d})$, and $\mathsf{ch}_i = 0$ otherwise.
5. $\mathsf{Sim}_{\blacksquare}$ runs $\forall i \in [L]$ the simulator Sim of the underlying effective Sigma protocol $\Sigma_\star$ on statement x and challenge ch_i, generating L transcripts $(f_i, \mathsf{ch}_i, z_i)$.
6. $\mathsf{Sim}_{\blacksquare 1}$ programs the hash table H_1 as H_0 to which are added the pairs,

$$((\overline{f}, i, \mathsf{ch}_i, z_i), d_i^{(k)}), \forall k \leq \overline{k},$$

i.e. $H_1(k, x, \overline{f}, i, \mathsf{ch}_i, z_i) = d_i^{(k)}$ and outputs $(\pi, H_1) = \left(\{(\overline{k}, f_i, \mathsf{ch}_i, z_i)\}_{i \in [L]}, H_1\right)$.

Online Extractability According to Def. 4.3.3, the scheme described in Construction 4 admits an online extractor if there exists an algorithm Ext which, on input $\pi = (k, x, (f_i, \mathsf{ch}_i, z_i)_{i \in [L]})$ and $Q_H(A)$, outputs (except with negligible probability) a witness w for x. As for the extractor described in the proof of Theorem 4.3.11, this extractor loops through the queries in $Q_H(A)$ and the transcripts included in the proof looking for two valid transcripts for the statement x with the same first message and different challenges.

We can assume that the algorithm A, that the extractor $\mathsf{Ext}_{\blacksquare}$ interacts with, only makes queries

- of the form $(k, x, \overline{f}, i, \mathsf{ch}_i, z_i)$;
- with $k \leq k_{\max}$;
- for $\mathsf{ch}_i = 1$;
- such that $V(x, f_i, \mathsf{ch}_i, z_i) = 1$;

and also that it makes a meaningful random oracle query for every transcript included in the proof π associated to $\mathsf{ch}_i = 1$. We can restrict to this kind of algorithm for the same reasons described in the proof of theorem 4.3.11.

Let $\pi = (\overline{k}, \{(f_i, \mathsf{ch}_i, z_i)\}_{i \in [L]})$ be the proof of x that A has generated and $\overline{f} = (f_1, \ldots, f_L)$. If A has managed to produce a proof without letting $\mathsf{Ext}_{\blacksquare}$ extract the witness, then $Q_H(A)$ does not contain a query $(\overline{k}, x, \overline{f}, i, 1, z'_i)$ for any $i \in [L]$ such that $\mathsf{ch}_i = 0$, where $(\overline{k}, \overline{f}, x)$ is the prefix associated to π. This means that A has managed to guess τ components that for the prefix $(\overline{k}, \overline{f}, x)$ yield the same digest $\overline{d}$. But this happens with probability

$$\epsilon_s^{A, \mathsf{Ext}_{\blacksquare}} = 1 - \left(1 - \frac{1}{2^{b(\tau-1)}}\right)^{k_{\max}} \leq 2^{-\lambda}$$

since, for any value of $\overline{k} \in [k_{\max}]$, the security is the same as in the proof of Theorem 4.3.11.

□

Observe that the prover can reduce the amount of group action computations by accepting to be forced to compute a higher amount of inexpensive hash computations. In particular, the expected number of hash computations $Q(b, L)$ will be given by $Q(b, L) = L \cdot \Pr[\{d : \mathcal{T}(d) \geq \tau\} \neq \emptyset]^{-1}$. In Table 4.6 we provide the expected number of first messages needed to guarantee that the prover can create a NIZKP. This number corresponds to the completeness error $\epsilon_c^{\mathsf{P}_\blacksquare} = \frac{1}{2}$.

	b	τ	L	#Group Actions	#1s	Query Compl.	Overhead (Kb)
$\mathrm{GAO}_{0.5}$	2	68	229	229	68	458	-
	3	46	284	284	45	568	-
	4	35	385	385	34	770	-
	5	28	570	570	28	1140	-
	6	24	863	863	24	1726	-
	7	20	1344	1344	20	2688	-
	8	18	1994	1994	18	3988	-

Table 4.6: As b, τ and s vary, L is computed so that the completeness error is approximately 0.5. For each choice (b, τ), the tests were done on 1000 random samples, taking as a result the smallest L such that the associated completeness error was smaller than 0.5. The query complexity reported in this table is the theoretical mean of the associated random variable.

Notice that, for the parameter choice described in the previous table, since $\epsilon_c^{\mathsf{P}_\blacksquare} = 0.5$, if the corresponding value of $k_{\max}$ is set to 40, then the expected number of different k that $\mathsf{P}_\blacksquare$ must try to generate a NIZKP π is 2, and the prover fails (outputting $\perp$) only with probability 2^{-40}.

4.3.6 Optimizations and Applications

Below we propose two standard optimizations, the seed tree and the multiple public-key optimizations, observing how the transformation introduced above behaves in this regard.

The Seed Tree Optimization

To reduce the size of the proof, it is possible to optimize the way seeds are built and revealed. A standard technique to generate the L seeds used to generate the first messages of the sigma protocol consists in generating them all starting from a single λ-bits seed seed. This seed is given in input to a $\mathsf{PRF} : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ where the first λ bits of the output will be the left child and the second λ bits are the right child of seed. The procedure is repeated until all the needed leaves have been generated. Note that to reveal the seeds corresponding to the challenges $\mathrm{ch}_i = 0$ in our constructions will be sufficient to reveal the parents of these elements that are not shared with the seeds corresponding to $\mathrm{ch}_i = 1$. According to [Bor+23, Proposition 2], let $L = L_1 + L_2 + \cdots + L_u$, where L_i are the powers of two defining the structure of the seed tree. Then, let $U \subseteq [L], |U| = \tau$ be the set of leaves of the tree that must remain hidden, the number of seeds that must be revealed to disclose all the other leaves in $L \setminus U$ is not greater than

$$N_{\mathsf{seed}} = \tau \log(L/\tau) + u - 1.$$

The Multiple Public-Key Optimization

Using multiple public keys is a well-established technique to manage the trade-off between the size of the public key and the size of the proof [DFG19]. The idea behind it is to introduce several public keys $y_1, \dots, y_s$, obtained from several secret keys $g_1, \dots, g_s$ by computing $y_i = g_i \star x$. In this way, given a commitment $\tilde{x} = g \star x$, the possible challenges are given by the union of all the paths $\tilde{g}g_i^{-1}$ between y_i and the commitment, together with the path $\tilde{g}$ which sends x to $\tilde{x}$. The challenge space then grows from $\{0, 1\}$ to a larger set $\{0, \dots, s\}$. A schematic representation of this technique is proposed in Figure 4.5.

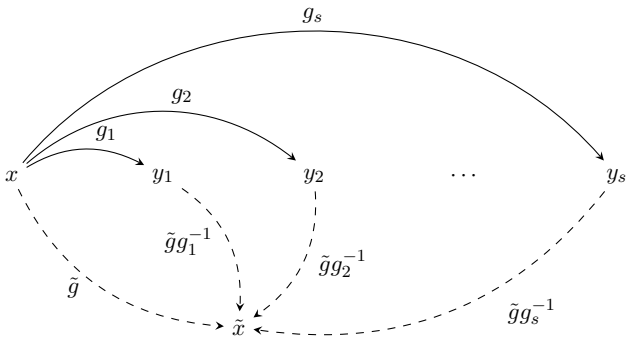

Figure 4.5: Graphical representation of $\Sigma_\star$ when s public keys are used.

We adapt the description of the GAO transform to the case in which the public key of the prover is given by s set elements $y_1, \dots, y_s$. In this new scenario, the challenge space Ch has cardinality $s+1$. Notice that in this case, the security of the whole protocol is based on mGAIP. Furthermore, the new protocol preserves the correctness, online extractability, and zero-knowledge properties. We observe that, unlike the effect that multiple public keys have on NIZKP obtained by applying the Fiat-Shamir transform, the use of multiple public keys with our transform does not reduce the soundness error of the protocol, but instead reduces its completeness error and allows the prover to consider a smaller number L of first messages. As before

$$\epsilon_c^{\mathsf{P}} = \Pr\left[\perp \xleftarrow{\$} \mathsf{P}(x, y) \mid (x, y) \in \mathcal{R}\right] = (\Pr[\ \{d : |\mathcal{T}_k(d)| \geq \tau\} = \emptyset])^{k_{\max}}\,.$$

If $s = 1$, as we have previously noted, this probability can be computed looking for the probability of having at least τ collisions between L random variables. If $s > 1$, we have to look for the probability of a τ-collision between L sets of random variables, where every set has s elements. Given the difficulty of providing an explicit formula for this probability, we resort to experimental results, which are reported in Table 4.7.

Applications. In the following, we propose an application of this transform to LESS [Bia+20], a code-based scheme currently competing for the NIST standardization of post-quantum signatures alternative to those based on lattices. LESS is based on a sigma protocol $\Sigma_\star$ where the basic set X is a $[n, k]$ linear code over $\mathbb{F}_q$, and the group acting on this set is the group of monomial maps, i.e. those maps obtained by permuting the starting code, and possibly scaling the coefficients. In particular, a $[n, k]$ linear code can be represented in systematic form using $\ell_X = k(n-k)\log(q))s$ bits, while a monomial map can be seen as the composition of a permutation matrix S_n with a diagonal matrix $(\mathbb{F}_q^*)^n$ of non-zero entries, and can be represented with $n\,(\log(n) + \lceil \log(q) \rceil)$ bits. For the security

b	ρ	$L_{s=1}$	$L_{s=2}$	$L_{s=3}$	$L_{s=4}$	$L_{s=5}$	$L_{s=6}$	$L_{s=7}$
2	68	229	134	102	88	80	75	72
3	46	284	153	111	89	77	69	63
4	34	385	199	139	109	91	79	70
5	28	570	290	200	151	124	105	93
6	24	863	437	296	225	182	153	133
7	20	1344	674	444	337	276	231	198
8	18	1994	1001	669	501	405	336	289

Table 4.7: As b, ρ and s vary, the table shows the values of L so that the completeness error (of a single iteration) is approximately 0.5. For each choice (b, ρ, s), the tests were done on 1000 random samples, taking as a result the smallest L such that the associated completeness error was smaller than 0.5.

parameter $\lambda = 128$ bits we have $q = 127, n = 252$ and $k = 126$, so that the size of a group elements is $\mathsf{size}_G^{\mathsf{LESS}} = 237$ bytes, and the size of set element is $\mathsf{size}_X^{\mathsf{LESS}} = 13892$ bytes. We stress that a seed is a λ bit sting, so it can be represented with 16 bytes. Table 4.8 reports the associated values for different parameters choices. The code used to compute data for this section can be found at https://github.com/triki96/GAO-transform

	b	τ	L	N_{seed}	$\mathsf{size}_{\mathsf{pk}}^{\mathsf{LESS}}$ (Kb)	$\mathsf{size}_{\mathsf{Sig}}^{\mathsf{LESS}}$ (Kb)		b	τ	L	N_{seed}	$\mathsf{size}_{\mathsf{pk}}^{\mathsf{LESS}}$ (Kb)	$\mathsf{size}_{\mathsf{Sig}}^{\mathsf{LESS}}$ (Kb)
$s=1$	2	68	229	140	13.9	18.4	$s=5$	2	68	80	68	69.5	17.2
	3	46	284	141	13.9	13.2		3	46	77	49	69.5	11.7
	4	35	385	138	13.9	10.2		4	35	91	72	69.5	9.2
	5	28	570	144	13.9	8.9		5	28	124	88	69.5	8
	6	24	863	151	13.9	8.1		6	24	182	76	69.5	6.9
	7	20	1344	142	13.9	7		7	20	276	82	69.5	6.1
	8	18	1994	132	13.9	6.4		8	18	405	94	69.5	5.7
$s=2$	2	68	134	70	27.8	17.2	$s=6$	2	68	75	71	83.4	17.3
	3	46	153	95	27.8	12.4		3	46	69	48	83.4	11.7
	4	35	199	106	27.8	9.8		4	34	79	72	83.4	9.2
	5	28	290	114	27.8	8.4		5	28	105	59	83.4	7.6
	6	24	437	125	27.8	7.7		6	24	153	75	83.4	6.8
	7	20	674	123	27.8	6.7		7	20	231	85	83.4	6.1
	8	18	1001	114	27.8	6.1		8	18	336	92	83.4	5.7
$s=3$	2	68	102	71	41.7	17.3	$s=7$	2	68	72	69	97.3	17.2
	3	46	111	97	41.7	12.5		3	46	63	51	97.3	11.7
	4	34	139	105	41.7	9.7		4	34	70	70	97.3	9.1
	5	28	200	86	41.7	8		5	28	93	60	97.3	7.6
	6	24	296	98	41.7	7.3		6	24	133	74	97.3	6.9
	7	20	444	105	41.7	6.4		7	20	198	83	97.3	6
	8	18	669	113	41.7	6.1		8	18	289	92	97.3	5.7
$s=4$	2	68	88	70	55.6	17.2							
	3	46	89	49	55.6	11.7							
	4	34	109	72	55.6	9.2							
	5	28	151	88	55.6	8							
	6	24	225	99	55.6	7.3							
	7	20	337	103	55.6	6.4							
	8	18	501	96	55.6	5.8							

Table 4.8: Once fixed the values s, b, τ, the table above represents the value L necessary to have $\epsilon_c^{\mathsf{P}^\blacksquare} = 0.5$ in Construction 4, as well as the number of needs N_{seed} needed in the proof, the size $\mathsf{size}_{\mathsf{pk}}^{\mathsf{LESS}}$ of the public key, and the size $\mathsf{size}_{\mathsf{Sig}}^{\mathsf{LESS}}$ of the proof.

Conclusions

Throughout this thesis, we have considered some of the main aspects behind code-based digital signatures, such as their construction, cryptanalysis, and theoretical foundations. For each of these contexts, we asked ourselves several questions that we tried to answer.

We wondered whether it was possible to build a hash&sign digital signature scheme that was both secure and efficient, trying to answer this question by proposing a scheme that takes advantage of both QC and LDPC codes, and we reported a recent attack that unfortunately makes its use impractical.

Regarding cryptanalysis, we break the UF-CMA security of HWQCS, describing a theoretical attack, and implementing it concretely. We also tried to build a distinguisher for the codes used by Wave, the generalized normalized $(U, U + V)$ codes. We showed the limits of this approach and how the work done in this direction, although not sufficient to build an efficient distinguisher, can have value in its own right. In particular, we built new algorithms to estimate the weight distribution of a given linear code.

Regarding theoretical foundations, we considered the fixed-weight optimization under a security standpoint, proving that a fixed-weight repetition of a $(k_1, \ldots, k_\mu)$-special-sound $(2\mu + 1)$-round public-coin interactive proof enjoys knowledge soundness. With this, we provided a direct, tight results on the security of the underlying interactive proofs of many recent signatures, such as CROSS. Finally, in contexts in which it is necessary to produce proofs of knowledge, but an online-extractable transform is required (and therefore Fiat-Shamir is not a viable option), we asked ourselves whether it was possible to construct a transform that took into account a priori the unbalanced challenges. We proposed a new transform, showing how it behaves compared to those already known.

In an attempt to answer the above questions, we ended up with even more doubts than we started with. The problem of building code-based hash&sign signature schemes is still open, despite some promising schemes recently proposed, such as [DAST19], and our attempt to build a distinguisher has left more questions than answers. Similarly, having proven knowledge soundness for the interactive protocol underlying CROSS is a good first step in proving the formal security of this scheme, but there is more work to be done in this direction. Ultimately, the proposed transformation represents an initial, exploratory effort that requires further investigation and refinement to reach a satisfactory state of maturity.

> *"As long as a branch of science offers an abundance of problems, so long it is alive; a lack of problems foreshadows extinction or the cessation of independent development."*
>
> *David Hilbert, 1900*

Bibliography

[AB09] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

[AB98] A. Ashikhmin and A. Barg. "Minimal vectors in linear codes". In: *IEEE Transactions on Information Theory* 44.5 (1998), pp. 2010–2017.

[Abd+02] M. Abdalla, J. H. An, M. Bellare, and C. Namprempre. "From Identification to Signatures via the Fiat-Shamir Transform: Minimizing Assumptions for Security and Forward-Security". In: *Advances in Cryptology — EUROCRYPT 2002*. Ed. by L. R. Knudsen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 418–433.

[ACK21a] T. Attema, R. Cramer, and L. Kohl. "A compressed Σ-protocol theory for lattices". In: *Annual International Cryptology Conference*. Springer. 2021, pp. 549–579.

[ACK21b] T. Attema, R. Cramer, and L. Kohl. "A Compressed Σ-Protocol Theory for Lattices". In: *CRYPTO 2021, Part II*. Ed. by T. Malkin and C. Peikert. Vol. 12826. LNCS. Virtual Event: Springer, Heidelberg, Aug. 2021, pp. 549–579. DOI: 10.1007/978-3-030-84245-1_19.

[AF22a] T. Attema and S. Fehr. "Parallel Repetition of $(k_1, \ldots, k_\mu)$-Special-Sound Multi-round Interactive Proofs". In: *CRYPTO 2022, Part I*. Ed. by Y. Dodis and T. Shrimpton. Vol. 13507. LNCS. Springer, Heidelberg, Aug. 2022, pp. 415–443. DOI: 10.1007/978-3-031-15802-5_15.

[AF22b] T. Attema and S. Fehr. "Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs". In: *Annual International Cryptology Conference*. Springer. 2022, pp. 415–443.

[AFK22a] T. Attema, S. Fehr, and M. Klooß. "Fiat-shamir transformation of multi-round interactive proofs". In: *Theory of Cryptography Conference*. Springer. 2022, pp. 113–142.

[AFK22b] T. Attema, S. Fehr, and M. Klooß. "Fiat-Shamir Transformation of Multi-round Interactive Proofs". In: *TCC 2022, Part I*. Ed. by E. Kiltz and V. Vaikuntanathan. Vol. 13747. LNCS. Springer, Heidelberg, Nov. 2022, pp. 113–142. DOI: 10.1007/978-3-031-22318-1_5.

[AFR23] T. Attema, S. Fehr, and N. Resch. "Generalized Special-Sound Interactive Proofs and Their Knowledge Soundness". In: *TCC 2023, Part III*. Ed. by G. N. Rothblum and H. Wee. Vol. 14371. LNCS. Springer, Heidelberg, Nov.–Dec. 2023, pp. 424–454. DOI: 10.1007/978-3-031-48621-0_15.

[AFS05] D. Augot, M. Finiasz, and N. Sendrier. "A family of fast syndrome based cryptographic hash functions". In: *International Conference on Cryptology in Malaysia*. Springer. 2005, pp. 64–83.

[AHU74] A. Aho, J. Hopcroft, and J. Ullman. "The analysis and design of computer algorithms". In: *Add1˜"-son-Wes——-fey-˜ Reading, Mass* (1974).

[Ala+20] N. Alamati, L. De Feo, H. Montgomery, and S. Patranabis. "Cryptographic group actions and applications". In: *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*. Springer. 2020, pp. 411–439.

[Ara+17] N. Aragon, P. S. Barreto, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, S. Gueron, T. Guneysu, C. A. Melchor, et al. "BIKE: bit flipping key encapsulation". In: (2017).

[Ara+21] N. Aragon, M. Baldi, J.-C. Deneuville, K. Khathuria, E. Persichetti, and P. Santini. "Cryptanalysis of a code-based full-time signature". In: *Designs, Codes and Cryptography* 89 (2021), pp. 2097–2112.

[Bal+13] M. Baldi, M. Bianchi, F. Chiaraluce, J. Rosenthal, and D. Schipani. "Using LDGM codes and sparse syndromes to achieve digital signatures". In: *Post-Quantum Cryptography: 5th International Workshop, PQCrypto 2013, Limoges, France, June 4-7, 2013. Proceedings 5*. Springer. 2013, pp. 1–15.

[Bal14] M. Baldi. *QC-LDPC code-based cryptography*. Springer Science & Business, 2014.

[Bal+19] M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini. "LEDAcrypt: QC-LDPC code-based cryptosystems with bounded decryption failure rate". In: *Code-Based Cryptography Workshop*. Springer. 2019, pp. 11–43.

[Bal+20] M. Baldi, M. Battaglioni, F. Chiaraluce, A.-L. Horlemann-Trautmann, E. Persichetti, P. Santini, and V. Weger. "A new path to code-based signatures via identification schemes with restricted errors". In: *arXiv preprint arXiv:2008.06403* (2020).

[Bal+21a] M. Baldi, M. Battaglioni, F. Chiaraluce, A.-L. Horlemann-Trautmann, E. Persichetti, P. Santini, and V. Weger. *A New Path to Code-based Signatures via Identification Schemes with Restricted Errors*. 2021. eprint: 2008.06403.

[Bal+21b] M. Baldi, J.-C. Deneuville, E. Persichetti, and P. Santini. "Cryptanalysis of a Code-Based Signature scheme based on the Schnorr-Lyubashevsky framework". In: *IEEE Communications Letters* 25.9 (2021), pp. 2829–2833.

[Bal+23a] M. Baldi, A. Barenghi, S. Bitzer, P. Karl, F. Manganiello, A. Pavoni, G. Pelosi, P. Santini, J. Schupp, F. Slaughter, et al. "CROSS-Codes and Restricted Objects Signature Scheme". In: *2024 Spring Eastern Sectional Meeting*. AMS. 2023.

[Bal+23b] M. Baldi et al. *CROSS — Codes and Restricted Objects Signature Scheme*. Tech. rep. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures. National Institute of Standards and Technology, 2023.

[Bar+21a] A. Barenghi, J.-F. Biasse, E. Persichetti, and P. Santini. "LESS-FM: Fine-Tuning Signatures from the Code Equivalence Problem". In: *Post-Quantum Cryptography - 12th International Workshop, PQCrypto 2021*. Ed. by J. H. Cheon and J.-P. Tillich. Springer, Heidelberg, 2021, pp. 23–43. DOI: 10.1007/978-3-030-81293-5_2.

[Bar+21b] J. Bariffi, H. Bartz, G. Liva, and J. Rosenthal. "On the properties of error patterns in the constant Lee weight channel". In: *arXiv preprint arXiv:2110.01878* (2021).

[Bar94] S. Barg. "Some new NP-complete coding problems". In: *Problemy Peredachi Informatsii* 30.3 (1994), pp. 23–28.

[Bat+24] M. Battagliola, R. Longo, F. Pintore, E. Signorini, and G. Tognolini. "Security of Fixed-Weight Repetitions of Special-Sound Multi-Round Proofs". In: *Cryptology ePrint Archive* (2024).

[BE21] E. Bellini and A. Esser. *Syndrome Decoding Estimator*. 2021. URL: https://github.com/Crypto-TII/syndrome_decoding_estimator.

[Bec+12] A. Becker, A. Joux, A. May, and A. Meurer. "Decoding random binary linear codes in $2^{n/20}$: How 1+1 = 0 improves information set decoding". In: *Advances in Cryptology–EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings 31*. Springer. 2012, pp. 520–536.

[Beu+23] W. Beullens, S. Dobson, S. Katsumata, Y.-F. Lai, and F. Pintore. "Group signatures and more from isogenies and lattices: generic, simple, and efficient". In: *DCC* 91.6 (2023), pp. 2141–2200. DOI: 10.1007/s10623-023-01192-x.

[BF02] A. Barg and G. D. Forney. "Random codes: Minimum distances and error exponents". In: *IEEE Transactions on Information Theory* 48.9 (2002), pp. 2568–2573.

[Bia+20] J.-F. Biasse, G. Micheli, E. Persichetti, and P. Santini. "LESS is more: code-based signatures without syndromes". In: *Progress in Cryptology-AFRICACRYPT 2020: 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20–22, 2020, Proceedings 12*. Springer. 2020, pp. 45–65.

[BKP20] W. Beullens, S. Katsumata, and F. Pintore. "Calamari and Falafl: Logarithmic (Linkable) Ring Signatures from Isogenies and Lattices". In: *ASIACRYPT 2020, Part II*. Ed. by S. Moriai and H. Wang. Vol. 12492. LNCS. Springer, Heidelberg, Dec. 2020, pp. 464–492. DOI: 10.1007/978-3-030-64834-3_16.

[BKT99] A. Barg, E. Krouk, and H. C. van Tilborg. "On the complexity of minimum distance decoding of long linear codes". In: *IEEE Transactions on Information Theory* 45.5 (1999), pp. 1392–1405.

[BKW22] J. Bariffi, K. Khathuria, and V. Weger. "Information set decoding for Lee-metric codes using restricted balls". In: *Code-Based Cryptography Workshop*. Springer. 2022, pp. 110–136.

[BMVT78] E. Berlekamp, R. McEliece, and H. Van Tilborg. "On the inherent intractability of certain coding problems (corresp.)" In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 384–386.

[Bon+11] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, and M. Zhandry. "Random oracles in a quantum world". In: *Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings 17*. Springer. 2011, pp. 41–69.

[Bor+23] G. Borin, E. Persichetti, P. Santini, F. Pintore, and K. Reijnders. "A Guide to the Design of Digital Signatures based on Cryptographic Group Actions". In: *Cryptology ePrint Archive* (2023).

[Bou07] I Bouyukliev. "About the code equivalence". In: *Advances in Coding Theory and Cryptography* 3 (2007), pp. 126–151.

[BR93] M. Bellare and P. Rogaway. "Random oracles are practical: A paradigm for designing efficient protocols". In: *Proceedings of the 1st ACM Conference on Computer and Communications Security*. 1993, pp. 62–73.

[BS20] D. Boneh and V. Shoup. "A graduate course in applied cryptography". In: *Draft 0.5* (2020).

[Cas+18] W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes. "CSIDH: an efficient post-quantum commutative group action". In: *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III 24*. Springer. 2018, pp. 395–427.

[CFS01] N. T. Courtois, M. Finiasz, and N. Sendrier. "How to achieve a McEliece-based digital signature scheme". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2001, pp. 157–174.

[Che+16] M.-S. Chen, A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe. "From 5-Pass MQ-Based Identification to MQ-Based Signatures". In: *ASIACRYPT 2016, Part II*. Ed. by J. H. Cheon and T. Takagi. Vol. 10032. LNCS. Springer, Heidelberg, Dec. 2016, pp. 135–165. DOI: 10.1007/978-3-662-53890-6_5.

[Cho+23a] T. Chou, R. Niederhagen, E. Persichetti, T. H. Randrianarisoa, K. Reijnders, S. Samardjiska, and M. Trimoska. "Take your MEDS: digital signatures from matrix code equivalence". In: *International conference on cryptology in Africa*. Springer. 2023, pp. 28–52.

[Cho+23b] T. Chou, R. Niederhagen, E. Persichetti, T. H. Randrianarisoa, K. Reijnders, S. Samardjiska, and M. Trimoska. "Take Your MEDS: Digital Signatures from Matrix Code Equivalence". In: *AFRICACRYPT 23*. Ed. by N. El Mrabet, L. De Feo, and S. Duquesne. Vol. 14064. LNCS. Springer Nature, July 2023, pp. 28–52. DOI: 10.1007/978-3-031-37679-5_2.

[CL04] J. Camenisch and A. Lysyanskaya. "Signature schemes and anonymous credentials from bilinear maps". In: *Annual international cryptology conference*. Springer. 2004, pp. 56–72.

[CL24] Y.-H. Chen and Y. Lindell. "Optimizing and Implementing Fischlin's Transform for UC-Secure Zero-Knowledge". In: *Cryptology ePrint Archive* (2024).

[Coo00] S. Cook. "The P versus NP problem". In: *Clay Mathematics Institute* 2.6 (2000), p. 3.

[Coo23] S. A. Cook. "The complexity of theorem-proving procedures". In: *Logic, automata, and computational complexity: The works of Stephen A. Cook*. 2023, pp. 143–152.

[CVA10] P.-L. Cayrel, P. Véron, and S. M. E. Y. Alaoui. "A Zero-Knowledge Identification Scheme Based on the q-ary Syndrome Decoding Problem." In: *Selected Areas in Cryptography*. Vol. 6544. Springer. 2010, pp. 171–186.

[Dal07] L. Dallot. "Towards a concrete security proof of Courtois, Finiasz and Sendrier signature scheme". In: *Western European Workshop on Research in Cryptology*. Springer. 2007, pp. 65–77.

[Dam02] I. Damgård. "On Σ-protocols". In: *Lecture Notes, University of Aarhus, Department for Computer Science* 84 (2002).

[Dam89] I. B. Damgård. "A design principle for hash functions". In: *Conference on the Theory and Application of Cryptology*. Springer. 1989, pp. 416–427.

[DAST17] T. Debris-Alazard, N. Sendrier, and J.-P. Tillich. "The problem with the SURF scheme". In: *arXiv preprint arXiv:1706.08065* (2017).

[DAST19] T. Debris-Alazard, N. Sendrier, and J.-P. Tillich. "Wave: A new family of trapdoor one-way preimage sampleable functions based on codes". In: *Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part I*. Springer. 2019, pp. 21–51.

[DAT18] T. Debris-Alazard and J.-P. Tillich. "Two attacks on rank metric code-based schemes: RankSign and an IBE scheme". In: *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part I 24*. Springer. 2018, pp. 62–92.

[Del78] P. Delsarte. "Bilinear forms over a finite field, with applications to coding theory". In: *Journal of combinatorial theory, Series A* 25.3 (1978), pp. 226–241.

[DFG19] L. De Feo and S. D. Galbraith. "SeaSign: compact isogeny signatures from class group actions". In: *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38*. Springer. 2019, pp. 759–789.

[DG20] J.-C. Deneuville and P. Gaborit. "Cryptanalysis of a code-based one-time signature". In: *Designs, Codes and Cryptography* 88.9 (2020), pp. 1857–1866.

[DH22] W. Diffie and M. E. Hellman. "New directions in cryptography". In: *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*. 2022, pp. 365–390.

[DMP21] G. D'Alconzo, A. Meneghetti, and P. Piasenti. "Security issues of CFS-like digital signature algorithms". In: *arXiv preprint arXiv:2112.00429* (2021).

[DW22] L. Ducas and W. van Woerden. "On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2022, pp. 643–673.

[Fau+13] J.-C. Faugere, V. Gauthier-Umana, A. Otmani, L. Perret, and J.-P. Tillich. "A distinguisher for high-rate McEliece cryptosystems". In: *IEEE Transactions on Information Theory* 59.10 (2013), pp. 6830–6844.

[Fis05] M. Fischlin. "Communication-efficient non-interactive proofs of knowledge with online extractors". In: *Annual International Cryptology Conference*. Springer. 2005, pp. 152–168.

[FJR22] T. Feneuil, A. Joux, and M. Rivain. "Syndrome Decoding in the Head: Shorter Signatures from Zero-Knowledge Proofs". In: *Cryptology ePrint Archive* (2022).

[FJR23] T. Feneuil, A. Joux, and M. Rivain. "Shared permutation for syndrome decoding: new zero-knowledge protocol and code-based signature". In: *Designs, Codes and Cryptography* 91.2 (2023), pp. 563–608.

[FRX+17] K Fukushima, P. S. Roy, R Xu, et al. "Random code-based signature scheme (RaCoSS)". In: *First Round Submission to the NIST Post-quantum Cryptography Call*. 2017.

[FS87a] A. Fiat and A. Shamir. "How To Prove Yourself: Practical Solutions to Identification and Signature Problems". In: *Advances in Cryptology — CRYPTO' 86*. Ed. by A. M. Odlyzko. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 186–194.

[FS87b] A. Fiat and A. Shamir. "How to prove yourself: Practical solutions to identification and signature problems". In: *Advances in Cryptology—CRYPTO'86: Proceedings 6*. Springer. 1987, pp. 186–194.

[FS87c] A. Fiat and A. Shamir. "How to Prove Yourself: Practical Solutions to Identification and Signature Problems". In: *CRYPTO'86*. Ed. by A. M. Odlyzko. Vol. 263. LNCS. Springer, Heidelberg, Aug. 1987, pp. 186–194. DOI: 10.1007/3-540-47721-7_12.

[Gab+14] P. Gaborit, O. Ruatta, J. Schrek, and G. Zémor. "RankSign: an efficient signature algorithm based on the rank metric". In: *International Workshop on Post-Quantum Cryptography*. Springer. 2014, pp. 88–107.

[Gab85] E. M. Gabidulin. "Theory of codes with maximum rank distance". In: *Problemy peredachi informatsii* 21.1 (1985), pp. 3–16.

[Gal62] R. Gallager. "Low-density parity-check codes". In: *IRE Transactions on information theory* 8.1 (1962), pp. 21–28.

[GG07] P. Gaborit and M. Girault. "Lightweight code-based identification and signature". In: *2007 IEEE International Symposium on Information Theory*. IEEE. 2007, pp. 191–195.

[GJ79a] M. R. Garey and D. S. Johnson. *Computers and intractability*. Vol. 174. freeman San Francisco, 1979.

[GJ79b] M. R. Gary and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. 1979.

[GMR19] S. Goldwasser, S. Micali, and C. Rackoff. "The knowledge complexity of interactive proof-systems". In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. 2019, pp. 203–225.

[GMR88] S. Goldwasser, S. Micali, and R. L. Rivest. "A digital signature scheme secure against adaptive chosen-message attacks". In: *SIAM Journal on computing* 17.2 (1988), pp. 281–308.

[GOT12] V. Gauthier, A. Otmani, and J.-P. Tillich. "A Distinguisher-based attack on a variant of McEliece's cryptosystem based on Reed-Solomon codes". In: *arXiv preprint arXiv:1204.6459* (2012).

[GPS22] S. Gueron, E. Persichetti, and P. Santini. "Designing a Practical Code-Based Signature Scheme from Zero-Knowledge Proofs with Trusted Setup". In: *Cryptography* 6.1 (2022), p. 5.

[GPV24] W. Ghantous, F. Pintore, and M. Veroni. "Efficiency of SIDH-based signatures (yes, SIDH)". In: *J. Math. Cryptol.* 18.1 (2024). DOI: 10.1515/JMC-2023-0023. URL: https://doi.org/10.1515/jmc-2023-0023.

[Ham50] R. W. Hamming. "Error detecting and error correcting codes". In: *The Bell system technical journal* 29.2 (1950), pp. 147–160.

[HL10] C. Hazay and Y. Lindell. *Efficient secure two-party protocols: Techniques and constructions*. Springer Science & Business Media, 2010.

[HMM05] M. Hirotomo, M. Mohri, and M. Morii. "A probabilistic computation method for the weight distribution of low-density parity-check codes". In: *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005.* 2005, pp. 2166–2170. DOI: 10.1109/ISIT.2005.1523730.

[HTW20] A.-L. Horlemann-Trautmann and V. Weger. "Information set decoding in the Lee metric with applications to cryptography". In: *Advances in Mathematics of Communications* 15.4 (2020), pp. 677–699.

[Hül+16] A. Hülsing, J. Rijneveld, S. Samardjiska, and P. Schwabe. "From 5-pass MQ-based identification to MQ-based signatures." In: *IACR Cryptol. ePrint Arch.* 2016 (2016), p. 708.

[HW24] F. Hörmann and W. van Woerden. "FuLeakage: Breaking FuLeeca by learning attacks". In: *Annual International Cryptology Conference*. Springer. 2024, pp. 253–286.

[Jab01] A. A. Jabri. "A statistical decoding algorithm for general linear block codes". In: *Cryptography and Coding: 8th IMA International Conference Cirencester, UK, December 17–19, 2001 Proceedings 8*. Springer. 2001, pp. 1–8.

[Ji+19] Z. Ji, Y. Qiao, F. Song, and A. Yun. "General linear group action on tensors: A candidate for post-quantum cryptography". In: *Theory of cryptography conference*. Springer. 2019, pp. 251–281.

[JJ02] T. Johansson and F. Jonsson. "On the complexity of some cryptographic problems based on the general decoding problem". In: *IEEE Transactions on Information Theory* 48.10 (2002), pp. 2669–2678.

[Kar10] R. M. Karp. *Reducibility among combinatorial problems*. Springer, 2010.

[Kar72] R. M. Karp. "Reducibility among combinatorial problems". In: *Complexity of computer computations*. Springer, 1972, pp. 85–103.

[Kar75] R. M. Karp. "On the computational complexity of combinatorial problems". In: *Networks* 5.1 (1975), pp. 45–68.

[Kim+22] J.-L. Kim, J. Hong, T. S. C. Lau, Y. Lim, and B.-S. Won. "REDOG and Its Performance Analysis". In: *Cryptology ePrint Archive* (2022).

[KKS97] G. Kabatianskii, E. Krouk, and B. Smeets. "A digital signature scheme based on random error-correcting codes". In: *IMA International Conference on Cryptography and Coding*. Springer. 1997, pp. 161–167.

[KMP16] E. Kiltz, D. Masny, and J. Pan. "Optimal security proofs for signatures from identification schemes". In: *Annual International Cryptology Conference*. Springer. 2016, pp. 33–61.

[Kna06] A. W. Knapp. *Basic algebra*. Springer Science & Business Media, 2006.

[Knu92] D. E. Knuth. "Two notes on notation". In: *American Mathematical Monthly* 99.5 (1992), pp. 403–422. DOI: 10.2307/2325085.

[KS22] Y. Kondi and A. Shelat. "Improved straight-line extraction in the random oracle model with applications to signature aggregation". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2022, pp. 279–309.

[KZ20] D. Kales and G. Zaverucha. "An attack on some signature schemes constructed from five-pass identification schemes". In: *International Conference on Cryptology and Network Security*. Springer. 2020, pp. 3–22.

[LB88] P. J. Lee and E. F. Brickell. "An observation on the security of McEliece's public-key cryptosystem". In: *Advances in Cryptology—EUROCRYPT'88: Workshop on the Theory and Application of Cryptographic Techniques Davos, Switzerland, May 25–27, 1988 Proceedings 7*. Springer. 1988, pp. 275–280.

[Lee58] C Lee. "Some properties of nonbinary error-correcting codes". In: *IRE Transactions on Information Theory* 4.2 (1958), pp. 77–82.

[Leo82] J. Leon. "Computing automorphism groups of error-correcting codes". In: *IEEE Transactions on Information Theory* 28.3 (1982), pp. 496–511.

[Leo88] J. Leon. "A probabilistic algorithm for computing minimum weights of large error-correcting codes". In: *IEEE Transactions on Information Theory* 34.5 (1988), pp. 1354–1359. DOI: 10.1109/18.21270.

[LN97] R. Lidl and H. Niederreiter. *Finite fields*. 20. Cambridge university press, 1997.

[LXY20] Z. Li, C. Xing, and S. L. Yeo. "A new code based signature scheme without trapdoors". In: *Cryptology ePrint Archive* (2020).

[Lyu09] V. Lyubashevsky. "Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2009, pp. 598–616.

[MC09] D. MacKay and E.-C. Codes. "David MacKay's Gallager code resources". In: (2009). URL: http://www.inference.phy.cam.ac.uk/mackay/CodesFiles.html.

[McE78] R. J. McEliece. "A public-key cryptosystem based on algebraic". In: *Coding Thv* 4244 (1978), pp. 114–116.

[Mel+18a] C. A. Melchor, N. Aragon, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, E. Persichetti, G. Zémor, and I. Bourges. "Hamming quasi-cyclic (HQC)". In: *NIST PQC Round* 2 (2018), pp. 4–13.

[Mel+18b] C. A. Melchor, N. Aragon, S. Bettaieb, L. Bidoux, O. Blazy, J.-C. Deneuville, P. Gaborit, E. Persichetti, G. Zémor, and I. Bourges. "Hamming quasi-cyclic (HQC)". In: *NIST PQC Round* 2 (2018), pp. 4–13.

[Mer89] R. C. Merkle. "One way hash functions and DES". In: *Conference on the Theory and Application of Cryptology*. Springer. 1989, pp. 428–446.

[MMT11] A. May, A. Meurer, and E. Thomae. "Decoding random linear codes in $\mathcal{O}(2^{0.054n})$". In: *Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings 17*. Springer. 2011, pp. 107–124.

[MN97] D. J. MacKay and R. M. Neal. "Near Shannon limit performance of low density parity check codes". In: *Electronics letters* 33.6 (1997), pp. 457–458.

[MS77] F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes*. Vol. 16. Elsevier, 1977.

[Nie86] H. Niederreiter. "Knapsack-type cryptosystems and algebraic coding theory". In: *Prob. Contr. Inform. Theory* 15.2 (1986), pp. 157–166.

[OT11] A. Otmani and J.-P. Tillich. "An efficient attack on all concrete KKS proposals". In: *International Workshop on Post-Quantum Cryptography*. Springer. 2011, pp. 98–116.

[Pas03] R. Pass. "On deniability in the common reference string and random oracle model". In: *Advances in Cryptology-CRYPTO 2003: 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003. Proceedings 23*. Springer. 2003, pp. 316–337.

[Per18] E. Persichetti. "Efficient one-time signatures from quasi-cyclic codes: A full treatment". In: *Cryptography* 2.4 (2018), p. 30.

[Pie67] J. Pierce. "Limit distribution of the minimum distance of random linear codes". In: *IEEE Transactions on Information Theory* 13.4 (1967), pp. 595–599.

[PMT22] C. Picozzi, A. Meneghetti, and G. Tognolini. "A Post-Quantum Digital Signature Scheme from QC-LDPC Codes". In: *Cryptology ePrint Archive* (2022).

[Pra62] E. Prange. "The use of information sets in decoding cyclic codes". In: *IRE Trans. Inf. Theory* 8.5 (1962), pp. 5–9.

[PS96] D. Pointcheval and J. Stern. "Security proofs for signature schemes". In: *International conference on the theory and applications of cryptographic techniques*. Springer. 1996, pp. 387–398.

[PT16] A. Phesso and J.-P. Tillich. "An efficient attack on a code-based signature scheme". In: *Post-Quantum Cryptography*. Springer. 2016, pp. 86–103.

[PT23] T. F. Prabowo and C. H. Tan. "Attack on a Code-Based Signature Scheme from QC-LDPC Codes". In: *Codes, Cryptology and Information Security: 4th International Conference, C2SI 2023, Rabat, Morocco, May 29–31, 2023, Proceedings*. Springer. 2023, pp. 136–149.

[PT24] A. Pellegrini and G. Tognolini. "Breaking HWQCS: a code-based signature scheme from high weight QC-LDPC codes". In: *Cryptology ePrint Archive* (2024).

[Rit+23] S. Ritterhoff, G. Maringer, S. Bitzer, V. Weger, P. Karl, T. Schamberger, J. Schupp, and A. Wachter-Zeh. "FuLeeca: A Lee-based Signature Scheme". In: *Cryptology ePrint Archive* (2023).

[Rot06] R. M. Roth. "Introduction to coding theory". In: *IET Communications* 47.18-19 (2006), p. 4.

[RSA83] R. L. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems". In: *Communications of the ACM* 26.1 (1983), pp. 96–99.

[RST23] L. Ran, S. Samardjiska, and M. Trimoska. "Algebraic Algorithm for the Alternating Trilinear Form Equivalence Problem". In: *Code-Based Cryptography - 11th International Workshop, CBCrypto 2023, Lyon, France, April 22-23, 2023, Revised Selected Papers*. Ed. by A. Esser and P. Santini. Vol. 14311. Lecture Notes in Computer Science. Springer, 2023, pp. 84–103. DOI: 10.1007/978-3-031-46495-9_5. URL: https://doi.org/10.1007/978-3-031-46495-9_5.

[RZW+17] F. Ren, D. Zheng, W. Wang, et al. "An Efficient Code Based Digital Signature Algorithm." In: *Int. J. Netw. Secur.* 19.6 (2017), pp. 1072–1079.

[SBC19] P. Santini, M. Baldi, and F. Chiaraluce. "Cryptanalysis of a one-time code-based digital signature scheme". In: *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2019, pp. 2594–2598.

[Sch91] C.-P. Schnorr. "Efficient signature generation by smart cards". In: *Journal of cryptology* 4 (1991), pp. 161–174.

[Sen11] N. Sendrier. "Decoding one out of many". In: *Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29–December 2, 2011. Proceedings 4*. Springer. 2011, pp. 51–67.

[Sho94] P. W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th annual symposium on foundations of computer science*. Ieee. 1994, pp. 124–134.

[Sip96] M. Sipser. "Introduction to the Theory of Computation". In: *ACM Sigact News* 27.1 (1996), pp. 27–29.

[Son+20] Y. Song, X. Huang, Y. Mu, W. Wu, and H. Wang. "A code-based signature scheme from the Lyubashevsky framework". In: *Theoretical Computer Science* 835 (2020), pp. 15–30.

[Ste01] J. Stern. "A new identification scheme based on syndrome decoding". In: *Advances in Cryptology—CRYPTO'93: 13th Annual International Cryptology Conference Santa Barbara, California, USA August 22–26, 1993 Proceedings*. Springer. 2001, pp. 13–21.

[Ste89] J. Stern. "A method for finding codewords of small weight". In: *Coding theory and applications* 388 (1989), pp. 106–113.

[Ste90] J. Stern. "An alternative to the Fiat-Shamir protocol". In: *Advances in Cryptology—EUROCRYPT'89: Workshop on the Theory and Application of Cryptographic Techniques Houthalen, Belgium, April 10–13, 1989 Proceedings 8*. Springer. 1990, pp. 173–180.

[Ste93] J. Stern. "A new identification scheme based on syndrome decoding". In: *Annual International Cryptology Conference*. Springer. 1993, pp. 13–21.

[Tan+22] G. Tang, D. H. Duong, A. Joux, T. Plantard, Y. Qiao, and W. Susilo. "Practical post-quantum signature schemes from isomorphism problems of trilinear forms". In: *Annual international conference on the theory and applications of cryptographic techniques*. Springer. 2022, pp. 582–612.

[TP24] C. H. Tan and T. F. Prabowo. "High Weight Code-Based Signature Scheme from QC-LDPC Codes". In: *Information Security and Cryptology – ICISC 2023*. Ed. by H. Seo and S. Kim. Singapore: Springer Nature Singapore, 2024, pp. 306–323.

[Unr15] D. Unruh. "Non-interactive zero-knowledge proofs in the quantum random oracle model". In: *Advances in Cryptology-EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II 34*. Springer. 2015, pp. 755–784.

[Var97] A. Vardy. "The intractability of computing the minimum distance of a code". In: *IEEE Transactions on Information Theory* 43.6 (1997), pp. 1757–1766.

[Vér97] P. Véron. "Improved identification schemes based on error-correcting codes". In: *Applicable Algebra in Engineering, Communication and Computing* 8 (1997), pp. 57–69.

[Weg+24] V. Weger, K. Khathuria, A.-L. Horlemann, M. Battaglioni, P. Santini, and E. Persichetti. "On the hardness of the Lee syndrome decoding problem". In: *Advances in Mathematics of Communications* 18.1 (2024), pp. 233–266.

[WGR22] V. Weger, N. Gassner, and J. Rosenthal. "A survey on code-based cryptography". In: *arXiv preprint arXiv:2201.07119* (2022).

[Xag18] K. Xagawa. "Practical attack on racoss-r". In: *Cryptology ePrint Archive* (2018).