Quaternion algebras and isogeny-based cryptography

Antonin Leroux

September 6, 2022

Contents

List of algorithms

We give below a list containing most of the algorithms used in this work, together with a quick description. When the full description is given in the manuscript, we also include the reference of the algorithm and the page where it can be found. In all the algorithms below, O_0 is the special extremal order of $B_{p;1}$ and N is a subset of N.

- ConnectingIdeal: takes two maximal orders $O_1; O_2$ in $B_{p;1}$ and computes the connecting ideal $I(O_1; O_2)$.
- Cornacchia(Algorithm 1, Page 64): takes two integers $q; m$ and nds a solution $t; x$ to $t^2 + qx^2 = m$ when it's possible.
- RepresentInteger$_N$ (Algorithm 2, Page 65): Finds of norm in N in the maximal extremal order of $B_{p;1}$.
- StrongApproximation$_N$ (Algorithm 3, Page 66): takes a prime N, two values $C; D$ 2 Z and nds of norm in N with $j(C + !D) + N$ $_1$ with $_1$ in the maximal extremal order of $B_{p;1}$.
- FullRepresentInteger$_N$ (Algorithm 4, Page 68): same as RepresentInteger but with a di erent output distribution.
- FullStrongApproximation$_N$ (Algorithm 5, Page 69): same as StrongApproximation but with a di erent output distribution.
- EquivalentPrimeIdeal(Algorithm 6, Page 71): takes an ideal I of maximal order in $B_{p;1}$ and outputs the equivalent ideal of smallest possible prime norm.
- RandomEquivalentPrimeIdeal: same as EquivalentPrimeIdeal but outputs a random choice among a set of ideals of small prime norm.
- IdealModConstraint: takes an ideal L of maximal order in $B_{p;1}$, an element 2 $B_{p;1}$ with gcd($n(\ ); n(L)^2$) = $n(L)$, and outputs (C : D) 2 P^1(Z=$n(L)$Z) s.t $j(C + !D)$ 2 L.
- EichlerModConstraint: takes an ideal L of maximal order in $B_{p;1}$, two elements $_1; _2$ 2 $B_{p;1}$ with gcd($n(\ ); n(L)$) = 1, and outputs (C : D) 2 P^1(Z=$n(L)$Z) s.t $_1 j(C + !D)$ $_2$ 2 Z + L.
- KLPT$_N$ (Algorithm 7, Page 73): takes an ideal I of the special extremal order and nds J I of norm contained in N.

- EichlerNorm$_N$ (Algorithm 8, Page 75): takes an ideal I and finds $\alpha \in \mathbb{Z} + I$ of norm contained in N.
- SpecialEichlerNorm$_N$ (Algorithm 9, Page 76): takes a maximal order $\mathcal{O}$ and K, an $\mathcal{O}$-ideal, and finds $\alpha \in \mathcal{O} \setminus (\mathbb{Z} + K)$.
- IdealEichlerNorm$_N$ (Algorithm 10, Page 77): takes two ideals I, J of the special extremal order of $B_{p,1}$, and finds $\alpha \in (\mathbb{Z} + I) \setminus J$ of norm contained in $n(J)N$.
- GenericKLPT$_N$ (Algorithm 11, Page 78): takes a maximal order $\mathcal{O}_1$ of $B_{p,1}$ and an $\mathcal{O}_1$-ideal I, and finds $J \sim I$ of norm in N.
- SuborderEichlerNorm$_N$ (Algorithm 12, Page 80): takes an integer D, an ideal I of the special extremal order of $B_{p,1}$, and computes $\alpha \in \mathbb{Z} + DI$ of norm contained in N.
- GeneratingFamily$_N$ (Algorithm 13, Page 81): takes an integer D and a maximal order $\mathcal{O}_1$ in $B_{p,1}$, and computes a generating family of $\mathbb{Z} + D\mathcal{O}_1$ whose elements have norm in N.
- IdealSuborderEichlerNorm$_N$ (Algorithm 14, Page 82): takes an integer D, two ideals I, J of the special extremal order of $B_{p,1}$, and finds $\alpha \in (\mathbb{Z} + DI) \setminus J$ of norm contained in $n(J)N$.
- Compression(Algorithm 17, Page 96): takes a curve E and an isogeny φ, and computes a string s representing the isogeny φ.
- Decompression(Algorithm 18, Page 96): takes a curve E and a compressed representation s, and computes the corresponding isogeny φ of domain E.
- IdealToKernel(Algorithm 19, Page 98): takes a curve E_1, I a cyclic ideal of maximal order in $B_{p,1}$, and computes a generator of of the cyclic subgroup $E_1[I]$.
- KernelToIdeal(Algorithm 20, Page 98): takes a point P in $E_1[D]$ and computes the kernel ideal corresponding to the group generated by P.
- IdealToIsogenyCoprime$_T$(Algorithm 21, Page 103): takes two equivalent left ideals J, K of $\mathcal{O}_0$, with J of norm dividing T^2 and K of norm ℓ, the corresponding isogeny φ_K, and computes the isogeny φ_J.
- IdealToIsogenySmallFromKLPT(Algorithm 22, Page 103): takes I a left $\mathcal{O}_0$-ideal of norm dividing $T^{2\ell^{2f+}}$, an $\mathcal{O}_0$-ideal in J containing I of norm dividing T^2, and an ideal $K \sim J$ of norm a power of ℓ, as well as φ_J and φ_K. Computes $\varphi = \varphi_2 \circ \varphi_1 : E_1 \to E_2$ of degree ℓ^{2f+} such that $\varphi_I = \widehat{\varphi} \circ \varphi_J$, $L \sim I$ of norm dividing T^2 and φ_L.
- IdealToIsogenyFromKLPT$_\ell$ (I, K, φ_K)(Algorithm 23, Page 104): takes I, a left $\mathcal{O}$-ideal, for a maximal order $\mathcal{O}$, of norm a power of ℓ, K a left $\mathcal{O}_0$-ideal and right $\mathcal{O}$-ideal of norm ℓ, the corresponding φ_K, and computes φ_I.
- IdealToIsogenySmallFromEichler$_{\ell^f}$ $(\mathcal{O}, I, J, \varphi_J, P)$(Algorithm 24, Page 105): takes I a left $\mathcal{O}$-ideal of norm ℓ^f, an $(\mathcal{O}_0, \mathcal{O})$-ideal J of norm in ℓ and $\varphi_J : E_0 \to E$ the corresponding isogeny, the generator $P \in E[\ell^f]$ of $\ker \varphi_K$ s.t $\hat{}_J = \varphi_{K^0} \circ \varphi_K$. Computes φ_I of degree ℓ^f.

- IdealToIsogenyFromEichler$_{\cdot}$ $(I; J; \varphi_J)$(Algorithm 25, Page 107): takes I a left O-ideal of norm ℓ^e with $e = fv$, an $(O_0; O)$-ideal J of norm ℓ and $\varphi_J : E_0 \to E$ the corresponding isogeny. Computes φ_I of degree ℓ^e.

- EndomorphismEvaluation$_{\cdot f}$ (Algorithm 26, Page 108): takes two isogenies $\varphi_1; \varphi_2 : E \to E'$, scalars $C; D$, the trace $t = \mathrm{tr}(\hat{\varphi}_2 \varphi_1)$ and a point P of order ℓ^f. Computes $[C]P + [D]\hat{\varphi}_2 \varphi_1(P)$.

- IdealVeri cation(Algorithm 27, Page 110): takes $x \in \mathbb{N} \setminus S(p)^2$ and I an ideal of $B_{p;1}$ and outputs a bit indicating if $x \in L_{\mathrm{isog}}$.

- IdealEvaluation(Algorithm 28, Page 111): takes I an O_0-ideal of $B_{p;1}$ and $P \in E_0(\overline{\mathbb{F}_q})$ of order coprime to $D = n(I)$, and computes $\varphi_I(P)$.

- IdealToSuborder(Algorithm 29, Page 113): takes I an integral ideal of maximal orders inside $B_{p;1}$ of norm ℓ, and computes endomorphisms $\varphi_i : E_2 \to E_2$ such that $\theta : \mathrm{End}(E_2) \to O_R(I)$ sends $\varphi_1; \ldots; \varphi_n$ to a generating family $\gamma_1; \ldots; \gamma_n$ for $\mathbb{Z} + DO_L(I)$.

- SuborderVeri cation(Algorithm 30, Page 115): takes $x \in \mathbb{P} \setminus S(p)^2$ and a suborder witness, and outputs a bit indicating if $x \in L_{\mathrm{p\ isog}}$.

- CheckTrace$_M$ (Algorithm 31, Page 116): takes $\gamma_1; \ldots; \gamma_n$, n endomorphisms of E and n elements of $B_{p;1}$ $\omega_1; \ldots; \omega_n$. Outputs a bit b equal to 1 if and only if $\mathrm{tr}(\gamma_i) = \mathrm{tr}(\omega_i) \mod M$ for all $i \in [1; n]$.

- SuborderEvaluation(Algorithm 32, Page 118): takes a suborder witness for $(D; E_1; E_2) \in L_{\mathrm{p\ isog}}$ and an ideal J of norm coprime to ℓ, and computes $\varphi(E_1[J])$.

- SigningKLPT$_{2^e}$(Algorithm 33, Page 127): takes I a left O_0-ideal and right O_1-ideal of prime norm N inert in O, and I, a left O_1-ideal. Computes $J \sim I$ of norm 2^e.

- RandomEquivalentEichlerIdeal(Algorithm 34, Page 128): takes I a left O_1-ideal, and outputs $K \sim I$ of norm coprime to N .

- InverseTrapdoor(Algorithm 35, Page 159): takes $j_T \in J_{T;p}$, a trapdoor T and c, outputs $m \in [\ (D)]$ such that $f_{j_T}^{D;N}(m) = \mathrm{c.}\ \ell$

- SetaQuadraticOrderGen(Algorithm 36, Page 163): takes $D; N$ as above. Let S be the product of primes dividing D, nds $(d; e)$ such that $\frac{N^2 e^2 - d^2}{D^2} < 0$ is a quadratic non-residue modulo every prime dividing D and is a quadratic non-residue modulo p.

- SetaCurveGen(Algorithm 37, Page 165): takes a prime p, an integer N, a quadratic order O, a bound B, a length ℓ. Computes a uniformly random curve $E_{j_T} \in E_O(p)$, a basis $P_{j_T}; Q_{j_T}$ of $E_{j_T}[N]$, and $\theta(P_{j_T})$; $\theta(Q_{j_T})$ with $\theta \in \mathrm{End}(E_{j_T})$ such that $\mathbb{Z}[\theta] = O$.

- pSIDHKeyGen(Algorithm 38, Page 181): takes a prime number $D \leq p$, outputs the pSIDH public key pk = $E; \theta$ and the pSIDH secret key sk = I where θ is a suborder witness and I an ideal witness for $(D; E_0; E) \in L_{\mathrm{p\ isog}}$.

- pSIDHKeyExchange(Algorithm 39, Page 182): takes I an ideal of degree D and a prime $D' \not\equiv D; p$. A curve E' and a suborder witness , computes a j-invariant.

Introduction

Cryptography is the science of secrecy. The need to protect a message against eavesdroppers can be traced back at least to antiquity, where we can find numerous examples of ciphers.

Despite this long history, the discipline has changed radically since the time when messages were written with quills and sent on parchments. The revolution of cryptography emerged in the second part of the 20th century with the development of computers. In their seminal work "New directions for cryptography", Diffie and Hellman [DH76] led the way into a new era for security. Based on hard computational problems, public key cryptography enables a set of parties to secure their communication without any prior agreement of a shared secret. In public key cryptography, number theory has found a surprisingly practical field of application by supplying the aforementioned hard computational problems, the two most prominent ones by far being the integer factorization problem and the discrete logarithm problem (in either a finite field or the group of points of an elliptic curve).

Today we are on the brink of a new revolution in cryptography with the arrival of quantum computers. While this new generation of computers promises interesting applications across computing, to cryptographers they mainly represent a threat. The problem for cryptography was uncovered by Peter Shor who showed in 1994 [Sho97] that a quantum computer could solve the integer factorization and discrete logarithm problems in polynomial time, thus breaking any hope of building security upon their hardness in a world where attackers can use quantum computers.

Fortunately, these machines have their limitations and there are some problems that are believed to be hard even for a quantum computer. This is the founding principle of post-quantum cryptography, which studies the protocols that can be built on quantum-hard problems (in contrast, we call *classical* any cryptography that is not targeting security against a quantum attacker). To encourage the effort of the cryptographic community, the National Institute of Standards and Technology (NIST) launched in 2016 a competition to determine the most promising candidates. This competition is now at its third round and big families of proposals have started to emerge.

This work will focus exclusively on one of those families: protocols based on isogenies of elliptic curves. Isogenies are algebraic morphisms between elliptic curves. There is currently one isogeny-based candidate and it has been classified by NIST as one of the alternate candidates in the third round (meaning that it is a promising scheme that needs more understanding). Cryptographers were no stranger to isogenies due to the important part played by elliptic curves in classical cryptography. Isogeny-based cryptography is thus merely the new chapter

of the common story between number theory and public key cryptography.

A link between elliptic curves, isogenies, and quaternion algebras, another family of objects inherited from number theory, was discovered by the German mathematician Max Deuring (see [Deu41] for instance). Throughout this work, we call this link *the Deuring correspondence* and we will study it in Chapter 2. Due to the rise of interest in isogenies, the fascinating mathematical results from Deuring have started to gain the attention of cryptographers. In this thesis, we strive to take this adventure further by studying in detail the objects of the Deuring correspondence and their applications to cryptography.

The hard problem that is the cornerstone of isogeny-based cryptography is the following: given two isogenous elliptic curves, find an isogeny between them. The hardness of this problem depends on a number of parameters but it is generally believed to have at least sub-exponential complexity even for quantum computers.

The first appearances of isogenies in cryptography date back to the beginning of the 21st century with Teske's trapdoor system [Tes06] and Galbraith, Hess, and Smart's use of isogenies in attacks on protocols based on elliptic curves [GHS02]. The first cryptographic key exchange based on isogenies was independently proposed by Couveignes [Cou06] and Rostovtsev and Stolbunov [RS06], and is called CRS. The security of this protocol is based on the isogeny problem in the setting of ordinary curves. While the isogeny problem is hard for ordinary curves, it is harder for supersingular curves (sub-exponential vs. exponential quantum complexity) and it is with supersingular curves that isogeny-based cryptography really started to become relevant. The first proposal based on isogenies of supersingular curve is the CGL hash function from Charles, Goren and Lauter [CLG09]. Then came the SIDH (Supersingular Isogeny Diffie–Hellman) two-parties key exchange from De Feo and Jao in 2011 [JD11]. This protocol recreates the usual Diffie–Hellman mechanism by decomposing an isogeny of composite degree in two different ways (one for each participant). With very compact public keys and reasonable efficiency, SIDH is the first real practical application of isogeny-based cryptography and SIKE, the only isogeny-based candidate at the NIST competition, is a simple variant of SIDH.

A few years after that, Castryck, Lange, Martindale, Panny and Renes revisited the CRS key exchange in the context of supersingular curves. Their idea is to restrict to the set of supersingular curves defined over $\mathbb{F}_p$ in order to get a structure similar to CRS, while using the nice properties of supersingular curves (in particular the control we have on the group of points defined over $\mathbb{F}_p$) to get a much more efficient scheme than CRS. While less computationally efficient than SIDH, CSIDH has smaller keys for the same security level and, contrary to SIDH, their validity can be easily verified, making CSIDH the first post-quantum Non Interactive Key Exchange (NIKE), i.e., a key exchange with static public keys. More generally, CRS and CSIDH are two instantiations of a cryptographic group action (see for instance [AFMP20]), an abstract framework from which protocols can be built in a black-box manner.

Recently, the increased interest in post-quantum cryptography has pushed isogeny-based cryptography in new directions. During the last five years, a lot of progress has been made, both in the theoretical and algorithmic study of isogenies and in the conception of protocols based on isogenies. Among those, we are interested in and have contributed to the works making use of the Deuring

correspondence.

The first step of this journey was taken by Kohel, Lauter, Petit and Tignol in 2014 with their paper [KLPT14] where they studied the quaternion path problem, the analogue of the isogeny problem under the Deuring correspondence. They showed that this problem could be solved in heuristic polynomial time by solving some norm equations in ideals of the quaternion algebra ramified at p and ∞. Their efficient algorithm, called KLPT, led to a deeper understanding of the different problems in isogeny-based cryptography, for instance a proof that the problem of computing the endomorphism ring of a supersingular curve is equivalent to the supersingular isogeny problem [EHL+18, Wes22].

The protocols presented in this work continue this line of work by exploring cryptographic applications of the Deuring correspondence. Our results were obtained by understanding how to use the endomorphism rings of supersingular curves to make the Deuring correspondence effective. Concretely, this means being able to translate objects such as elliptic curves, isogenies and endomorphisms to their counterparts in the world of quaternions: orders and ideals. Once on the quaternion side, we can apply KLPT and its variants to solve the task at hand, before going back to elliptic curves with the inverse translation.

This generic mechanism was used constructively in the GPS signature scheme by Galbraith, Petit, and Silva [GPS17]. This signature is derived with the Fiat–Shamir transform from the repetition of an identification scheme whose main step is to compute an isogeny between two curves from the knowledge of their endomorphism ring.

One of our most important contributions is to improve upon the GPS construction, both in theory and practice, by building the SQISign signature scheme presented in Chapter 5. In that regard, our contributions can be divided in two parts.

The first one is to generalize the KLPT algorithm to Eichler orders and their ideals in order to obtain more generic isogenies through the Deuring correspondence. Using that, we can construct an identification scheme whose main operation remains the computation of an isogeny using endomorphism rings, but with a much larger challenge space than the one of GPS. Thus, we have to repeat this sub-protocol only once to reach a good security level and we derive with Fiat–Shamir a signature protocol that is a lot more compact than GPS.

Our second contribution is to make the translation between isogenies and quaternion ideals a lot more efficient using various tricks and the choice of a very specific prime p. In particular, we improve the method outlined in [EHL+18]. Contrary to what is proposed in GPS, which is to perform the translation with a one-shot algorithm, the idea introduced in [EHL+18] is to decompose the target isogeny in smaller bits and treat each of these parts with the one-shot algorithm from [GPS17] in an efficient manner.

This idea introduces a new problem: evaluating endomorphisms of an arbitrary elliptic curve of known endomorphism ring. We propose two different ways of solving this issue, either by using another isogeny, of degree coprime to the one we try to translate, or by computing a well-chosen endomorphism (also of degree coprime to the isogeny to be translated). The latter appears faster than the former because finding endomorphisms is slightly easier than finding isogenies.

The two contributions we outlined above are respectively part of two broader axes of research on the algorithmic Deuring correspondence we will explore

throughout this thesis. The first one is treated in Chapter 3 and is related to norm equations in lattices of the quaternion algebra ramified at p and ∞. Kohel et al. looked at maximal orders and their ideals for the KLPT algorithm [KLPT14] and we have continued their work by exploring the rich zoology of quaternion orders. In particular, we have looked at orders whose Gorenstein closures are Eichler orders. We were able to show that the sub-algorithms introduced in [KLPT14] could be used to perform all the required computations. We obtained these extensions of KLPT by exploring how the Deuring correspondence can be extended to Eichler orders of level N by considering curves augmented with cyclic subgroups of order N.

Our second important direction targets what we call representations of isogenies, that is ways of computing and manipulating isogenies, and it is the focus of Chapter 4. Overall, we use three distinct isogeny representations: the *kernel representation* that is the standard way of computing isogenies from the Velu formulas [Vel71], the *ideal representation* derived from the Deuring correspondence and the *suborder representation*, a new representation made of a suborder of the endomorphism ring. We have contributed to the algorithmic computation of these three representations in one way or another. For the kernel representation, we have introduced a new algorithm to compute the Velu formulas by applying a baby-step giant-step type of optimization to the computation of the kernel polynomial by using the biquadratic relations on x-coordinates of elliptic curve points. This idea gives a quadratic speed-up on the classical approach.

Our improvements to the ideal representation computation come from our efforts to improve the ideal to isogeny translation algorithm used in GPS and SQISign. The ideal representation is generally much more efficient than the one based on the Velu formulas as the complexities of the algorithms involved are polynomial in the degree (contrary to the Velu formulas that are exponential in the worst case), an idea first introduced by Bröker, Charles and Lauter [BCL08]. While not providing any asymptotic gains, our algorithms give ways of handling the ideal isogeny representation much more efficiently in practice.

Following this idea, we have introduced another isogeny representation called the suborder representation. This new representation makes use of the fact that the existence of an isogeny $\varphi' : E \to E'$ of degree D implies an embedding of the order $\mathbb{Z} + D\,\mathrm{End}(E)$ inside $\mathrm{End}(E')$. We can find nice generating endomorphisms of this suborder using our extended norm equation algorithms, thus providing a way to manipulate this representation and perform some interesting computations. We believe that the ideal representation of the isogeny φ' is hard to recover from the suborder representation when D is prime, and we formulate this idea as a new computational assumption. Under this assumption, the ideal and suborder representations are not equivalent and we can use this gap to build new cryptography.

As a consequence of these various algorithmic results, we propose the first isogeny-based protocols that are explicitly using isogenies of large prime degree. The SQISign signature scheme introduced in Chapter 5 uses large prime degree isogenies as private keys and we introduce in Chapter 7 a new key exchange called pSIDH based on the suborder representation and whose security relies on our new assumption. This key exchange is basically a variant of SIDH with large prime degrees (instead of smooth ones) and the public keys are made of the suborder representation of the secret key isogenies. Isogenies of large prime degree are interesting for two main reasons: the quadratic speed-up of the

meet-in-the-middle attack in the smooth-degree case cannot be applied to prime degrees, and the complexity of the computation becomes exponential when the endomorphism ring is unknown which makes for a good trapdoor mechanism.

In 2017, before we had introduced the suborder representation, Christophe Petit [Pet17] showed how to recover e ciently the kernel representation of the isogeny from the suborder representation when D is smooth. He showed how to exploit this mechanism to produce what has become known as *torsion point attacks* against the SIDH scheme. These attacks provide speed-ups against various degenerate variants of SIDH but not against the initial protocol. In fact, only a partial suborder representation, in the form of the embedding of a speci c quadratic order, is required to apply Petit's attack. When the endomorphism ring of the domain curve contains this special quadratic order, we get a so-called \backdoor curve". The embedding of this quadratic order constitutes a trapdoor allowing the owner of this knowledge to compute a partial suborder representation and thus recover the isogeny (only when D is smooth). The only additional information required to apply this mechanism is the action of the isogeny to be recovered on some torsion points.

Another of our contributions is to use this idea to de ne a trapdoor one-way function. We build a public key encryption scheme called Seta from this primitive, it is presented in Chapter 6. The key generation requires nding one supersingular curve with the special quadratic order embedded in its endomorphism ring. To solve that task, we use again the Deuring correspondence. First, we nd the solutions over the quaternions, before translating it into the isogeny setting.

To study the security of the Seta encryption scheme we introduced a new family of computational problems called the \Uber Isogeny Problems". This family is parameterized by quadratic orders and, depending on the type of order used, we have showed a connection of the Uber Isogeny Problem with problems involved in SIDH, CSIDH, Seta and even the generic isogeny problem. The complexity of this problem basically depends on the number of orientable curves, i.e., the number of supersingular curves that admit the embedding of a given quadratic order inside their endomorphism ring. By considering the quaternion orders generated by two such distinct embeddings, and generalizing some ideas due to Kaneko [Kan89], we have managed to prove the rst generic and e ective lower bound on the number of orientable supersingular curves. This gives the rst lower-bound on the complexity of the brute-force attack against the generic Uber Isogeny Problem.

Outline of the thesis

This thesis is divided in two main parts. In Part I, we study the Deuring correspondence from the theoretic and algorithmic standpoint, while in Part II, we introduce some protocols built upon the results of the rst part.

Part I is made of four chapters:

Chapter 1 introduces all the necessary preliminaries on elliptic curves, isogenies and quaternions. We list all the important results on these objects for the rest of this thesis.

Chapter 2 is dedicated to the Deuring correspondence. We provide a rather complete overview of all the aspects of this correspondence, and focus on the subsidiary results that will prove useful later on.

Chapter 3 presents a family of algorithms to solve norm equations inside lattices of a quaternion algebra. These algorithms will be the basis of all the powerful applications that we will build on top of the Deuring correspondence.

Chapter 4 presents various e cient ways to compute, evaluate and manipulate isogenies. The rst one is *the kernel representation* and is closely related to the basic de nition of isogenies. The second one, *the ideal representation,* is naturally derived from the results of the Deuring correspondence. We call the third one *the suborder representation*, and it is one of our contributions. It is also based on the Deuring correspondence, but in a less obvious manner.

Part II is made of three chapters:

Chapter 5 presents SQISign, a new signature scheme based on a proof of knowledge of an endomorphism ring and on the algorithms for the ideal representation.

Chapter 6 presents Seta, a new public key encryption scheme based on a trapdoor one-way function. The key generation uses the tools of the ideal representation, while the encryption/decryption mechanism is based on more traditional objects of isogeny-based cryptography. In the end of this chapter, we introduce a new isogeny assumption that is tied to various problems arising in isogeny-based cryptography, in the rst attempt to provide a uni ed vision on the various families of schemes constituting isogeny-based cryptography.

Chapter 7 introduces some applications of the suborder representation introduced in Chapter 4. In particular, we present pSIDH, a new non-interactive key exchange based on the problem of computing the ideal representation of an isogeny from its suborder representation.

Part I

Theory and algorithms

Chapter 1

Preliminaries

In this chapter, we introduce all the necessary mathematical background for the rest of this work. We try to cover all the relevant material in a brief but rather complete overview. Our account will only graze the surface of the deep mathematical subjects upon which we stand, but it should be enough for our purpose. More often than not, we will favor concrete examples and definitions to abstract and generic ones. The introduction of the mathematical background for this work is divided in three main parts. The first and second will be treated in Section 1.1 and Section 1.2. The last and most important part will be treated on its own in Chapter 2. We start with Section 1.0.1 to introduce some useful notations.

1.0.1 Notations

Below, we describe a list of generic notations and notions that we will use throughout this work.

- We write $O(\text{poly}(x))$ for a quantity asymptotically upper bounded by a polynomial in x.
- For a field k, we write $\overline{k}$, an algebraic closure of k.
- For a field k, we write $\mathrm{P}^n(k)$ for the projective space of dimension n. It is made of the equivalence classes in $k^{n+1} \setminus \{0\}^{n+1}$ for the equivalence relation $\sim_k$ defined as $(x_1 : \dots : x_{n+1}) \sim_k (y_1 : \dots : y_{n+1})$ when $x_i = \lambda y_i$ for all $i \in [1; n+1]$ for some $\lambda \in k^*$.
- The set of prime integers is written $\mathbb{P}$ and the set of prime divisors of an integer n is written $\mathbb{P}_n$.
- A B-smooth integer n has all its prime factors smaller than B. The smallest possible B is called the smoothness bound of n. When $B \leq n$, we will sometimes informally say that n is \smooth". We usually mean that the smoothness bound B of n is in $O(\text{poly}(\log(n)))$ in that case.
- A B-powersmooth number is a number n such that all the prime power divisors of n are smaller than B. As for smooth number, a number is powersmooth when $B \leq n$.

- A near-prime integer n is a prime multiplied by a smooth number.
- A negligible function $f : \mathbb{Z}_{>0} \to \mathbb{R}_{>0}$ is a function whose growth is bounded by $O(x^{-n})$ for all $n > 0$. In the analysis of a probabilistic algorithm, we say that an event happens with *overwhelming probability* if its probability of failure is a negligible function of the length of the input.
- We will write $\mathrm{CRT}_{N_1,\dots,N_n}(x_1,\dots,x_n)$ for the Chinese Remainder algorithm, that takes $x_i \in \mathbb{Z}/N_i\mathbb{Z}$ for $1 \le i \le n$ with all N_i pairwise coprime, and returns $z \in \mathbb{Z}/(\prod_{i=1}^{n} N_i)\mathbb{Z}$ with $z = x_i \bmod N_i$ for all $i \in [1,n]$.
- We write $v_\ell(n)$ for the ℓ-adic valuation of an integer n.
- For any prime $\ell \in \mathbb{P}$, $\mathbb{Q}_\ell$ is the field of ℓ-adic numbers and $\mathbb{Z}_\ell$ is the ring of ℓ-adic integers.

The local-global principle. We present below a concept that will follow us throughout this work. What we will call henceforth the *local-global principle* is the idea that some objects can be understood by looking at what happens *locally*, i.e from the ℓ-adic point of view at every prime ℓ. This was the approach favored by Deuring in [Deu41] to prove all the results upon which most of our content is built, and we include it in this work in the hope that it will help provide some insights on the mathematical concepts involved. Our account of the *local-global* interpretation is mostly inspired by the 13th chapter in the book Elliptic Functions of Serge Lang [Lan87] (Lang claims to follow closely the work of Deuring in this chapter) and some content spread throughout the book Quaternion Algebras by John Voight [Voi21], which are both excellent references to get a better understanding on the objects treated in this work.

1.1 Elliptic curves and isogenies

In this section, we introduce the main characters of our story: isogenies. However, as in any good story, we need to set the landscape before the hero's entrance. In Section 1.1.1, we introduce another family of main characters: elliptic curves, and postpone the formal presentation of isogenies to Section 1.1.2. A good and accessible reference for the content of this section is the book "The Arithmetic of Elliptic Curves" by J. Silverman [Sil86].

1.1.1 Elliptic curves

Elliptic curves have been studied from various mathematical perspectives over the long course of their mathematical story, but we are going to stick to the most basic and concrete definitions in this work. We give below, without all the necessary background, a generic and abstract definition of elliptic curves. We will explain what an elliptic curve concretely is right after that.

Definition 1.1.1. An elliptic curve over a field k is a smooth projective curve of genus 1 with a distinguished base point defined over k.

For simplicity, we will focus henceforth on fields of characteristic $\neq 2, 3$. What is hiding behind Definition 1.1.1 is that elliptic curves are varieties of

the projective plane, i.e homogeneous polynomials in three variables, that we usually call the *equation of the curve*. The points of an elliptic curve E over k, denoted by $E(k)$, are simply the solutions to this equation in k. As we are going to see, the equations de ning elliptic curves have a very speci c \shape". We use the terminology *elliptic curve model* to talk about a family of polynomials de ning elliptic curves.

Elliptic curve models. The most classical family are *Weierstrass curves*. The *Weierstrass model* is often considered the canonical way of representing elliptic curves because every elliptic curve has one.

De nition 1.1.2. A *short Weierstrass curve* E over k is de ned by the equation

$$E : Y^2Z = X^3 + aXZ^2 + bZ^3 \tag{1.1.1}$$

with $a; b \in k$ such that $= 16(4a^3 + 27b^2) \neq 0$. The point $(0 : 1 : 0)$ is the *point at in nity* of E and is denoted by 0_E.

Even if the Weierstrass model is often used for theoretical results, it is not the most e cient model for concrete algorithms, in the sense that the operations are more expensive. In cryptography, the *Montgomery* or *Edwards* models are typical choices for fast implementation. Not every elliptic curve has a rational Montgomery or an Edward model, and this is why their main interest is for practical use. We present below *Montgomery curves* that became popular after P. Montgomery introduced them in [Mon87] to speed-up Lenstra's ECM integer factorization method. Most of the implementations of algorithms introduced in this thesis are done in the Montgomery model. A good review on Montgomery curves and the related e cient algorithms can be found at [CS18].

De nition 1.1.3. A *Montgomery curve* E over k is de ned by the equation

$$E : BY^2Z = X(X^2 + AXZ + Z^2) \tag{1.1.2}$$

with $A; B \in k$ such that $B \neq 0$ and $A^2 \neq 4$. The point $(0 : 1 : 0)$ is the *point at in nity* of E.

For both models, each point of $P \in E(k)$ can be represented by its *projective coordinates* $(X_P : Y_P : Z_P) \in \mathrm{P}^2(k)$. Under the mapping $x = X/Z$ and $y = Y/Z$, the Weierstrass and Montgomery equations can be simpli ed to what we call the a ne model of E:

$$E : y^2 = x^3 + ax + b \qquad E : By^2 = x(x^2 + Ax + 1) \tag{1.1.3}$$

$(x_P; y_P)$ are the $(x; y)$-coordinates of the point P. This simpli cation is well-de ned for all the point of $E(k)$, except for 0_E that cannot be represented in $(x; y)$-coordinates. In that case, 0_E is considered as an abstract point of E. We de ne implicitly the coordinate projection functions $x; y$ for any elliptic curve E, and will sometimes write $x(P); y(P)$.

The group law. If elliptic curves have garnered so much interest from mathematicians and, more recently, from cryptographers, it is partly because their equation hides a deeper mathematical structure. Indeed, for any eld k, it can

be shown that $E(k)$ is an abelian group of neutral element 0_E under a composition law that we write and that can be de ned by rational maps on the projective coordinates. We will give concrete examples later in this document (see for instance Section 4.1.3 where we introduce the biquadratic relations between $x(P); x(Q)$ and $x(P \quad Q); x(P \quad Q)$).

De nition 1.1.4. For any $` \ 2 \ Z$, we write $[`] : E(k) \ ! \ E(k)$ for the *scalar-multiplication*-by-$`$ morphism. The kernel of $`$ over $\overline{k}$ is called the $`$*-torsion subgroup* and is written $E[`]$.

There exists a family of polynomials called the *division polynomials* and often denoted by $_{E;`}$ that can be used to de ne the rational maps for the $[`]$ map. $_{E;`}$ is basically the polynomial whose roots are the $x$-coordinates of the points in $E[`]$ (the exact de nition is slightly more complicated, but it is not important for us). They can be computed using a simple recurrence relation if needed.

The structure of $E[`]$ is easy to determine in most cases. By the chinese remainder theorem, it su ces to consider the structure for prime-power $`$ to infer the result for any $` \ 2 \ Z$.

Proposition 1.1.5. *Let $`$ be the power of a prime.*

$$E[`] = \begin{cases} Z=`Z \quad Z=`Z & \text{if } \mathrm{char}(k) = 0 \text{ or } \gcd(`; \mathrm{char}(k)) = 1; \\ Z=`Z \text{ or } f0_E g & \text{otherwise:} \end{cases} \tag{1.1.4}$$

When $\mathrm{char}(k) = p$, the two structures $Z=pZ$ and $f0_E g$ are possible for $E[p]$. The former case is called *ordinary* and the latter *supersingular*. This distinction has important consequences, and we will see equivalent de nitions later in Section 1.1.2.

From the structure on the $`$-torsion given by Proposition 1.1.5, we can deduce various things. For instance, we can count the number of cyclic subgroups of order $`$, something that will prove useful.

Proposition 1.1.6. *For a prime $` \ 6= \mathrm{char}(k)$, there are $` + 1$ cyclic subgroups of order $`$. Given a basis $P; Q$ of $E[`]$, these subgroups are in bijection with* $\mathrm{P}^1(Z=`Z)$ *under the map*

$$(i; j) \ 7! \ h[i]P + [j]Q i \tag{1.1.5}$$

The Tate module. This paragraph is the rst occurrence of the local-global principle that was teased in Section 1.0.1. Let $`$ be a prime number. The idea is to construct, for the groups $E[`^n]$, the analogue of what the $`$-adic integers $Z_`$ are for the groups $Z=`^n Z$ under the inverse limit $\lim_n$.

De nition 1.1.7. The ($`$-adic) *Tate module* of a curve E is

$$T_`(E) = \lim_n E[`^n] \tag{1.1.6}$$

w.r.t the natural maps $[`] : E[`^{n+1}] \ ! \ E[`^n]$.

More concretely $T_`(E)$ is made of all the in nite sequences of points $(P_1; : : :)$ with each $P_i \ 2 \ E[`^i]$ and $[`]P_{i+1} = P_i$. The Tate module is a useful abstraction because it allows us to manipulate all the $`^n$-torsion subgroups at once. The natural corollary of Proposition 1.1.5 is Proposition 1.1.8.

Proposition 1.1.8. *Let ℓ be the power of a prime.*

$$T_\ell(E) = \begin{cases} \mathbb{Z}_\ell \quad \mathbb{Z}_\ell & \text{if } \mathrm{char}(k) = 0 \text{ or } \gcd(\ell, \mathrm{char}(k)) = 1; \\ \mathbb{Z}_\ell \text{ or } \{0_E\} & \text{otherwise.} \end{cases} \tag{1.1.7}$$

Isomorphism class and the j-invariant. One might wonder if all elliptic curves over k are different, or if some of them are equivalent in some sense. Two curves E, E' are said to be isomorphic over a field k if there exists, between them, a rational isomorphism whose coefficients are defined over k. This creates an equivalence relation on the set of elliptic curves over k. Throughout this work, we will often consider isomorphism classes of elliptic curves over $\overline{k}$ and more often than not when saying "an elliptic curve" we will mean "an isomorphism class of elliptic curves".

The *j-invariant* gives a canonical representative for classes of isomorphic curves. For Weierstrass and Montgomery curves, this quantity is equal respectively to

$$j(E) = 1728\frac{4a^3}{4a^3 + 27b^2} \quad \text{and} \quad j(E) = \frac{256(A^2 - 3)^3}{A^2 - 4} \tag{1.1.8}$$

Proposition 1.1.9. *Two curves E, E' defined over k are isomorphic over $\overline{k}$ if and only if $j(E) = j(E')$.*

Remark 1.1.10. Isogenies, that we introduce next in Section 1.1.2, will provide another equivalence relation on elliptic curves of which the isomorphic relation is a specialization, as isomorphisms are in fact isogenies of degree 1.

1.1.2 Isogenies

With isogenies of elliptic curves, we go beyond the consideration of the set of elliptic curves defined over k as a collection of individual objects.

Definition 1.1.11. An *isogeny* between two elliptic curves E, E' is a non-constant morphism of projective varieties mapping 0_E to $0_{E'}$.

Similarly to isomorphisms, we say that an isogeny is defined over k if the isogeny can be defined as a rational map whose coefficients are in k. Two curves E, E' are said to be *isogenous* if there exists an isogeny $\phi : E \to E'$.

For us, the most important feature of an isogeny is going to be its *degree*. We give below the formal definition of the degree, but we will stick to cases where it can be computed quite simply.

Definition 1.1.12. An isogeny $\phi : E \to E'$ induces an embedding $\phi : f \mapsto f \circ \phi$ of the function field $k(E')$ in $k(E)$. The *degree* of ϕ is the degree of the extension $k(E)/\phi(k(E'))$. If this extension is *separable* (resp. inseparable, resp. *purely inseparable*), ϕ is said to be *separable* (resp. inseparable, resp. *purely inseparable*). In the separable case, we have $\deg \phi = \#\ker \phi$.

In characteristic $p > 0$, the reality behind Definition 1.1.12 is rather simple. Any isogeny of degree coprime to p is separable, and the only possible purely inseparable isogenies have degree equal to a power of p. Every isogeny can be seen as the composition of a purely inseparable isogeny and a separable one. For

separable isogenies, when $\ker \phi$ is cyclic, we say that ϕ is cyclic. Every separable isogeny is the composition of a scalar multiplication and a cyclic isogeny. In this work, we will mostly deal with *cyclic*, *separable* isogenies. The set of cyclic separable isogenies (up to post-composition with isomorphisms) of degree D whose domain has j-invariant equal to j is denoted by $\mathrm{isog}_k(D; j)$. We stress that the elements of $\mathrm{isog}_k(D; j)$ are classes of isogenies, but we often call them simply \isogenies". When the eld of de nition is clear from the context, we omit k and simply write $\mathrm{isog}(D; j)$. An isogeny of degree D is sometimes called a D-isogeny and the curves E and E' are said to be D-isogenous.

Kernels and duals. For separable isogenies, there is a strong link between isogenies and their kernels. For any elliptic curve E, there is a 1-to-1 correspondence between separable isogenies $\phi : E \to E'$ and nite subgroups of E (de ned over $\bar{k}$) by associating the subgroup $\ker \phi$ to the isogeny ϕ. Conversely, we write

$$\phi : E \to E/G$$

for the isogeny whose kernel is G. With Proposition 1.1.6, this correspondence between isogenies and kernels proves that the number of ℓ-isogenies is exactly $\ell + 1$ for any prime $\ell \neq p$. Throughout this work, we will often identify an isogeny with its kernel, and we will expand upon this relation in Section 4.1.

De nition 1.1.13. For any isogeny $\phi : E_1 \to E_2$ of degree d, there exists a unique isogeny $\hat{\phi} : E_2 \to E_1$ of the same degree satisfying $\phi \circ \hat{\phi} = [d]$. We call $\hat{\phi}$, *the dual* of ϕ. When d is coprime to p, we have $\ker \hat{\phi} = \phi(E_1[d])$.

It can be easily veri ed that the dual map, sending ϕ to $\hat{\phi}$, is an involution. The existence of the dual proves that the relation of being isogenous is actually an equivalence relation. The curves within a same isogeny class share important properties (see Proposition 1.1.20 for instance). The supersingular curves (see De nition 1.1.22) that will be our main object of interest are all in the same isogeny class.

1.2.1 The structure of the morphisms. With isogenies, we get a collection of morphisms. Next, we see that we have several families of these morphisms to which we can give a richer structure.

De nition 1.1.14. Let $E; E'$ be two elliptic curves de ned over k. The set of isogenies de ned over k between E and E' (completed with the trivial isogeny $P \mapsto 0_{E'}$) is written $\mathrm{Hom}_k(E; E')$. With the addition law $\phi + \psi : P \mapsto \phi(P) \oplus \psi(P)$, $\mathrm{Hom}(E; E')$ is a $\mathbb{Z}$-module.

De nition 1.1.15. An endomorphism is an isogeny from a curve to itself. We write $\mathrm{End}_k(E) = \mathrm{Hom}_k(E; E)$ for the set of endomorphisms. $\mathrm{End}_k(E)$ is a ring with the addition from De nition 1.1.14 and the composition operation $\circ$.

De nition 1.1.16. Automorphisms are endomorphisms of degree 1. We write $\mathrm{Aut}_k(E)$ for the group (for the composition) of automorphisms de ned over k.

Remark 1.1.17. The map $m \mapsto [m]$ yields an embedding $\mathbb{Z} \hookrightarrow \mathrm{End}_k(E)$ for any k and elliptic curve $E(k)$. For a generic eld k, the embedding of $\mathbb{Z}$ inside $\mathrm{End}_k(E)$ is the only information that we have on $\mathrm{End}_k(E)$. For nite elds, the story is more complex, as we are going to see.

When the field k is not specified in Hom, End, Aut, we consider the sets of morphisms defined over $\bar{k}$. Together with the degree map, the sets $\mathrm{Hom}(E, E')$ have a structure of quadratic module. The analysis of the quadratic modules obtained in this manner can be found in David Kohel's thesis [Koh96].

Definition 1.1.18. The *endomorphism algebra* A_E of a curve E is $\mathrm{End}(E) \otimes \mathbb{Q}$.

The study of endomorphism algebras is the first step toward a deeper understanding of the morphism modules of elliptic curves. For instance, it can be proven that two isogenous curves have the same endomorphism algebra. These objects are at the heart of Proposition 1.1.21 below.

1.1.3 Elliptic curves and isogenies over finite fields.

For all the remainder of this thesis, we fix a prime $p > 3$ and take $k = \mathbb{F}_q$ where q is a power of p. In that setting, $E(k)$ is a finite group, and we can bound the order with the Hasse bound.

Proposition 1.1.19. $|\#E(\mathbb{F}_q) - q - 1| \leq 2\sqrt{q}$.

We have also a strong relation between order and isogenies as illustrated by Proposition 1.1.20.

Proposition 1.1.20. *Two elliptic curves E and E' defined over $\mathbb{F}_q$ satisfy $\#E(\mathbb{F}_q) = \#E'(\mathbb{F}_q)$ if and only if E and E' are isogenous over $\mathbb{F}_q$.*

The Frobenius morphisms. The p^r-power Frobenius is

$$\pi_r : (x, y) \mapsto (x^{p^r}, y^{p^r}), \tag{1.1.9}$$

it is a morphism of degree p^r. For any elliptic curve E, we have $\pi_r : E \to E^{p^r}$ with E^{p^r} defined from the equation of E by putting the coefficients to the power p^r. Thus, $E^{p^r} = E$ when E is defined over $\mathbb{F}_{p^r}$. When $q > p^r$, the r-power Frobenius is a non-trivial isogeny of E and when E is defined over $\mathbb{F}_{p^r}$, π_r is a non-trivial element of $\mathrm{End}_{\mathbb{F}_{p^r}}(E)$. Hence, over finite fields, we almost always get non-trivial endomorphism rings (i.e., bigger than $\mathbb{Z}$). Sometimes $\mathbb{Z}[\pi]$ is the whole structure of the endomorphism ring, and sometimes not.

Proposition 1.1.21 lists all the possible structures for $\mathrm{End}(E)$. The objects of Proposition 1.1.21 will be introduced in Section 1.2.2 and we will provide more insights on Proposition 1.1.21 in Chapter 2.

Proposition 1.1.21. *For E an elliptic curve defined over k of characteristic $p > 0$, $\mathrm{End}(E)$ is either:*

(i) A quadratic order inside the quadratic imaginary field A_E.

(ii) A maximal order inside A_E, the quaternion algebra ramified at p and ∞.

Supersingular curves. In this thesis, we will focus on curves matching with (ii) in Proposition 1.1.21. We call them *supersingular elliptic curves,* and they admit several equivalent definitions.

De nition 1.1.22. An elliptic curve E over a eld of characteristic $p > 0$ is *supersingular* (and *ordinary* otherwise) if one (all) of the equivalent following properties is satis ed:

(i) $T_p(E) = f0g$.

(ii) All the $^{\wedge}r$ are purely inseparable.

(iii) The map $[p]$ is purely inseparable and $j(E)\ 2\ \mathbb{F}_{p^2}$.

(iv) $\mathrm{End}(E)$ is a maximal order in a quaternion algebra.

We can specify a bit the number of points of an elliptic curve following [Wat69, Theorem 4.1].

Proposition 1.1.23. *If E is a supersingular elliptic curve over $\mathbb{F}_{p^2}$, then:*

- *k is odd and $\#E(\mathbb{F}_{p^k}) = p^k + 1$.*
- *k is even and $\#E(\mathbb{F}_{p^k}) = p^k + 1$, $\#E(\mathbb{F}_{p^k}) = p^k \quad p^{k=2} + 1$ or $\#E(\mathbb{F}_{p^k}) = (p^{k=2} \quad 1)^2$.*

Henceforth, when we take an elliptic curve E, it can be considered supersingular unless it is explicitly said otherwise. Following De nition 1.1.22, we can de ne $S(p)$ as the set of supersingular j-invariants. The fact that $S(p) \quad \mathbb{F}_{p^2}$ has several important consequences. First, it proves that every supersingular class has a compact representative (which is going to be important for the cryptographic applications). Second, it means that $S(p)$ is a nite set. In fact, there is an exact formula to compute its size that we write N_p.

Proposition 1.1.24.

$$N_p = \#S(p) = \left\lfloor \frac{p}{12} \right\rfloor + \begin{cases} 0 & \text{if } p = 1 \mod 12; \\ 1 & \text{if } p = 5 \mod 12; \\ 1 & \text{if } p = 7 \mod 12; \\ 2 & \text{if } p = 11 \mod 12; \end{cases} \tag{1.1.10}$$

When E is de ned over $\mathbb{F}_p$, the Frobenius is an endomorphism of E. When E is de ned purely over $\mathbb{F}_{p^2}$, we get that the Frobenius is an isogeny $\quad : E \ ! \ E^p$ where $j(E^p) = j(E)^p$. The two curves E and E^p share a powerful connection. For instance, we have that $\mathrm{End}(E) = \mathrm{End}(E^p)$. It follows from $S(p) \quad \mathbb{F}_{p^2}$, that exponentiation by p is an involution on $S(p)$. Sometimes, it will be useful for us to consider $S(p)=$ rather than $S(p)$.

One of the numerous interesting properties of supersingular elliptic curves is that they all lie in the same isogeny class. Moreover, we can restrict to $`^e$-isogenies and still get the same result. This is formalized in Proposition 1.1.25.

Proposition 1.1.25. *Let $E_1; E_2$ be two supersingular elliptic curves, and let $` \not\equiv p$ be a prime. There exists a cyclic separable $`^e$-isogeny $' : E_1 \ ! \ E_2$ for some exponent $e\ 2\ \mathbb{N}$.*

Remark 1.1.26. Proving Proposition 1.1.25 can be done by looking at the numbers represented by the quadratic module associated to $\mathrm{Hom}(E_1; E_2)$ with the

degree quadratic form. This is what is done, for instance, in David Kohel's thesis [Koh96, Chapter 7]. We have the estimate $e = O(\log(p))$ for the exponent in Proposition 1.1.25. This can be also interpreted in terms of diameter of the supersingular ℓ-isogeny graph that we will introduce next.

We define $\mathrm{isog}(D) = \bigcup_{j \in S(p)} \mathrm{isog}(D; j)$. The *language of isogenous curves* that we define below will be at the heart of our study in Chapter 4 where we are going to be interested in ways to prove membership to L_{isog}, i.e. show that two supersingular curves are D-isogenous.

Definition 1.1.27. The *language of isogenous supersingular curves* is

$$L_{\mathrm{isog}} = \{(D; E_1; E_2) \in \mathbb{N} \times S(p) \times S(p) \mid \exists \varphi : E_1 \to E_2 \in \mathrm{isog}(D)\}.$$

The graph of supersingular ℓ-isogenies. The graph structure induced by ℓ-isogenies on supersingular elliptic curves is a useful representation.

Definition 1.1.28. Let G_p^ℓ be the graph whose vertices are the supersingular j-invariants over $\mathbb{F}_{p^2}$ and whose edges are the ℓ-isogenies. We call G_p^ℓ the graph of supersingular ℓ-isogenies.

Since $S(p) \subset \mathbb{F}_{p^2}$, G_p^ℓ is *finite*. It is also *undirected* because each isogeny has a dual. Proposition 1.1.25 proves that G_p^ℓ is *fully connected,* and the number of cyclic ℓ-isogenies proves that it is $\ell + 1$-*regular*.

Proposition 1.1.29. *G_p^ℓ has the Ramanujan property, i.e., the second-largest eigenvalue in absolute value of its adjacency matrix is smaller than $2\sqrt{\ell}$.*

Proposition 1.1.29 was proven by Pizer in [Piz90], it is a consequence of the Riemann hypothesis for function fields proven by Deligne. We will give more details on that topic in Section 2.4.3. It implies that G_p^ℓ is an expander graph family as $p \to \infty$. Concretely, this means that it is *fast mixing*: at the end of a random walk of length $O(\log(\#G_p^\ell))$, we get a vertex distributed as a uniformly random vertex. This property is why expander graphs are known as good candidates to build hash functions. Charles, Goren and Lauter [CLG09] used G_p^ℓ to build a cryptographic hash function whose security is based on the hardness of finding a path between two vertex of G_p^ℓ. Their work opened the way to all the applications that we will present in Part II.

Isogeny decomposition and isogeny commutative diagrams. Any isogeny φ of degree $D = \prod_{i=1}^{n} d_i$ where all the d_i are (non-necessarily coprime) integers can be decomposed as $\varphi = \varphi_n \circ \cdots \circ \varphi_1$ where each φ_i has degree d_i. If we assume that the d_i are pairwise coprime, each reordering of the d_i will lead to a different set of isogenies. We illustrate this principle in Figure 1.1 for $n = 2$. There are two possibilities in that case. We will use abundantly the type of commutative diagrams depicted in Figure 1.1 throughout this thesis and call them *isogeny diagrams*. We also introduce specific notations for the isogenies involved in the two possible decomposition. Let $d_1; d_2$ be two coprime integers and φ a $d_1 d_2$-isogeny. We can write $\varphi = \psi_2 \circ \varphi_1 = \psi_1 \circ \varphi_2$ and it can be verified that $\ker \psi_1 = \varphi_2(\ker \varphi_2)$ and conversely for ψ_2. Alternatively, the same isogeny diagram can be defined from φ_1 and φ_2. In that case, we call ψ_1 the *push-forward* of φ_1 through φ_2 and use the notation $\psi_1 = [\varphi_2]_* \varphi_1$. We can also see φ_1 as the *pull-back* of ψ_1 by φ_2 (which, up to isomorphism, is the same as the push-forward through $\hat{\varphi}_2$) and we write it $\varphi_1 = [\varphi_2]^* \psi_1 = [\hat{\varphi}_2]_* \psi_1$.

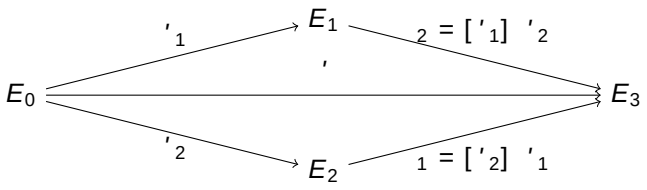


Figure 1.1: A commutative isogeny diagram.

1.2 Quadratic imaginary fields and quaternion algebras, orders and ideals

In this section, we study integral rings in imaginary Q-algebras and their ideals. We are mostly interested in quaternion algebras, but we start with the simpler case of quadratic imaginary fields, as a warm-up. They will also play their part in our story. Throughout our small introduction, we will try to highlight the similarities and differences between the two cases.

1.2.1 Quadratic imaginary fields

A good reference for the content of this section and the link with elliptic curves is the book "Primes of the form $x^2 + ny^2$" by D.A Cox [Cox11]. A quadratic imaginary field K is a quadratic extension of Q of the form $K = Q(\sqrt{d})$ where $d \in \mathbb{N}$ is a square-free integer.

Definition 1.2.1. The *discriminant* of K is

$$\mathrm{disc}(K) = \begin{cases} d & \text{when } d = 3 \mod 4 \\ 4d & \text{otherwise.} \end{cases} \tag{1.2.1}$$

We sometimes write Δ for disc(K). The discriminant of a quadratic imaginary field is called a *fundamental discriminant.* We write ω for the distinguished element in K satisfying $\omega^2 = \Delta$ and $(1, \omega)$ is a basis of K. The canonical involution sends an element $\alpha = a + \omega b$ to its conjugate $\bar{\alpha} = a - \omega b$. We can define the norm and trace of elements with $n(\alpha) = \alpha\bar{\alpha}$ and $\mathrm{tr}(\alpha) = \alpha + \bar{\alpha}$. The norm is multiplicative.

Quadratic orders and ideals. An element α is said to be *integral* when $n(\alpha), \mathrm{tr}(\alpha) \in \mathbb{Z}$. The study of integral elements is one of the original motivations behind the interest in imaginary quadratic fields, and it cannot be done without looking at orders and ideals inside K.

Definition 1.2.2. A *fractional ideal* of K is a Z-module of rank 2 contained in K. An *order* is a fractional ideal that is also a subring of K.

We denote by $n(I)$ the *norm* of I, defined as the Z-module generated by the norms of the elements of I. Given fractional ideals I and J, if $J \subset I$, then the index $[I : J]$ is defined to be the order of the finite quotient group I/J.

It can be verified that orders of K contain only integral elements. They can be written as $\mathcal{O} = \mathbb{Z}[\alpha]$ for some element α that we call the generator.

The *discriminant* of O is the discriminant of the minimal polynomial of , it is disc(O) = tr( )^2 4n( ). Sometimes we also use the notation ( ). A *maximal* order is one that is not contained in any superorder. In any quadratic imaginary elds, there is only one maximal order, it is called the *ring of integers* of K, and it contains all the integral elements of K. It can be described with the generic formula O_K = Z[(+ $\sqrt{p}$)=2] and it has discriminant equal to disc(K). Any other quadratic order in K can be decomposed as O = Z + f(O)O_K where f(O) 2 N is the *conductor* of O. The conductor is also equal to the index [O : O_K], and we have disc(O) = f(O)^{2}disc(K).

The *integral ideals* of K are the ideals of the orders of K (according to the usual de nition of an ideal in a ring). For *integral ideals*, we can de ne the *norm of an ideal* as the *gcd* of the norm of its elements. Quadratic orders are *Dedekind domains,* and so their ideals have a unique factorization into a product of prime ideals.

De nition 1.2.3. A prime number ` is *inert* in O when `O is a prime ideal. It is *split* if `O = l$\bar{l}$ and *rami ed* if `O = l^2 where l is a prime ideal of norm `.

Proposition 1.2.4 shows that we can determine if a prime is split, inert or rami ed with the *Kronecker symbol*, which agrees with the Legendre symbol when it is de ned.

Proposition 1.2.4.

$$\left(\frac{d}{\ell}\right) = \begin{cases} 1 & , \ \ell \text{ is split;} \\ 0 & , \ \ell \text{ is rami ed;} \\ -1 & , \ \ell \text{ is inert:} \end{cases} \tag{1.2.2}$$

An ideal of a quadratic order O is *principal* if it is generated by one element 2 O (and is written O).

The set of O-ideals quotiented by the principal ideals is an abelian group Cl(O) that we call the *class group of* O. The *class number* is h(O) = #Cl(O).

Under GRH, Littlewood [Lit28] proved the inequality

$$h(O) > \left(\frac{}{12e^{}} + o(1)\right)\frac{\sqrt{p}}{\log\log( )} \tag{1.2.3}$$

where is the Euler-Mascheroni constant with e = 0:56693.

1.2.2 Quaternion algebras

The book \Quaternion Algebras" by John Voight [Voi21] is the de nitive reference on the content of this section. The book of Marie-France Vigneras [Vig06] also covers most of the necessary notions. Quaternion algebras are basically two quadratic imaginary elds glued together in a non-commutative manner. In particular, there is an in nity of quadratic imaginary elds embedded inside any given quaternion algebra, and we will see that most of the notions and properties introduced in Section 1.2.1 can be translated one way or another in the setting of quaternions.

De nition 1.2.5. Let K be a eld. A K-algebra B is a quaternion algebra if there exists $a; b$ 2 K such that B is the K-vector space with basis $f1; i; j; kg$

satisfying the multiplication rules $i^2 = a$, $j^2 = b$, $k = ij = -ji$. In that case, we write $H(a;b)$ for B.

There are two possibilities for a quaternion algebra B over a field K. Either it is isomorphic to $M_2(K)$ or it is a *division ring* (sometimes called a *skew field*). We are mainly interested in quaternion algebras over Q, where the two elements $a;b$ are negative integers. In that case, the quaternion algebra $H(a;b)$ is said to be *definite*. Henceforth, we restrict to that case for ease of exposition. By the local-global principle, a quaternion algebra B over Q is uniquely identified by its completions $B_\ell = B \otimes_{\mathbb{Q}} \mathbb{Q}_\ell$ where ℓ ranges over the places of Q (so it is either a prime or the infinite place and $\mathbb{Q}_\ell = \mathbb{R}$). We say that B is *split* at ℓ if $B_\ell = M_2(\mathbb{Q}_\ell)$ and *ramified* otherwise. The product of the finite ramified places is called the *discriminant* of B. Given what we just explained, a complete study of a quaternion algebra over Q cannot really be led without understanding its completions over the local fields $\mathbb{Q}_\ell$. Quaternion algebras over local fields is a subject of its own, and we will only catch glances of it through the local-global interpretation.

In this manuscript, we will focus on the quaternion algebra $B_{p;\infty}$ ramified at $p \in \mathbb{P}$ and ∞ (where the prime p will coincide with the characteristic of the finite fields we consider for our elliptic curves and isogenies). The fact that the completion $B_{p;\infty} \otimes_{\mathbb{Q}} \mathbb{Q}_\ell$ is isomorphic to $M_2(\mathbb{Q}_\ell)$ for all the primes $\ell \neq p$ will have its importance in our local-global principle interpretation of the Deuring correspondence (see Chapter 2). For what follows, we fix our quaternion algebra as $B_{p;\infty}$, even if most of what we present remains true for a generic quaternion algebra under small modifications.

As in the quadratic case, every quaternion algebra has a canonical involution that sends an element $\alpha = a_1 + a_2 i + a_3 j + a_4 k$ to its conjugate $\bar{\alpha} = a_1 - a_2 i - a_3 j - a_4 k$. We can define norm and traces from the conjugation in the same manner as in the quadratic case. *Integral* elements have integral norm and trace. This norm is multiplicative, and the induced inner product

$$(\alpha ; \beta) \mapsto \frac{1}{2}(n(\alpha + \beta) - n(\alpha) - n(\beta)) = \frac{1}{2}\mathrm{tr}(\alpha\bar{\beta}) \tag{1.2.4}$$

is positive definite with orthogonal basis $\{1; i; j; k\}$. A consequence of that fact is that any element $\alpha \in B_{p;\infty}$ satisfies $\alpha^2 \leq 0$ and so $\mathbb{Q}(\alpha)$ is either Q or a quadratic imaginary field.

Quaternion orders and ideals. The definition of quaternion orders and ideals is the analogue of the definition in the quadratic case.

Definition 1.2.6. A *lattice* of $B_{p;\infty}$ is a Z-submodule of rank 4. An *order* is a lattice that is also a subring of $B_{p;\infty}$.

Following the parallelism between Definition 1.2.2 and Definition 1.2.6, we can define norms of ideals and indexes in the same manner as in Section 1.2.1. The discriminant of O is defined as $\mathrm{disc}(O) = \sqrt{|\det((\alpha_i, \alpha_j))|_{i,j \in \{1;2;3;4\}}}$ given a basis $\langle \alpha_1, \alpha_2, \alpha_3, \alpha_4 \rangle$ of O; $\mathrm{disc}(O) \in \mathbb{Z}$ and is independent of a choice of basis. An order is called *maximal* when it is not contained in any other strictly larger order. In $B_{p;\infty}$, the discriminant of the maximal orders is always equal to p. When $O' \subset O$ is a suborder of O, the index $[O : O']$ is defined as the ideal

generated by $\#O'/O$. Let us write $N = [O : O']$, then the discriminant of O' satisfies $\mathrm{disc}(O') = N^2\mathrm{disc}(O)$. We have $O' = O$, if and only if $N = 1$.

The local-global principle tells us that orders can be understood through their completions at the finite places $O_\ell = O \otimes \mathbb{Z}_\ell$ where $\ell \mid \mathrm{disc}(O)$. Each coprime ℓ can be considered independently.

The classification of quaternion orders. The similarities between the quadratic case and the quaternion case, stops at the basic definitions. We saw in the quadratic case that the hierarchy of quadratic orders is fairly simple to understand. The fact that we go from dimension 2 to 4 and that quaternion algebras are non-commutative explain that the story is much richer with quaternions.

A first example of this complexity is that we have an infinity of non-trivial isomorphisms indexed by the elements $\alpha \in B_{p,1}$ with $\alpha \notin \mathbb{Q}$ and $\alpha^2 \in \mathbb{Q}$. In fact, it can be shown that all isomorphisms are of this form (as a consequence of the Skolem-Noether Theorem). This means that two distinct orders can be isomorphic, and this is what motivates to introduce the notion of *types*.

Definition 1.2.7. The *type* of an order O written $\mathrm{Typ}\, O$ is the isomorphism class of O.

In the Deuring correspondence, we will often work with types of orders rather than orders directly. Apart from maximal orders, a class of order that will be of great importance to us are *Eichler orders*. They can be defined as the intersection of two maximal orders (not necessarily distinct). The *level* of an Eichler order is equal to $\mathrm{disc}(O)/p$ in our case. A maximal order is simply an Eichler order of level 1. When the level of an Eichler order is *square-free,* it is sometimes said to be *hereditary*. This distinction will not really matter to us, so we will not develop upon what makes hereditary orders different. When ℓ^e is the biggest power of ℓ dividing the level of O, we have that O_ℓ is an Eichler order of level ℓ^e in $\mathrm{M}_2(\mathbb{Q}_\ell)$ and we know that

$$O_\ell = \begin{pmatrix} \mathbb{Z}_\ell & \mathbb{Z}_\ell \\ \ell^e\mathbb{Z}_\ell & \mathbb{Z}_\ell \end{pmatrix} \tag{1.2.5}$$

Eichler orders have been studied extensively in the literature on quaternion algebras [Eic38, Piz80].

There is an analogue notion of conductor for quaternion orders, but it is not tied to maximal orders as in the quadratic case.

Proposition 1.2.8. *[Voi21, Proposition 24.2.15] Every quaternion order O admits the unique decomposition $\mathbb{Z} + f(O)\,\mathrm{Gor}\, O$ where $f(O) \in \mathbb{N}$ and $\mathrm{Gor}\, O$ is another quaternion order.*

Definition 1.2.9. For every order O, we call *conductor*, or *Brandt Invariant*, the integer $f(O)$ from Proposition 1.2.8. The order $\mathrm{Gor}\, O$ from the same result is called the *Gorenstein saturation* of O.

For this work, it will be sufficient to define *Gorenstein orders* as the orders O with $f(O) = 1$. There is a more generic definition using the coideal, but we will not use it here. As the name and definition suggest, $\mathrm{Gor}\, O$ is always a Gorenstein order.

An order is *Bass* when all its superorders are Gorenstein. Equivalently, Bass orders of $B_{p;\infty}$ are the orders containing a maximal order of a quadratic imaginary field (this was originally the definition of *basic* orders, but the two notions were proven equivalent by Chari, Smertnig, and Voight in [CSV21]).

We have a chain of proper implication between all these notions of quaternion orders

$$\text{maximal} \Rightarrow \text{Eichler} \Rightarrow \text{Bass} \Rightarrow \text{Gorenstein}.$$

The Eichler symbol. A very useful tool to understand the structure of a quaternion order using the local-global principle is the *Eichler symbol*. It was introduced by Eichler in [Eic36]. Let us write $\mathbb{F}_\ell$ for the residue field of $\mathbb{Q}_\ell$ and J for the Jacobson radical of O_ℓ. Then, we define the Eichler symbol as follows:

$$\left(\frac{O}{\ell}\right) = \begin{cases} ? & \text{if } O_\ell/J = \mathrm{M}_2(\mathbb{F}_\ell); \\ 1 & \text{if } O_\ell/J = \mathbb{F}_\ell \times \mathbb{F}_\ell; \\ 0 & \text{if } O_\ell/J = \mathbb{F}_\ell; \\ -1 & \text{if } O_\ell/J \text{ is a quadratic extension of } \mathbb{F}_\ell. \end{cases} \tag{1.2.6}$$

The Eichler symbol can be seen as a generalization of the Kronecker symbol, as becomes explicit with the reinterpretation presented as Proposition 1.2.10. We recall that for any α, we write $\Delta(\alpha) = \mathrm{disc}(\mathbb{Z}[\alpha]) = \mathrm{tr}(\alpha)^2 - 4n(\alpha)$.

Proposition 1.2.10. *$\left(\frac{O}{\ell}\right) = \varepsilon$ for $\varepsilon \in \{1, 0, -1\}$ if and only if $\left(\frac{\Delta(\alpha)}{\ell}\right)$ takes all the values $0, \varepsilon$ when α ranges over all the elements of O.*

Definition 1.2.11.

$$\left(\frac{O}{\ell}\right) = \begin{cases} 1 & \text{, } \ell \text{ is } \textit{residually split} \text{ in } O; \\ 0 & \text{, } \ell \text{ is } \textit{residually ramified} \text{ in } O; \\ -1 & \text{, } \ell \text{ is } \textit{residually inert} \text{ in } O. \end{cases} \tag{1.2.7}$$

The Eichler symbol has numerous ties with the various notions of orders we introduced. For instance, the order O_ℓ is Eichler in $\mathrm{M}_2(\mathbb{Q}_\ell)$ if and only if $(O/\ell) = 1$.

More on quaternion ideals. The two notions of orders and ideals can be tied together by the *right and left orders* of an ideal.

Definition 1.2.12. Let I be a lattice in $B_{p;\infty}$. We define the *left order* of I to be:

$$O_L(I) = \{\alpha \in B_{p;\infty} \mid \alpha I \subseteq I\}; \tag{1.2.8}$$

and similarly for the *right order* $O_R(I)$. We say that I is a left fractional $O_L(I)$-ideal (resp. right $O_R(I)$) and that I is a *connecting ideal* for $O_L(I)$ and $O_R(I)$.

A fractional ideal is *integral* if it is contained in its left order, or equivalently in its right order. Since we will deal almost exclusively with integral ideals, we refer to them hereafter as ideals. Most of the time, in this manuscript, we will consider ideals of maximal orders, but we will have the time to make little detours for ideals in other orders. When O is a Bass order, O-ideals are

generated by two elements. In this work, we will always consider the case where I is locally principal. In this case, we can take one of the generator to be the norm of I and so we have $I = O_L(I) + O_L(I)n(I)$. We simplify the notation by writing $O + ON = O\langle \ ; N\rangle$ for any order O and $N \in \mathbb{N}$. Among *integral* ideals, we are mainly interested in *cyclic* ideals that we introduce next. This terminology is not really standard but we use it to highlight the parallelism with isogenies.

Definition 1.2.13. Let I be an integral left O-ideal for some order O. We say that I is *cyclic* if for all prime ℓ we have that $I \not\subset \ell O$.

Cyclic ideals as we define them in Definition 1.2.13 are usually called *primitive* (and cyclic is used for a different notion) but we use that notation to highlight the link with isogenies. When the order O in Definition 1.2.13 is Bass, we have that there exists an element $\in O$ with $\gcd(n(\ ); n(I)^2) = n(I)$ and $I = \langle \ ; n(I)\rangle$.

Quaternion ideal multiplication. In our setting, the product of ideals I, J is only defined when $O_R(I) = O_L(J)$. In that case, IJ is the ideal generated by the products of pairs in $I \times J$. It follows that IJ is also an (integral) ideal and $O_L(IJ) = O_L(I)$ and $O_R(IJ) = O_R(J)$. The ideal norm is multiplicative with respect to ideal products. An ideal I is invertible if there exists another ideal I^{-1} verifying $II^{-1} = O_L(I) = O_R(I^{-1})$ and $I^{-1}I = O_R(I) = O_L(I^{-1})$. The conjugate of an ideal $\overline{I}$ is the set of conjugates of elements of I, which is an ideal satisfying $O_L(\overline{I}) = O_R(I)$ (and conversely). When I is invertible, we also have $I\overline{I} = n(I)O_L(I)$ and $\overline{I}I = n(I)O_R(I)$ and we can get the multiplicative inverse of I as

$$I^{-1} = \frac{1}{n(I)}\overline{I}$$

Note that invertibility is not a feature of every O-ideal when O is generic, but we will see mostly examples where ideals are invertible. In fact the invertible ideals are exactly the locally principle ideals.

Henceforth, we arbitrarily choose to focus on left O-ideals (the situation is perfectly symmetric in any case, and we can switch from left O-ideals to right O-ideals by considering the conjugates). Even though we are mostly interested in these one-sided ideals, the existence of two-sided O-ideals is also worth mentioning. One fact in particular is going to be useful for us: given a maximal order O in $B_{p,1}$, there is a unique two-sided ideal of norm p. Note that this ideal is not necessarily principal. Moreover, and its powers are the only integral O-ideals of norm a power of p. Every ideal I of norm $p^r N$ where N is coprime to p admits the factorization $I = {}^{r}I'$.

Ideal class set. We define an equivalence relation on left O-ideals by right scalar multiplication. So we have that $I \sim J$ for two left O-ideals I and J if there exists $\in B_{p,1}^{\times}$, such that $I = J$.

Definition 1.2.14. For any invertible ideal J and any $\in B_{p,1}$, we write

$$_J(\ ) = J\frac{\overline{\ }}{n(J)}: \tag{1.2.9}$$

Ideals equivalent to J are precisely the ideals $_J(\ )$ with $\ 2\ J \cap f0g$ and we will make an explicit use of the map in Chapter 3. If $I = J\ $, then it follows that $O_R(I)$ and $O_R(J)$ are in the same type, and we have $O_R(I) = \ ^1O_R(J)\ $.

Remark 1.2.15. Note that the converse is not true. For two left O-ideals $I; J$, if $O_R(I) = O_R(J)$, then I is not necessarily equivalent to J (due to the possible existence of a non-principal two-sided ideal of norm p).

For a given O, we can de ne equivalence classes of left O-ideals, and we denote the set of such classes by Cl(O). The property above implies that each class of Cl(O) corresponds to a single type of orders.

De nition 1.2.16. Two orders $O_1; O_2$ are *connected* if there exists an invertible ideal I with $O_R(I) = O_1$ and $O_L(I) = O_2$. The *genus* of an order O is noted Gen O and it is made of all the orders connected to O.

By taking the conjugate ideals, we get that Gen $O' =$ Gen O i $O' \ 2$ Gen(O). Interestingly enough, grouping orders by genera agrees with the classi cation of orders that we have introduced above. This means that locally, the orders in the same genus are isomorphic and conversely, that orders that are isomorphic locally are in the same genus. For instance, this means that maximal orders are all contained in the same genus. Similarly, we get that all orders O of discriminant $p^{`e}$ where $O_`$ is an Eichler order of $M_2(Q_`)$ of level $`^e$ are in the same genus.

Given two orders $O_1; O_2$ in the same genus, the ideal $I(O_1; O_2)$ connecting $O_1; O_2$ is well-de ned up to scalar multiplication and multiplication by a two-sided ideal. If we look at integral ideals, we can de ne $I(O_1; O_2)$ as the connecting integral ideal with the smallest norm. It is easily veri ed that $I(O_1; O_2)$ is cyclic. Henceforth, we will keep this de nition and call $I(O_1; O_2)$ *the connecting ideal* of O_1 and O_2.

Remark 1.2.17. Note that if $O_2 = O_2^0$ (but they are not equal), the connecting ideals $I = I(O_1; O_2)$ and $J = I(O_1; O_2^0)$ may be equivalent but will not be equal.

Embedding of orders. Another aspect where quaternion orders o er much more complexity than quadratic orders is the one of *embedding*.

De nition 1.2.18. Given two orders $O; O'$, we say that O is *embedded* inside O' if there exists an injective morphism $\ : O\ ,!\ O'$. We call $\ $ an *embedding* and say that $\ $ is *optimal* if there does not exist any $O'' \ \ O'$ such that $\ (O) = Z + DO''$ for some integer $D > 1$.

Remark 1.2.19. De nition 1.2.18 also works if O is a quadratic order and O' is a quaternion order. For instance, it agrees with the de nition of *optimal embedding* given in [LB20]. We will talk more about embeddings of quadratic orders inside quaternion orders in Section 2.4.

In particular, we are interested in the *embedding number* of quaternion orders.

De nition 1.2.20. Let O be a quaternion order in $B_{p;\ 1}$. The *embedding number* $e(O)$ of O is the number of distinct maximal orders in which O is optimally embedded.

Eichler and Brzezinski [EB92, Brz83] studied embedding numbers of Bass orders, and their results were recently used in [EHL+20] to estimate the complexity of an algorithm to compute the endomorphism ring of a supersingular curve that we will briefly present in Section 2.2.1. We will also use their results in Section 2.4 and we give a brief summary of it in the remainder of this paragraph. We will talk about optimal embeddings of non-Gorenstein orders in Section 2.3.2.

In the rest of this paragraph, we fix a Bass order $\mathcal{O}$ of discriminant D. It turns out that $e(\mathcal{O})$ can be computed efficiently using the local-global principle with the formula $e(\mathcal{O}) = \prod_{\ell \in \mathrm{P}} e_\ell(\mathcal{O})$ where $e_\ell(\mathcal{O})$ is the analogue of $e(\mathcal{O})$ over the ℓ-adics: $e_\ell(\mathcal{O})$ is the embedding number of $\mathcal{O}_\ell$ inside $B_{p,\infty} \otimes \mathbb{Q}_\ell$. An easy preliminary observation is that, $e_\ell(\mathcal{O}) = 1$ when ℓ is coprime to D. Thus, we can rewrite the above formula as $e(\mathcal{O}) = \prod_{\ell \in \mathrm{P}_D} e_\ell(\mathcal{O})$. The value of $e_\ell(\mathcal{O})$ is in fact closely related to the Eichler symbol $(\mathcal{O}/\ell)$.

Then, it was shown by Eichler in [Eic36] (see [Brz83] for an account in English of this result) how the value of the Eichler symbol was linked to $e_\ell(\mathcal{O})$.

Proposition 1.2.21. *[Voi21] Let $\mathcal{O}$ be a Bass order in $B_{p,\infty}$ of discriminant D and $\ell \in \mathrm{P}_D$:*

$$e_\ell(\mathcal{O}) = \begin{cases} v_\ell(D) + 1 & \text{if } (\mathcal{O}/\ell) = 1; \\ 2 & \text{if } (\mathcal{O}/\ell) = 0 \text{ and } \ell \neq p; \\ 1 & \text{if } (\mathcal{O}/\ell) = -1 \text{ or } (\mathcal{O}/\ell) = 0 \text{ and } \ell = p. \end{cases}$$

Remark 1.2.22. Note that $e_p(\mathcal{O})$ is always equal to 1. This follows from Proposition 1.2.21 and the fact that $(\mathcal{O}/p)$ cannot be 1.

Chapter 2

The Deuring correspondence

This chapter is dedicated to the study of the Deuring correspondence from a theoretical point of view. Our goal is to provide a slightly informal but complete account of this topic, from the most fundamental theorems to the more evolved and recent results that will underlie several of the contributions introduced later in this thesis. We will cover quickly the standard material in Section 2.1, providing only statements and intuitions, to focus on the more advanced developments that will be exposed in Section 2.3. We claim no novelty in this section but precise proofs and statements for these results cannot really be considered standard (at least we are not aware of any other similar account in the literature) and our goal is thus to provide hereafter a complete reference. The content of Section 2.3 was introduced partly in [DFKL+20] and partly in [Ler21]. We also cover some algorithmic aspects of the Deuring correspondence in Section 2.2. Finally, in Section 2.4, we study the role of quadratic orders in the Deuring correspondence and prove some results on the number of curves admitting an embedding of a given quadratic order inside its endomorphism ring that were first presented in [Ler22].

For the remainder of this manuscript, we take the setting of supersingular elliptic curves over $\mathbb{F}_{p^2}$ for a fixed prime $p > 3$.

2.1 An equivalence of category in three acts

In this section, we conclude our mathematical preliminaries by tying together supersingular curves and quaternion orders in the powerful result presented in Theorem 2.1.8, and that we call the Deuring correspondence.

2.1.1 Endomorphism algebra and endomorphism rings, where it all begins

Let E be an elliptic curve defined over K a field of prime characteristic p. Throughout this work, unless said otherwise, we always consider the endomorphism ring over $\overline{K}$ and we write it $\mathrm{End}(E)$. In this section, we return to Proposition 1.1.21 by listing the possibilities for $\mathrm{End}(E)$ as either an order in a quadratic

imaginary field or a maximal order in a quaternion algebra. We will not provide a full proof, but will outline the reasoning and try to provide some insights.

The first step is to understand the *endomorphism algebra* $A_E = \mathrm{End}(E) \otimes \mathbb{Q}$ (Definition 1.1.18). The existence and properties of the involution given by the *dual map* implies that every element in A_E must have a minimal polynomial of degree at most 2 given by $X^2 - \mathrm{tr}(\alpha)X + n(\alpha)$ for any element α. Since the quadratic form $x, y \mapsto n(x\alpha + y) = x^2 + xy\,\mathrm{tr}(\alpha) + y^2 n(\alpha)$ is positive definite, we must have that A_E is either a quadratic imaginary field or a definite quaternion algebra.

When we are in the quaternion algebra case, we get a natural isomorphism between the ℓ-adic endomorphism algebras $\mathrm{End}(E)_\ell = \mathrm{End}(E) \otimes \mathbb{Q}_\ell$ and the endomorphism algebra of the Tate module $\mathbb{Q}_\ell \otimes \mathrm{End}(T_\ell(E))$. From Proposition 1.1.8, we get that $\mathrm{End}(T_\ell(E))$ must be equal to $\mathrm{M}_2(\mathbb{Z}_\ell)$ when ℓ is coprime to p and this proves that all ℓ coprime to p are split in A_E and this suffices to conclude that we must have $A_E = B_{p,\infty}$. To see that $\mathrm{End}(E)$ is a maximal order of $B_{p,\infty}$, we need to prove it locally, which is true by restricting the previous isomorphism to the tensor product with $\mathbb{Z}_\ell$ for all $\ell \neq p$. Maximality locally at p is slightly more subtle to prove, and we will not explain it here.

Seeing that the quaternion algebra case appears if and only all the other equivalent definition of supersingular curves (see Definition 1.1.22) are satisfied is also quite subtle and we refer the reader to [Sil86, Chapter 5].

A Concrete example : j-invariant 1728. We give below an example of a curve E_0 with a concrete isomorphism between $\mathrm{End}(E_0)$ and a maximal order in $B_{p,\infty}$. Let $p = 3 \bmod 4$, and let E_0 be the curve of j-invariant 1728, defined over $\mathbb{F}_{p^2}$ by $y^2 = x^3 + x$. The endomorphism ring of this curve is isomorphic to the maximal order $\mathcal{O}_0 = \langle 1, i, \frac{i+j}{2}, \frac{1+k}{2} \rangle$ with $i^2 = -1$, $j^2 = -p$ and $k = ij$. Moreover, we have explicit endomorphisms π and ϕ such that $\mathrm{End}(E_0) = \langle 1, \phi, \frac{\pi+\phi\pi}{2}, \frac{1+\phi\pi}{2} \rangle$, where π is the Frobenius morphism $(x, y) \mapsto (x^p, y^p)$ and ϕ is the map $(x, y) \mapsto (-x, \sqrt{-1}y)$. If we restrict to the endomorphisms defined over $\mathbb{F}_p$, we get the quadratic order $\mathbb{Z}[\pi]$. We will often use the curve E_0 throughout this thesis. When $p = 7 \bmod 12$, E_0 is the *special extremal curve* over $\mathbb{F}_{p^2}$ (see Definition 3.1.1 in Chapter 3).

2.1.2 Kernel ideals, the main development

Proposition 1.1.21 is the first building block of the equivalence of categories that we call the Deuring correspondence, as it ties endomorphism rings of supersingular curves (and so supersingular j-invariants up to Galois conjugacy) with types of maximal orders in $B_{p,\infty}$. A deeper and richer link between supersingular isogenies and ideals of maximal orders appears when we consider the full family of homomorphism modules. This is not surprising, as for any supersingular E, E' we can prove that $\mathrm{Hom}(E, E')$ is a rank 4 $\mathbb{Z}$-module in a manner analogous to the method briefly outlined in Section 2.1.1. Before getting to the final result in Section 2.1.3, we introduce the notion of *kernel ideals*. These ideals are isomorphic to our homomorphism modules as we prove in Proposition 2.1.6 and they provide an effective method to go from ideals to isogenies (and the converse) as we will see in Chapter 4. The results given in this section will be used as an informal proof of Theorem 2.1.8. Good references on *kernel*

ideals are the Chapter 3 of an article by Waterhouse [Wat69] and the Chapter 42 of John Voight's book [Voi21]. Proofs for the result we present below can also be found there.

Definition 2.1.1. Let k be a field and E/k an elliptic curve. For any finite subgroup $G \subset E$, we define its *kernel ideal*

$$I(G) = \{\alpha \in \mathrm{End}(E);\ \alpha(G) = 0\}. \tag{2.1.1}$$

When $G = \ker \phi$, for some separable isogeny ϕ, we write $I(\ker \phi) = I_\phi$.

Proposition 2.1.2. *For any supersingular elliptic curve E and finite subgroup $G \subset E$, the kernel ideal $I(G)$ is a left* $\mathrm{End}(E)$*-ideal of norm $\#G$ and $\mathcal{O}_R(I(G)) = \mathrm{End}(E/G)$. When G is cyclic, $I(G)$ is cyclic (see Definition 1.2.13).*

With Proposition 2.1.2, we see how to get left ideals of maximal orders from separable isogenies. We will treat *inseparable isogenies* after we show the dual construction of getting a separable isogeny from an ideal. Under the identification between isogenies and kernels, we define the *kernel of an ideal.*

Definition 2.1.3. Let k be a field and E/k an elliptic curve. Let I be an integral $\mathrm{End}(E)$-ideal of norm coprime to p. The *kernel of the ideal I* is defined as

$$E[I] = \{P \in E :\ \alpha(P) = 0 \text{ for all } \alpha \in I\}. \tag{2.1.2}$$

We then write $\phi_I : E \to E/E[I]$.

Naturally, the two notions that we introduced are dual of one another.

Proposition 2.1.4. *For any integral left* $\mathrm{End}(E)$*-ideal, the set of points $E[I]$ is a finite subgroup of order $n(I)$. Moreover, we have that $I(E[I]) = I$ and $E[I(G)] = G$ for any finite subgroup G.*

We now have an explicit correspondence between ideals and subgroups, and by extension a correspondence between ideals and separable isogenies. We can extend this to inseparable isogenies by identifying the unique two-sided $\mathrm{End}(E)$-ideal of norm p (and its powers) to the Frobenius morphism of E (and its powers). We remind the reader that since we consider isogenies over $\mathbb{F}_{p^2}$, the Frobenius is not always an endomorphism. This is consistent with the fact that the two-sided ideal of norm p is not always principal.

With that last idea, we obtain a complete correspondence between supersingular isogenies and integral ideals. There are several other notions that agree under this correspondence as shown in Proposition 2.1.5.

Proposition 2.1.5. *Let $\phi : E \to E'$ be an isogeny. We have $I_{\hat{\phi}} = \overline{I_\phi}$. If ϕ is an endomorphism, then I is a principal ideal.*
If $\psi : E' \to E''$ is another isogeny, then $I_{\psi\phi} = I_\phi I_\psi$.

The results above imply that for a maximal order $\mathcal{O}$ in $B_{p,1}$, the set $\mathrm{Cl}(\mathcal{O})$ is in bijection with $S(p)$. One of the consequences is that equivalent ideals correspond to isogenies with isomorphic domains and codomains. We will make good use of that fact in the second part of this thesis.

We now give another way of looking at kernel ideals that gives a better understanding of their structure.

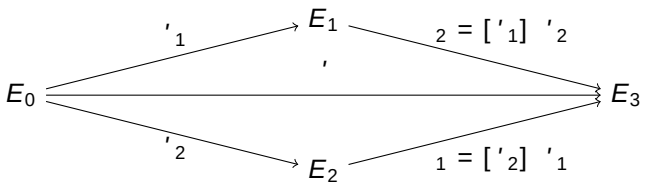


Figure 2.1: A commutative isogeny diagram.

Proposition 2.1.6. *Let $\phi : E \to E'$ be an isogeny, the kernel ideal I_ϕ is isomorphic to* $\mathrm{Hom}(E', E) \circ \phi$.

The notions of *pushforward* and *pullback* isogenies as defined in Section 1.1.2 can be translated to ideals under our correspondence. To highlight the parallelism, we write $[I]\, J$ for the ideal $I_{[\phi]\circ\phi_J}$ and call it the *pushforward of J through I*. The same holds for $[I]\, J$. With this convention, we extend the terms *pushforward* and *pullback* to ideals. Next, we describe formulas to compute $[I]\, J$ and $[I]\, J$ from I and J.

We take the notations of Figure 1.1 that we recall below as Figure 2.1 with the corresponding notations for ideals, and write $I_1 = I_{\phi_1}$ of norm N_1, $I_2 = I_{\phi_2}$ of norm N_2, $J_1 = [I_2]\, I_1$, $J_2 = [I_1]\, I_2$ and $K = I_{\hat{\phi}}$.

Lemma 2.1.7. *If $N_1 \wedge N_2 = 1$, the three ideals J_1, J_2 and K are well-defined and :*

(i) $K = I_1 \setminus I_2$.

(ii) $J_2 = {I_1}^{-1}(I_1 \setminus I_2)$ *and* $J_1 = {I_2}^{-1}(I_1 \setminus I_2)$.

(iii) $I_2 = [I_1]\, J_2 = I_1 J_2 + N_2 \mathcal{O}_0$ *and* $I_1 = I_2 J_1 + N_1 \mathcal{O}_0$.

Proof. When $N_1 \wedge N_2 = 1$ the situation depicted in Figure 2.1 and the pushforward and pullback isogenies are well-defined and so are the corresponding ideals. By definition of $\hat{\phi}$ we have $\ker \hat{\phi} = \ker \hat{\phi}_1 + \ker \hat{\phi}_2$, (i) follows from the definition of kernel ideals. The composition of isogenies can be rewritten in terms of ideals as $K = I_1 J_2 = I_2 J_1$, this together with (i) implies (ii). The equality $[I_1]\, J_2 = I_1 J_2 + N_2 \mathcal{O}_0$ of (iii) is a classical formula to decompose an ideal of norm $N_1 N_2$ with coprime N_1, N_2. For instance, it is used in [GPS17, EHL+18]. The fact $I_2 = [I_1]\, J_2$ stems from $I_2 = [I_1]\, [I_1]\, I_2$. The formula for I_1 follows similarly. □

2.1.3 The conclusion

In this section, we state the formal equivalence of categories. For that, we need to fix a supersingular curve E_0 and a maximal order $\mathcal{O}_0$, isomorphic to the endomorphism ring of E_0.

Theorem 2.1.8. *The association*

$$E \mapsto \mathrm{Hom}(E, E_0)$$
$$\phi \mapsto \hat{\phi}$$

where $' \ 2$ Hom$(E; E^0)$ *and*

$$' \quad : \text{Hom}(E^0; E_0) \quad ! \ \text{Hom}(E; E_0)$$
$$7 \ ! \qquad '$$

is functorial and de nes an equivalence of categories between

supersingular j-invariants over $\mathbb{F}_{p^2}$*, under isogenies*

and

invertible left $\mathcal{O}_0$*-modules, under left* $\mathcal{O}_0$*-modules homomorphisms.*

Theorem 2.1.8 follows from the results from Section 2.1.2. A detailed proof can be found in [Voi18, Chapter 42, Theorem 42.3.2].

Note that the functor introduced in Theorem 2.1.8 is rather a theoretical object that we will not use in practice. In the remainder of this work, what we call \the Deuring correspondence" is an implicit dictionary between the objects in the world of elliptic curves and the objects in the world of quaternion algebras. This dictionary sends a supersingular j-invariant to the type of its endomorphism ring in $B_{p; 1}$ and a supersingular isogeny to its kernel ideal (up to isomorphisms between left and right orders). While not exactly functorial, this correspondence is bijective. We summarize in Table 2.1 the main features of this association. We include some of the results that will be presented in Section 2.3.

Supersingular j-invariants over $\mathbb{F}_{p^2}$	Types of maximal orders in $B_{p; 1}$
$j(E)$ (up to Galois conjugacy)	$\mathcal{O} = \text{End}(E)$
$(E_1; ')$ with $' : E \ ! \ E_1$	$I_{'}$ integral left $\mathcal{O}$-ideal and right $\mathcal{O}_1$-ideal
$\ 2 \ \text{End}(E_0)$	Principal ideal $\mathcal{O}$
deg$(')$	$n(I_{'})$
^	$\overline{I_{'}}$
$' : E \ ! \ E_1; \quad : E \ ! \ E_1$	Equivalent Ideals $I_{'} \quad I$
Supersingular j-invariants over $\mathbb{F}_{p^2}$	Cl$(\mathcal{O})$
$\quad : E \ ! \ E_1 \ ! \ E_2$	$I \quad = I \quad I$
N-isogenies (up to isomorphism)	Class set of Eichler orders of level N

Table 2.1: The Deuring correspondence, a summary.

2.2 Algorithmic Deuring correspondence

The results from Section 2.1.3 are nice in theory, but the question of practicality remains. Existence results and theoretical properties are not enough to build cryptography. We need concrete algorithms and complexity estimates, either to build security assumptions or to argue that our protocols will be e cient. Thus, the natural question is: how can we perform concretely the translation between the two columns in Table 2.1? In this section, we will try to present an informal answer to this question. Providing a more detailed response will be at the heart of Chapters 3 and 4.

To understand a bit better the motivation, we give a quick preview of the content developed in Part II. Isogeny-based cryptography was born from the idea that computing an isogeny between two given isogenous elliptic curves is hard. When the curves in input are supersingular, the problem is believed to be particularly difficult. In that case, we call it the *supersingular isogeny problem* (SIP) and we add the prefix `-` if we restrict to isogenies in the `-isogeny graph. Naturally, given the dictionary from Table 2.1, one wonders if the Deuring correspondence could be used to help solving the isogeny problem more efficiently. The first result on the topic was provided by Kohel, Lauter, Petit and Tignol in [KLPT14] where they gave a heuristic polynomial algorithm (that we denote by KLPT) to solve the *quaternion `-isogeny path problem*: the translation of the `-SIP under the Deuring correspondence. We present the exact formulation of this problem and the algorithm KLPT in Chapter 3. KLPT was the first building block toward a more thorough analysis of the hardness of the problems related to the Deuring correspondence [EHL+18, Wes22]. The central problem in this context appears to be the *endomorphism ring problem* (ERP) presented as Problem 2.2.1 below.

Problem 2.2.1. *(ERP) Given a supersingular curve E, compute* $\mathrm{End}(E)$.

Using KLPT and several other heuristic arguments, the authors of [EHL+18] were able to prove that the SIP is equivalent to the ERP. Their work was later completed by Wesolowski [Wes22], who proved the same result assuming only the generalized Riemann hypothesis. These results justify that finding algorithms to solve the ERP is important. In Section 2.2.1, we outline such an algorithm. In Section 2.2.2, we briefly cover other problems and algorithm to solve them.

Remark 2.2.2. In the formulation of Problem 2.2.1, we stated the goal as \computing $\mathrm{End}(E)$" without explaining what this means exactly. We remained vague on purpose to keep the formulation compact. In the literature, there are usually two versions of the ERP: one where we want four elements of $B_{p,1}$ making a basis of a maximal order isomorphic to $\mathrm{End}(E)$ and one where we look for concrete endomorphisms generating $\mathrm{End}(E)$ as morphisms from E to E. It is not completely trivial to prove, but these two versions are equivalent (see [EHL+18]). In Section 2.2.1, we give an algorithm to solve the first version. As a generic rule, throughout this work, when an algorithm has an input or an output that is isomorphic to , a lattice of $B_{p,1}$, we assume that it should be put in the form of four elements of $B_{p,1}$ forming a basis of .

2.2.1 Endomorphism ring computation

The first concrete algorithm to solve the ERP is due to Kohel in his thesis [Koh96], and the literature has not evolved much since. We can also mention the work of Cervino [Cer04] based on ternary quadratic form (the goal of Cervino was slightly different since he solves the ERP for all supersingular curves over p at once). Below, we present briefly an algorithm from [EHL+20] that can be seen as an actualized version of the one introduced by Kohel. The complexity of this algorithm is conjectured to be $\tilde{O}(p)$. In Section 2.4 we actually prove one of the conjectures made in [EHL+20] (with a slightly different motivation in mind).

Note that the ERP is easy in some rare cases, such as the curve of j-invariant 1728 when $p = 3 \bmod 4$. In general, there are a few distinguished curves for which we can find the endomorphism ring quite easily, but a random element in $S(p)$ will not be among those examples with overwhelming probability when p grows.

Unsurprisingly, the main bottleneck in computing the endomorphism ring of a generic supersingular curve E is actually to compute some non-trivial endomorphisms of E. The conjecture made in [EHL+20], which was already at the heart of the work of Kohel, is that two non-trivial endomorphisms are already enough most of the time. If θ_1, θ_2 are two endomorphisms of E, then we can look at the order $\mathcal{O}_{1,2}$ generated by $1, \theta_1, \theta_2, \theta_1\theta_2$. Unless θ_1, θ_2 commute, this order has rank 4 and with good probability it will make a good part of $\mathrm{End}(E)$. If this order is big enough, it will be contained only in a few maximal orders and one can enumerate all of them until the correct one is found.

More precisely, the size of the list to enumerate through is exactly the embedding number of the order $\mathcal{O}_{1,2}$. When $\mathcal{O}_{1,2}$ is Bass, we have with Proposition 1.2.21, an exact formula to compute the embedding number from the factorization of $\mathrm{disc}(\mathcal{O}_{1,2})$. It can be upper-bounded by the number of the divisor function σ that we introduce later (Definition 2.4.17) and it is a classical result by Wigert [Wig07] that $\sigma(N) = O(N^{\varepsilon})$ for any $\varepsilon > 0$. Computing each of the maximal orders containing $\mathcal{O}_{1,2}$ can be done by using the local-global principle. For each prime ℓ dividing $\mathrm{disc}(\mathcal{O}_{1,2})$, we can find the $e_\ell(\mathcal{O}_{1,2})$ maximal orders containing $\mathbb{Z}_\ell \mathcal{O}_{1,2}$ using Bruhat-Tits trees (see [AIL+21] for an exposition on Bruhat-Tits trees and their link with the Deuring correspondence). Thus, under the heuristic that if θ_1, θ_2 are well-distributed in $\mathrm{End}(E)$, then the order $\mathcal{O}_{1,2}$ will be Bass with good probability, we get that we can recover efficiently the full endomorphism ring of E from the two endomorphisms θ_1, θ_2.

It remains to see how one can compute two non-trivial endomorphisms. The idea is to look for cycles in G_p^{ℓ} for any small prime ℓ by performing some random walks until we find a collision. In [EHL+20], they look for pairs j, j^p that are neighbors in G_p^{ℓ} to improve the chance of finding a collision, but the principle is always the same. Given the size and the expansion property of G_p^{ℓ}, we can expect to find collisions in $O(\sqrt{p}\log(p)^2)$.

Remark 2.2.3. The reasoning above explains that the endomorphism ring problem is basically considered equivalent to the endomorphism problem (which consists in computing an endomorphism of a curve E). It is hard to prove this reduction formally because we need some assumptions on the distributions of θ_1, θ_2 to be able to argue that the embedding number of $\mathcal{O}_{1,2}$ is small, but it appears plausible in practice. The authors of [EHL+20] provided empirical evidence to back their conjecture. With the suborder representation that we present in Section 4.3, we will introduce the suborder endomorphism ring problem, a new problem whose hardness is based on the reverse consideration that if the revealed endomorphisms are well-chosen and have a huge embedding number, it can be hard to recover the full endomorphism ring. We will use this problem to construct some new protocols in Chapter 7.

2.2.2 Other results and algorithms.

Below, we brie y review some other algorithmic problems occurring in the context of the Deuring correspondence. We will remain mostly informal in what follows and postpone detailed descriptions to Chapter 3 and 4. This section can be considered as a preview and a motivation of what we will present in the coming chapters.

In general, we will see that any problem starting from the quaternion side of Table 2.1 will prove to be easy. For the algorithms with inputs in the elliptic curve side, everything depends on the knowledge of the endomorphism ring. If the endomorphism ring of the curves involved in the problem are known, then the problem is easy, otherwise the problem is hard (and in fact as hard as the ERP in most cases). In particular, when End(E) is known, there are algorithms to translate an isogeny $' : E \to E'$ into the corresponding kernel ideal (and the converse) in time polynomial in $\log(p)$. The complexity also depends on $\deg '$ as we will see in Section 4.2 where we present two algorithms IsogenyToIdeal and IdealToIsogeny dedicated to that task. We will also present some e cient version of IdealToIsogeny when the degree of $'$ is the power of a small prime.

We have also the inverse problem of the ERP: given a maximal order O in $B_{p;\infty}$, nd a curve E with End(E) = O. This problem can actually be solved in polynomial time by taking advantage of the few supersingular curves whose endomorphism rings are known. We have for instance [EHL+18, Algorithm 12] to perform that task. We describe brie y how this algorithm works. Let E_0 be such a curve of known endomorphism ring, and let us write $O_0 \simeq B_{p;\infty}$ a maximal order in the type of End(E_0). The rst ingredient to compute the curve E is an algorithm ConnectingIdeal. We brie y explain the purpose of this algorithm below, but we do not give a detailed description because is just simple linear algebra. We state in Proposition 2.2.4 a complexity result on ConnectingIdeal that we will reuse later.

ConnectingIdeal: takes takes two maximal orders O_1, O_2 in $B_{p;\infty}$ and outputs $I(O_1, O_2)$.

Proposition 2.2.4. ConnectingIdeal *terminates in* $O(\text{poly}(\log(pC)))$ *when the coe cients of the bases of the two maximal orders in input are smaller than* C.

With ConnectingIdeal, we can compute a connecting ideal between O_0 and O. This is equivalent to an isogeny between E_0 and E under the Deuring correspondence. Thus, to compute the correct output, it su ces to compute the codomain of this isogeny. This can be done with IdealToIsogeny and the usual isogeny computation algorithms (we give more details on this topic in Section 4.1 and Section 4.2). However, as the result of Chapter 4 will prove, the complexity of these operations depends on the norm of $I(O_0, O)$ and so we have no guarantee for the complexity will be polynomial (in particular, if the norm has big prime factors, it will have exponential complexity). As we will explain in Chapter 4, the only way to guarantee a polynomial computation is to have a powersmooth norm (with smoothness bound of polynomial size). Thus, we need an algorithm that takes an ideal I of any norm and output an equivalent ideal $J \sim I$ of powersmooth norm. Luckily, this is exactly the purpose of the KLPT algorithm from [KLPT14] that we will describe in Chapter 3. If $J \sim I(O_0, O)$, the corresponding isogeny $'_J$ has a codomain $E_0/E_0[J]$ that is a correct output for our algorithm. When $n(J)$ is powersmooth with a polynomial smoothness

bound, we have a polynomial algorithm to compute E_0=$E_0[J]$ and so we get a polynomial algorithm to solve the reverse ERP because KLPT is also polynomial.

We conclude this section with a small result to prove that we always have a polynomial-size representation of ideals in $B_{p;1}$. We will use this fact in various occasions in the coming chapters.

Lemma 2.2.5. *For any integral ideal I of a maximal order in $B_{p;1}$ of norm D, there exists J, isomorphic to I, that admits a basis composed of elements with coefficients of size $O(\log(pD))$ over the basis $\langle 1; i; j; k \rangle$.*

Proof. It was shown in [EHL+18] that any maximal order admits a basis whose coefficients have size $O(\log(p))$ in the basis $\langle 1; i; j; k \rangle$ of $B_{p;1}$. Since $DO \subset I$ for any cyclic O-ideal of norm D, we see that we can choose coefficients to represent any element of I inside the basis of O with coefficients of size $O(\log(D))$. Thus, there exists a representation of a basis of I in $\langle 1; i; j; k \rangle$ whose coefficients have size $O(\log(p) + \log(D))$. □

2.3 Interpreting the zoology of quaternion orders under the Deuring correspondence

In this section, we expand upon the results of Section 2.1 by understanding how the objects introduced in the context of quaternion orders translate into the world of elliptic curves and isogenies. We will focus on two main subjects: Eichler orders in Section 2.3.1 and the Brandt-invariant in Section 2.3.2. We stress again that these results are considered folklore (see for instance [Voi18, Remark 42.3.10] for Eichler orders) but we address the lack of a clear and explicit reference. As we will need a lot of the intermediate results in Part II, our proof does not take the most direct way toward the result and we try to make each step as explicit as possible. This was also recently done in full details by Sarah Arpin [Arp22]. The most important result is about Eichler orders in Section 2.3.1, but we will also look at non-Gorenstein orders in Section 2.3.2.

2.3.1 Eichler orders

Looking at the interpretation of Eichler orders under the Deuring correspondence is quite natural, as there is a link between Eichler orders and cyclic ideals of maximal orders. This stems from Eichler orders being the intersection of two maximal orders. Let us take O such an order, and we write $O_1; O_2$ for the two maximal orders such that $O = O_1 \cap O_2$. We take I the cyclic ideal $I(O_1; O_2)$. It can be shown that the level of O is equal to $n(I)$. The results of this section will culminate in Proposition 2.3.14 where we show a correspondence between Eichler orders class sets of level N and N-isogenies. Throughout this section, we keep the same notations for $O; O_1; O_2$ and I.

The following proposition clarifies the link between ideals and Eichler orders in terms of structure of O.

Proposition 2.3.1. *Let $O_1; O_2$ be two maximal orders in $B_{p;1}$. We have $O_1 \cap O_2 = \mathbb{Z} + I(O_1; O_2)$. With the notations above, we get $O = \mathbb{Z} + I$.*

Proof. Since I is integral, it is contained in both O_1 and O_2 and we have that $\mathbb{Z} + I \quad O$. We conclude by observing that the index of $\mathbb{Z} + I$ in both O_1 and O_2 is $n(I)$, the same as O. □

Remark 2.3.2. With the decomposition from Proposition 2.3.1, we can partition the elements of O according to whether their norm is coprime to $n(I)$ or not. Given that $n(I)\mathbb{Z}_S \quad I$, it is easily veri ed that this partition can be written as $O = (I \setminus \overline{I}) \quad ((\mathbb{Z}_{\text{cop}}(n(I)) + I)$ where $\mathbb{Z}_{\text{cop}}(n)$ is the set of integers in $\mathbb{Z}$ coprime to n.

The endomorphisms of O. We strive to understand how the elements of O t into our framework. We have that the two maximal orders O_1,O_2 are isomorphic to the endomorphism rings $\text{End}(E_1)$; $\text{End}(E_2)$ of two supersingular curves $E_1; E_2$. Let us write $_i : O_i \; ! \; \text{End}(E_i)$ the isomorphism in question for $i = 1; 2$. Let us take an element $\quad 2 O$. By design $\quad 2 O_1$ and $\quad 2 O_2$, so we may ask what are the endomorphisms $_1(\;)$ and $_2(\;)$ and how they may be related. Proposition 2.3.3 below shows that these two endomorphisms are image of one another under pushforward/pullback by $'_I$. We formulate this result in terms of ideals.

Proposition 2.3.3. *If* $\quad 2 O_1$ *is of norm coprime to* N*, then* $[O_1 \;] I = I$ *if and only if* $\quad 2 O$ r $(I \setminus \overline{I})$*. In particular,* $[I] \; O_1$ *is a principal* O_2*-ideal equal to* O_2 *.*

Proof. When $\quad 2 O$ r $(I \setminus \overline{I})$, the norm of $\quad$ is coprime to N as noted in Remark 2.3.2. Thus, Lemma 2.1.7 applies and we have $[I] \; (O_1 \;) = I \; ^1(I \setminus O_1 \;)$. We now show that $I \setminus O_1 \quad = I \quad$. Indeed, since I is integral, $I \quad O_1$ and as $\quad 2 O \quad O_2 = O_R(I)$ we also have $I \quad I$. For the other side, let us take $x \; 2 \; O_1 \quad \setminus I$. We can write $x = \quad$ for $\quad 2 O_1$. Writing $\quad = \; + \;$ with $\quad 2 \mathbb{Z}$ invertible modulo N and $\quad 2 I$, we see that $\quad$ is necessarily in I. We have proven that $I \setminus O_1 \quad = I \quad$, and $[I] \; (O_1 \;) = I \; ^1(I \setminus O_1 \;)$ concludes the rst part of the proof with $I \; ^1 I = O_2$.

Now, we show that if $[O_1 \;] I = I$, then $\quad$ is necessarily in O. If $[O_1 \;] I = I$, we know that the kernel $E_1[I]$ of $'_I$ is xed by the action of $\quad$. This implies that $E_1[I]$ is in an eigenspace of $\quad$ (since $E_1[I] = \ker '_I$ is a cyclic subgroup) and there exists $\quad 2 \mathbb{Z}$ such that $\quad 2 I$. Hence, $\quad 2 O$ by Proposition 2.3.1.

We have shown that $I \setminus O_1 \quad = I \quad$, and we can conclude the proof using the formula $[I] \; (O_1 \;) = I \; ^1(I \setminus O_1 \;)$. We obtain $[I] \; (O_1 \;) = O_2 \quad$, and this ideal is principal since $\quad 2 O_2$. □

From Proposition 2.3.3, we deduce directly a complete analysis in Proposition 2.3.4 of the elements of $_1(O)$; $_2(O)$.

Proposition 2.3.4. *For* $\quad 2 O$ *one of the following holds :*

- $n(\;) = 0 \mod n(I)$ *and* $_1(\;) = \quad$ *or* $_1(\;) = \; ^- \;$ *with* $\quad = \quad '_I \; 2$ $\text{End}(E_1)$ *for* $\quad : E_1 \; ! \; E_2$ *and* $_2(\;) = \; ^\wedge \; ^\wedge_I$*.*
- $n(\;) \; 6 = 0 \mod n(I)$ *and* $_2(\;) = [' _I] \; _1(\;)$*.*

Remark 2.3.5. If we reformulate Proposition 2.3.3 with the de nition of pushforwards, we see that the endomorphisms $_1(\;)$ leaves stable $E_1[I]$ when $n(\;)$

is coprime to I. In fact, this is true for every element $\in O$, as can be easily verified by the decomposition $O = \mathbb{Z} + I$ and Definition 2.1.3. Conversely, it can be shown that any element in $\mathrm{End}(E_1)$ having this property is in fact contained in $\iota_1(O)$. We can also show that $E_1[I]$ is the only subgroup that is an eigenspace of all the elements of $\iota_1(O)$. This justifies that we can consider $\iota_1(O)$ as the endomorphism ring of the curves E_1 augmented with the cyclic subgroup $E_1[I]$ (and alternatively ι_2 is $\mathrm{End}(E_2; E_2[\bar{I}])$). In terms of local-global interpretation, this can be seen with Eq. (1.2.5), where we have that O_ℓ is isomorphic to a ring of triangular 2×2 matrices over the ℓ^e-torsion for every ℓ^e dividing the level of O. Since O_ℓ gives the action of the endomorphisms in $\iota_i(O)$ on the Tate modules $E_i[\ell]$, this means that there is always one common eigenvector among the ℓ^e-torsion.

From Proposition 2.3.3, we deduce the following result which will underlie the algorithm GenericKLPT from Chapter 3 and also be useful for our analysis of ideal class sets of Eichler orders; it is a reformulation using the map from Definition 1.2.14. We remind that we have $\iota_I(\ ) = I\overline{\ }/n(I)$.

Corollary 2.3.6. *Let $J_1; J_2$ be O_1-ideals, with $J_1 \sim J_2$ and $n(J_1); n(J_2); n(I)$ pairwise coprime. Suppose that $J_1 = \iota_{J_2}(\ )$ with $\in J_2 \setminus O$. Then $[I]J_1 \sim [I]J_2$ and $[I]J_1 = \iota_{[I]J_2}(\ )$.*

Proof. When $\iota_{J_2}(\ ) = J_1$, we can identify $J_2\overline{J_1}$ with O_1. By Proposition 2.3.3 we know that $[I]O_1 = O_2$ and by decomposing O_2 the same way as O_1, we see that $[I]J_1 = \iota_{[I]J_2}(\ )$. □

In fact, we can show that the converse of Corollary 2.3.6 does not hold in general. As shown in Lemma 2.3.7 below, there are cases where $\in O_1 \setminus O$ can be found such that $[I]O_1$ is principal (this result is not used anywhere else in this work but we found it interesting). In this context, there exists a $\gamma \in O_2$ such that $[I]O_1 = O_2\gamma$. Of course $n(\ ) = n(\gamma)$, however, it appears that the trace of is not necessarily preserved in this case. This means that even though is sent to an endomorphism over E, the suborder $\mathbb{Z}[\ ]$ of O_1 is not sent to an isomorphic suborder $\mathbb{Z}[\gamma] \subset O_2$.

Lemma 2.3.7. *If there exists $J \not\sim I$ of the same norm with $J \sim I$, then there exists $\in O_1 \setminus O$ such that $J = [O_1\ ]I$ and $[I]O_1$ is principal.*

Proof. We need to show that we can always find $\in O_1 \setminus O$ such that $[O_1\ ]I = J$ (i.e. $[I]O_1$ is principal since $J \sim I$). This is the case if $J \subset I$. Indeed, any endomorphism of J can be written as a composition of with an element of J. The kernel of the elements in J are exactly $E_1[J]$ by definition, but since J is in I, the elements of J send $E_1[I]$ to O. The only possibility is that $(E_1[I]) = E_1[J]$. By definition of our pushforward isogenies this is equivalent to $[O_1\ ]I = J$. Hence, $J \subset I$ is sufficient to prove the result.

We just need to justify that such a can be found for any given pair of distinct $I \subset J$. There are several ways to construct it, for instance we can do so by computing IdealModConstraint$(\ ; J)$ (the algorithm defined in Section 3.2.2) for any such that $I = \langle\ ; n(I)\rangle$. Finally, since $I \subset J$ we conclude that $[I]O_1$ is principal. □

Ideal class sets of Eichler orders. We have seen in Section 2.1.2 that the class set of a maximal order is in bijection $S(p)$. We are going to prove next that we can derive a similar result for class sets of Eichler orders. We derive it by explicitly showing the connection between ideals of maximal orders and ideals of Eichler orders. More than the final result Proposition 2.3.14, our goal is really to understand the different elements of this class set and how they relate to the class sets of maximal orders.

To motivate this study, note that the supersingular isogeny graphs from Definition 1.1.28 were first constructed by Pizer [Piz90] through class sets of quaternion orders, and only later reinterpreted as isogeny graphs in [CLG09].

For simplicity, we now assume that the norm of the ideal I is a prime N. Everything that follows remains mostly true for an arbitrary integer N when $N+1$ is replaced by $N \prod_{d \mid N} (1+1/d)$. Eichler [Eic38] proved a formula for the class number $h(\mathcal{O}) = \#\mathrm{Cl}(\mathcal{O})$. When N is prime, we obtain

$$h(\mathcal{O}) = \frac{(p+1)(N+1)}{12} + \varepsilon_{N,p}$$

where $\varepsilon_{N,p}$ is a small value depending on N and p modulo 12 (analogue to Proposition 1.1.24). This, combined with $h(\mathcal{O}_1) = p/12 + \varepsilon_p$, ($\varepsilon_p$ depends on the value $p \bmod 12$) suggests that there is a $(N+1)$-to-1 correspondence between $\mathrm{Cl}(\mathcal{O})$ and $\mathrm{Cl}(\mathcal{O}_1)$, which we are now going to exhibit.

Remark 2.3.8. By symmetry of the definition of $\mathcal{O} = \mathcal{O}_1 \cap \mathcal{O}_2$, everything could be restated replacing $\mathcal{O}_1$ by $\mathcal{O}_2$, up to replacing some pushforward notations $[\,]$ by pullbacks $[\,]$ when it makes sense (or equivalently replacing I by $\bar{I}$).

The first step in our approach is to understand how to construct ideals of Eichler orders from ideals of maximal orders. We focus our analysis on ideals of norm coprime to N because they are simpler to handle and the results we can state on them remain true when looking up to equivalence because every ideal class has a representative of norm coprime to N. Let us write $I_N(\mathcal{O})$ for the set of left integral $\mathcal{O}$-ideals of norm coprime to N for any order $\mathcal{O}$. We start by showing a connection between $I_N(\mathcal{O}_1)$ and $I_N(\mathcal{O})$.

Lemma 2.3.9. *The map*

$$\begin{aligned} \varphi : I_N(\mathcal{O}_1) &\to I_N(\mathcal{O}) \\ J &\mapsto J \cap \mathcal{O} \end{aligned}$$

is a well-defined bijection between the set of integral $\mathcal{O}_1$-ideals and $\mathcal{O}$-ideals of norm coprime to N. Its inverse is given by $\varphi^{-1} : J \mapsto \mathcal{O}_1 J$.

Proof. Verifying that the images of φ (resp. φ^{-1}) are left integral $\mathcal{O}$-ideals (resp. $\mathcal{O}_1$-ideals) is straightforward from the definition. Then, it suffices to show $I = \mathcal{O}_1(I \cap \mathcal{O})$ and $J = \mathcal{O} \cap \mathcal{O}_1 J$ for any $I \in I_N(\mathcal{O}_1)$ and $J \in I_N(\mathcal{O})$. This is straightforward after seeing that any $\mathcal{O}_1$-ideal of norm coprime to N can be written as $J = \mathcal{O}_1\langle \alpha, n(J)\rangle$ for some $\alpha \in \mathcal{O}$. The corresponding $\mathcal{O}$-ideal is $J = J \cap \mathcal{O} = \mathcal{O}\langle \alpha, n(J)\rangle$ and $\mathcal{O}_1 J = J$. Moreover, this decomposition justifies that the norm is preserved through φ. □

Note that locally, since $\mathcal{O} \otimes \mathbb{Z}_\ell$ and $\mathcal{O}_1 \otimes \mathbb{Z}_\ell$ are isomorphic for all primes $\ell \nmid N$, it is not surprising to see the existence of the correspondence between

ideals of norm coprime to N presented in Lemma 2.3.9. This bijection suggests de ning the following equivalence relation $_O$ on left O_1-ideals of norm coprime to N. We say that $J \ _O K$ if and only if $(J) \ (K)$ as O-ideals (here is the equivalence relation introduced in Section 1.2.2 between left ideals of a same order). The bijection transports the structure of to $_O$, and this implies that we have de ned an equivalence relation.

De nition 2.3.10. We write $\mathrm{Cl}_O(O_1)$ for the set of equivalence classes of $I_N(O_1)$ under $_O$.

From the de nition, we have that $\mathrm{Cl}_O(O_1)$ is in bijection with $\mathrm{Cl}(O)$ through . In the next proposition, we make the link between class sets and our previous results on the elements of O by showing that we can obtain an explicit correspondence between ideals of norm N and $\mathrm{Cl}_O(O_1)$ using pushforward ideals.

Proposition 2.3.11. *$J \ _O K$ if and only if there exists $2 O$ such that $K = \ _J( )$ and $^1[K] \ I \ = [J] \ I$.*

Proof. We start by noting that $^1[K] \ I \ = [J] \ I$ is an equality of left $O_R(J)$-ideals. Indeed, $K = \ _J( )$ implies $O_R(J) = \ ^1O_R(K)$ (equivalent ideals have equivalent right orders).

By de nition of $_O$ and properties of our bijection , $J \ _O K$, $K = \ _J( )$ for some $2 O$. In this case, applying the formula of Lemma 2.1.7 for $[K] \ I$ yields $^1[K] \ I \ = \overline{K} \ (I \ \backslash \ K) \ =n(\ )n(K)$, which can be simpli ed as $J \ ^1 \ (I \ \backslash \ K) \ =n(K)$ with $K = \ (J)$. As noted in Corollary 2.3.6, when $2 O$ we can write $[I] \ K = \ _{[I] \ J}( )$. With this and the decomposition $I \backslash J = I \ [I] \ J$, we see that $(I \backslash J) = (I \backslash K) \ =n(K)$. By replacing $(I \backslash K) \ =n(K)$ in $^1[K] \ I \ = J \ ^1 \ (I \ \backslash \ K) \ =n(K)$, we obtain $^1[K] \ I \ = J \ ^1(I \ \backslash \ J) = [J] \ I$. □

An interesting question is how the new equivalence relation $_O$ relates to the classical one . In fact, $_O$ is compatible with in the sense that $J \ _O K$ implies $J \ K$, as is easily veri ed from Corollary 2.3.6. This suggests partitioning $\mathrm{Cl}_O(O_1)$ in subsets indexed by the elements of $\mathrm{Cl}(O_1)$. Understanding this partition is the focus of Proposition 2.3.12 and will lead naturally to our nal result of Proposition 2.3.14. Hence, we write

$$\mathrm{Cl}_O(O_1) = \bigcup_{C 2 \mathrm{Cl}(O_1)} \mathrm{Cl}_O(C)$$

where $\mathrm{Cl}_O(C)$ is the set of classes in $\mathrm{Cl}_O(O_1)$ contained in C. As mentioned above, the respective sizes of $\mathrm{Cl}(O_1)$ and $\mathrm{Cl}(O)$ suggest that the partition above provides a $(N + 1)$-to-1 correspondence between $\mathrm{Cl}(O_1)$ and $\mathrm{Cl}(O)$. The di erence between $h(O)$ and $(N + 1)h(O_1)$ is entirely accountable to the classes C not treated by Proposition 2.3.12, which we will brie y describe in Remark 2.3.13.

Proposition 2.3.12. *For $C 2 \mathrm{Cl}(O_1)$, let us take $L 2 C$ and de ne $O_C := O_R(L)$. If $O_C = h \ 1i$, then for any $2 L \ r \ NO_C$ and quadratic order $S = \mathbb{Z}[! \ _S]$ of discriminant $_S$ inside O_1 in which N is inert, the map:*

$$\begin{aligned} : \mathrm{P}^1(\mathbb{Z}/N\mathbb{Z}) \ &! \ \mathrm{Cl}_O(C) \\ (C : D) \ 7 &! \ _L((C + \ !_S D) \ ) \end{aligned}$$

is a bijection. In particular, $j\mathrm{Cl}_O(C)j = N + 1$.

Proof. First, it is clear that such and S can be found for any class C and representative L. We propose to prove the proposition by decomposing in two bijections $_1$ and $_2$. For this, we reformulate our equivalence relation as a relation on the ideal elements. For $_0; _1 \in L$ of norm coprime to N, we de ne the relation $_O$ as $_0\overline{_1}=n(L) \in O$. It is an equivalence relation, and we have $_L(_0) \ _O \ _L(_1)$, $_0 \ _O \ _1$. Indeed, since $_L(_0) \ _O \ _L(_1)$ we know that there exists $\in O$ such that $_L(_0) = _L(_1=n_1)$ if we write $n( ) = n(L)n_1$. Then, since $O_R(L)$ only contains 1, we can say w.l.o.g that $_0 = _1=n_1$ which implies that $_0\overline{_1}=n(L) \in O$. Thus, we have showed that $_2 : \ 7! \ _L( )$ is a bijection between $L=_O$ and $\mathrm{Cl}_O(O_1)$. Then, it remains to show that $_1 : (C : D) \ 7! \ (C + !_S D)$ is a bijection between $\mathrm{P}^1(\mathbb{Z}=N\mathbb{Z})$ and $L=_O$. First, $_1$ is well-de ned. It stems from $C + !_S D \in O_1 = O_L(L)$. Then, $_1$ is injective. Indeed, if not, there exist $_1; _2 \in S$ such that $:= _1\overline{\ }\overline{_2}$ is in O. Let us rewrite $= n( )\ _1\overline{\ }_2 \in S$. Since N is inert in S, we can assume without loss of generality that $n( )$ is coprime to N. Otherwise, this would imply that either $_1$ or $_2$ has a norm that is a multiple of N, which contradicts the fact that N is inert in S. Since $O = \mathbb{Z} + I$ by Proposition 2.3.1, there must be some such that $(x \ ) + !_S y$ is in I and has its norm divisible by N. A necessary condition is that we can nd $\in \mathbb{Z}$ such that the norm of is divisible by N. Looking at the norm of , we see that this is possible only if $X^2 \ tr( )X + n( ) = 0$ has a solution in $\mathbb{Z}=N\mathbb{Z}$. The discriminant of this equation is $4 \ _S y^2 n( )^2$, and it is not a square since N is inert in S. Thus, there are no solutions to the equation, and this su ces to prove the injectivity of our map. Bijectivity follows from a counting argument. We know that $j\mathrm{P}^1(\mathbb{Z}=N\mathbb{Z})j = N + 1$ and we can show that $jL=_O j = j\mathrm{Cl}_O(C)j \ N + 1$. This last bound is a consequence of Proposition 2.3.11, which implies that $j\mathrm{Cl}_O(C)j$ is bounded by the number of $O_R(L)$-ideals of norm N. There are exactly $N+1$ such ideals (this is easy to see for instance by looking at the number of corresponding N-isogenies). Thus, we have showed that $_1$ and $_2$ are bijective maps. It is clear that their composition is , hence the result. □

Remark 2.3.13. Proposition 2.3.12 fails when O_C contains non-trivial automorphisms. Intuitively, this can be explained because the map $_I$ of Lemma 3.2.1 is not injective (up to signs) anymore. If is such an automorphism, taking $\in O$ is another solution to obtain equivalence in Proposition 2.3.11. In this case, we see that $[K]\ I$ and $[J]\ I$ are the same ideals up to multiplication by an automorphism. This justi es that the number $\mathrm{Cl}_O(C)$ is basically equal to $N + 1$ divided by the number of automorphisms (up to sign). The number of such exceptions depends on the value of p mod 12, and is at most 2.

We conclude this section by interpreting Eichler order's class set by putting $\mathrm{Cl}(O)$ in bijection with elements over the geometric world of elliptic curves. Remark 2.3.13 suggests that we consider isogenies up to isomorphisms. For j a supersingular j-invariant in $S(p)$, we write $\mathrm{Aut}(j)$ for the automorphism group of curves of j-invariant equal to j. With this, $\mathrm{isog}(N; j)=\mathrm{Aut}(j)$ is the set of N-isogenies whose domain has j-invariant equal to j up to composition with the elements of $\mathrm{Aut}(j)$.

Proposition 2.3.14. $\mathrm{Cl}(O)$ *is in bijection with the set* $\bigsqcup_{j \in S(p)} \mathrm{isog}(N; j)=\mathrm{Aut}(j)$.

Proof. We already mentioned (see Table 2.1) the bijection identifying a class $C \in \mathrm{Cl}(O_1)$ with a supersingular invariant j_C corresponding to the isomorphism

class of some elliptic curve E_C. This bijection is obtained by $E_C = E_0/E_0[J]$ for any $J \in C$. Similarly, if we take a class $C \in \mathrm{Cl}_O(O_1) \simeq \mathrm{Cl}(O)$ following Definition 2.3.10, and $J \in C$, we associate C with the isogeny φ_C between the pair of supersingular elliptic curves $(E_C; F_C)$ defined as $E_C = E_0/E_0[J]$ and $F_C = E/E[K]$ with $K = [I]J$. By the properties of pushforward isogenies, E_C and F_C are indeed N-isogenous, and we have $\varphi_C = [\varphi_J] \circ \varphi_I$ for any $J \in C$. By Propositions 2.3.11 and 2.3.12 and Remark 2.3.13, classes in $\mathrm{Cl}_O(O_1)$ can be associated with the set of O_C-ideals of norm N up to left multiplication by an automorphism of O_C. It is clear that this is in bijection with the set of N-isogenies up to isomorphisms under the Deuring correspondence. □

2.3.2 Non-Gorenstein orders

Contrary to the previous section, our goal here is not to provide an exhaustive analysis of non-Gorenstein orders, i.e., the orders of Brandt Invariant $f(O) \neq 1$, or their ideals under the Deuring correspondence, but rather to highlight the link between an order O and its Gorenstein closure $\mathrm{Gor}(O)$ and see what are the consequences over the world of elliptic curves when we translate everything under the Deuring correspondence. Our first result Lemma 2.3.15 is about quaternion orders and ideals, and we later reinterpret it under the Deuring correspondence.

Lemma 2.3.15. *Let $D \geq 1$ be an integer coprime to p and let $\mathcal{O}_0$ be a quaternion order of $B_{p;\infty}$ of discriminant coprime to D. The order $\mathcal{O} = \mathbb{Z} + D\mathcal{O}_0$ is optimally embedded in a maximal order $\mathcal{O}'$ if and only if there exists a cyclic left $\mathcal{O}'$-ideal I of norm D such that $\mathcal{O}_0$ is optimally embedded inside $\mathcal{O}_R(I)$.*

Proof. We start by proving the forward direction when D is prime. Let us take $\iota : \mathbb{Z} + D\mathcal{O}_0 \hookrightarrow \mathcal{O}'$. By tensor product by $\mathbb{Q}$ we can extend ι to $\mathcal{O}_0$ and we get the image $\iota(\mathcal{O}_0)$ as an order of $B_{p;\infty}$. We set $I = \{x \in \mathcal{O}; x\iota(\mathcal{O}_0) \subset \mathcal{O}'\}$. First, it is easy to verify that I is an integral left $\mathcal{O}'$-ideal since it is contained in $\mathcal{O}'$. Then, we are going to see that it has norm D. It suffices to show that $D\mathcal{O} \subset I \subset \mathcal{O}'$. To see that $I \neq \mathcal{O}'$, it suffices to note that $1 \notin I$ since $\iota(\mathcal{O}_0) \not\subset \mathcal{O}'$ (since the embedding is optimal). Then, with $D\,\iota(\mathcal{O}_0) \subset \mathcal{O}'$ we have $D\alpha\,\iota(\mathcal{O}_0) = \alpha D\,\iota(\mathcal{O}_0) \subset \mathcal{O}'$ for every $\alpha \in \mathcal{O}'$, which proves that $D\mathcal{O}' \subset I$. Finally, to prove that $D\mathcal{O}' \neq I$, we take $x_0 \in \iota(\mathcal{O}_0)$ and not contained in $\mathcal{O}'$. It is clear that $Dx_0 \in I$, but $Dx_0 \notin D\mathcal{O}'$. Finally, from the definition of I it is clear that $\iota(\mathcal{O}_0)$ is contained in $\mathcal{O}_R(I)$. If this embedding is not optimal, there exists $\mathcal{O}_0'' \supsetneq \mathcal{O}_R(I)$ such that $\iota(\mathcal{O}_0) = \mathbb{Z} + \ell\mathcal{O}_0''$. We have $D\mathcal{O}_0'' \subset I \subset \mathcal{O}'$ and so we would get $\iota(\mathcal{O}) = \mathbb{Z} + \ell(\mathbb{Z} + D\mathcal{O}_0'')$ which would contradict the fact that $\mathcal{O}$ is optimally embedded. This concludes the proof for the prime case D. To extend the proof to the composite settings, we can apply the previous proof to $\mathcal{O} = \mathbb{Z} + \ell(\mathbb{Z} + (D/\ell)\mathcal{O}_0)$ for any prime ℓ dividing D. At each given iteration, we will obtain an ideal of norm ℓ (and the fact that $\mathcal{O}$ is optimally embedded rules out the case where $\iota(\mathcal{O}_0) \subset \mathcal{O}'$). In the end, multiplying all these ideals together, we obtain an ideal of norm D between $\mathcal{O}'$ and a maximal ideal containing $\iota(\mathcal{O}_0)$. The final ideal is cyclic, as otherwise we could divide by some constant $d' \mid D$ and obtain that $\mathbb{Z} + (D/d')\mathcal{O}_0$ is embedded inside $\mathcal{O}'$.

For the other direction, we write I for the $\mathcal{O}'$-ideal of norm D. We write $\mathcal{O}''$ for $\mathcal{O}_R(I)$ and we have the embedding $\iota : \mathcal{O}_0 \hookrightarrow \mathcal{O}''$. First, it is easy to see that $D\,\iota(\mathcal{O}_0) \subset D\mathcal{O}'' \subset I \subset \mathcal{O}'$ implies that $\mathbb{Z} + D\mathcal{O}_0$ is embedded inside $\mathcal{O}'$

through . We now need to show that this embedding is optimal. Let us assume that it is not the case. Then, there exists O_0' O' such that $(O) = \mathbb{Z} + \grave{}O_0'$. If $\gcd(\grave{}; D) = 1$, then, by unicity of the Gorenstein conductor, there exists O_0'' such that $O_0' = \mathbb{Z} + DO_0''$. It is easily veri ed that we must have $(O_0) = \mathbb{Z} + \grave{}O_0''$. And since $\mathbb{Z} + D(\mathbb{Z} + \grave{}O_0'')$ 2 $\mathbb{Z} + I$ we also have O_0'' 2 $O_R(I) = O''$. Hence, this contradicts that O_0 is optimally embedded in O'. If $\grave{}$ and D are not coprime we must have $\grave{} = D$ and so we must have (O_0) 2 O'. This means that (O_0) O' O''. But $O' \setminus O''$ is an Eichler order of discriminant Dp, and this contradicts the assumption that $\mathrm{disc}(O)_0$ is coprime to D. We can apply the same argument recursively to treat the composite case. □

When the order O_0 is maximal, Lemma 2.3.15 gives a su cient and necessary condition for two elliptic curves to be D-isogenous under the Deuring correspondence. This is Proposition 2.3.16. Note that the concrete embedding $\mathbb{Z} + D\,\mathrm{End}(E_1)$,! $\mathrm{End}(E_2)$ is realized by the isogeny $' : E_1$! E_2 of degree D, whose existence is predicted by Proposition 2.3.16. We get that for 2 $\mathrm{End}(E_1)$, the image of $d + D$ under this embedding is given by the endomorphism $d + '$ ^. Proposition 2.3.16 is the main theoretical result behind the suborder representation that we will introduce in Section 4.3. It is also connected with the encryption scheme SETA that we present in Chapter 6. Proposition 2.3.16 is a direct reinterpretation of Lemma 2.3.15 under the Deuring correspondence.

Proposition 2.3.16. *Let D 1 be coprime to p and $E_1; E_2$ be two supersingular curves. The order $\mathbb{Z} + D\,\mathrm{End}(E_1)$ is optimally embedded inside $\mathrm{End}(E_2)$ if and only if E_1 is D-isogenous to one of E_2 or E_2^p.*

Similarly, with the results from Section 2.3.1, we can apply Lemma 2.3.15 to O_0, an Eichler order of level M coprime to D, to enhance Proposition 2.3.16 with subgroups $G_1; G_2$. We a curve E and subgroup of points G, we write $\mathrm{End}(E; G)$ for the set of endomorphisms that leave stable G. By the results of Section 2.3.1, $\mathrm{End}(E; G) = \mathbb{Z} + I(G)$.

Proposition 2.3.17. *Let $D > 1; M > 1$ be coprime to p and also pairwise coprime. Let $E_1; E_2$ be two supersingular curves and $G_1; G_2$ be subgroups of order M inside $E_1[M]; E_2[M]$ respectively. The order $\mathbb{Z} + D\,\mathrm{End}(E_1; G_1)$ is optimally embedded inside $\mathrm{End}(E_2; G_2)$ if and only if E_1 is D-isogenous to one of E_2 or E_2^p and the group G_2 (or (G_2)) is the image of G_1 through a D-isogeny.*

2.4 Quadratic orders in the Deuring correspondence and the number of orientable curves

In this section, we consider embedding of quadratic orders inside endomorphism rings of supersingular curves. For that, we will use the slightly di erent terminology of *orientations*, inspired by Colo and Kohel in [CK19]. The notion of orientation in De nition 2.4.1 below corresponds to the one of *primitive orientations with a p-orientation* in [CK19] and it is equivalent under the Deuring correspondence to *optimal embeddings* of quadratic orders inside maximal orders

of $B_{p;1}$ (see Definition 1.2.18). The same notion is referred to as normalized optimal embeddings in [Bel08].

For the rest of this section, we fix a quadratic imaginary field K and a quadratic order O in K.

Definition 2.4.1. For any elliptic curve E, a K-orientation is a ring homomorphism $\iota : K \hookrightarrow \mathrm{End}\,E \otimes \mathbb{Q}$. A K-orientation induces an O-orientation if $\iota(O) = \mathrm{End}\,E \cap \iota(K)$. In that case, the pair (E, ι) is called a O-oriented curve and E is an O-orientable curve.

In what follows, we consider the elements of $S(p)/\simeq$ rather than $S(p)$ because the Frobenius creates two orientations (one in E and one in $E^{(p)}$) from each optimal embedding of O in a quaternion maximal order of $B_{p;1}$. Note that this is not the convention taken in [Onu21, Wes21] where orientations are not considered up to Galois conjugacy.

Definition 2.4.2. $S_O(p)$ is the set of O-oriented curves (E, ι) up to isomorphisms and Galois conjugacy.

The following proposition follows from the results proven by Onuki [Onu21, Proposition 3.2, Proposition 3.3, Theorem 3.4] and gives a way to compute $\#S_O(p)$.

Proposition 2.4.3. *The set $S_O(p)$ is not empty if and only if p does not split in* K *and does not divide the conductor of* O*. When this condition is satisfied, and p is not ramified in* K*, we have $\#S_O(p) = h(O)$.*

When p is ramified in K, the situation is a bit more complicated but it can be shown [ACL+22] that

$$\#S_O(p) \in \left\{\frac{1}{2}h(O); h(O)\right\}.$$

When $S_O(p)$ is not empty, the class group Cl(O) acts on O-orientations through an operation that we write $\mathfrak{a} \star (E, \iota) = (E_{\mathfrak{a}}, \iota_{\mathfrak{a}})$. Onuki proved that this group action is free and transitive. This action is computed using isogenies. In fact, the definition of kernel of an ideal we gave as Definition 2.1.3, makes sense if we apply it to a quadratic ideal $\mathfrak{a}$ in a quadratic order O embedded inside End(E). We can define $E[\mathfrak{a}]$ for any $E, \iota \in S_O(p)$ and write $\varphi_{\mathfrak{a}}^{E}$ for the isogeny of kernel $E[\mathfrak{a}]$. We have

$$E_{\mathfrak{a}} = E/E[\mathfrak{a}] \quad \text{and} \quad \iota_{\mathfrak{a}}(x) = \frac{1}{n(\mathfrak{a})}\varphi_{\mathfrak{a}}^{E} \circ \iota(x) \circ \hat{\varphi}_{\mathfrak{a}}^{E}. \tag{2.4.1}$$

Definition 2.4.4. The set of O-orientable curves $E_O(p)$ is the set of curves $E \in S(p)/\simeq$ for which there exists an O-orientation ι with $(E, \iota) \in S_O(p)$.

Contrary to $\#S_O(p)$, we do not have a nice generic formula for $\#E_O(p)$. The hardness of the O-UIP, a generic isogeny problem that we will introduce in Chapter 6, mainly depends on the value of $\#E_O(p)$. This motivates that we focus, in the rest of this section, on computing a generic lower bound of $\#E_O(p)$. Aiming at the cryptographic applications, we look for an effective bound in the cases where the discriminant of O is polynomial in p and both have cryptographic size.

Related work. The number of orientable supersingular curves is related to the number of optimal embeddings of quadratic orders inside maximal orders of the quaternion algebra rami ed at p and ∞ and is also linked with the number of representations of integers by ternary quadratic forms. Both quantities have been studied in the literature but not with the same goal. As far as we know, prior to our work, an e ective bound is only known for a restricted range of discriminants and is due to Kaneko [Kan89]. In [Voi21, Chapter 30], several formulas are given involving sums of these numbers (such as the Eichler class number formula) from which it seems hard to derive a bound. There are also asymptotic results on number of representations by ternary quadratic forms (see for instance [IK21, Chapter 20]) but they rather target the case where the discriminant grows to in nity while p is xed. Our work also shares some similarities with the line of work started by Gross and Zagier [ZG85] on singular moduli and later enriched by Dorman, and Lauter and Viray [Dor87, LV15]. Their results cannot be directly applied to our case because they target simultaneous embeddings of quadratic orders of distinct discriminants while we are going to focus on simultaneous embeddings of quadratic orders with the same discriminant. Nonetheless, some of the techniques developed in these works have inspired part of our analysis.

For the remainder of this section, we x a quadratic order $\mathcal{O}$ of negative discriminant d. We assume that the Legendre symbol $(d/p) \neq 1$ and the conductor $f(\mathcal{O})$ is coprime to p, so we know by Proposition 2.4.3 that $\#E_{\mathcal{O}}(p) > 0$. We start with Section 2.4.1, where we introduce useful results from the literature and show a rst lower bound when $d \geq p$. In Section 2.4.2, we prove our main lower bound in the case where $f(\mathcal{O}) = 1$. We extend this result to the generic case using the expansion property of the isogeny graphs in Section 2.4.3.

2.4.1 A rst result for small discriminants

The main result of this Section is Proposition 2.4.6 that was rst proven by Kaneko in [Kan89]. This proposition allows us to derive interesting results on $E_{\mathcal{O}}(p)$ with Corollaries 2.4.8 and 2.4.9 (Corollary 2.4.8 being the only e ective lower bound on $\#E_{\mathcal{O}}(p)$ prior to this work). Proposition 2.4.6 is obtained by studying the quaternion order generated by two integral elements in $B_{p,\infty}$. The study of these objects already appeared in Section 2.2.1 and will prove of prime importance for the results in Section 2.4.2 as well.

The quaternion order generated by two non-commuting elements. Let us take θ_1, θ_2, two integral elements in $B_{p,\infty}$. We want to look at the order $\mathcal{O}_{1,2} = \langle 1, \theta_1, \theta_2, \theta_1\theta_2 \rangle$. When θ_1 and θ_2 are not commuting, $\mathcal{O}_{1,2}$ is a quaternion order, i.e has rank 4 as a Z-module. In Proposition 2.4.5, we give the classical formula to compute $\mathrm{disc}(\mathcal{O}_{1,2})$. Proposition 2.4.6 is a consequence of this formula and was proven in [Kan89].

Proposition 2.4.5. *[Koh96, Chapter 7] Let $\mathcal{O}_i$ be quadratic orders equal to $\mathbb{Z}[\theta_i]$ for $i = 1, 2$ such that θ_1, θ_2 are not commuting. Let $D_i = \mathrm{disc}(\mathcal{O}_i)$, $t_i = \mathrm{tr}(\theta_i)$ for $i \in \{1, 2\}$ and $s = \mathrm{tr}(\theta_1\theta_2)$, then $\mathrm{disc}(\mathcal{O}_{1,2}) = (D_1D_2(t_1t_2 - 2s)^2)/4$.*

Proposition 2.4.6. *[Kan89, Theorem 2'] Let $\mathcal{O}_i$ be quadratic orders equal to $\mathbb{Z}[\theta_i]$ for $i = 1, 2$ such that θ_1, θ_2 are not commuting. If $\mathcal{O}_1, \mathcal{O}_2$ have respective*

discriminant $f_i^2 d$ *(where* d *is a fundamental discriminant) and are contained inside the same quaternion maximal order* $O \supset B_{p,1}$*, we have that* $p \mid f_i f_j d$*.*

Proof. We can write $\mathcal{O}_i = \mathbb{Z}[\omega_i]$ for two integral elements ω_1, ω_2. We can consider the quaternion order $O_{1,2}$ generated by these two elements because ω_1, ω_2 do not commute. When $D_i = f_i^2 d$, we can rewrite the formula from Proposition 2.4.5 as $(D_1 D_2 - (t_1 t_2 - 2s)^2)/4 = (f_1 f_2 d - t_1 t_2 + 2s)(f_1 f_2 d + t_1 t_2 - 2s)/4$. Without loss of generality we can assume that $t_1 t_2 - 2s > 0$ (as we can replace ω_1 by $-\omega_1$ if needed). The discriminant of a quaternion order is always divisible by p. Since p is prime, we have that p divides either $(f_1 f_2 d - t_1 t_2 + 2s)/2$ or $(f_1 f_2 d + t_1 t_2 - 2s)/2$. The two numbers $(f_1 f_2 d - t_1 t_2 + 2s)$ and $(f_1 f_2 d + t_1 t_2 + 2s)$ have same parity and are divisible by 4 which means that $(f_1 f_2 d - t_1 t_2 + 2s)/2$ and $(f_1 f_2 d + t_1 t_2 - 2s)/2$ are both integers. Since $0 < t_1 t_2 - 2s < f_1 f_2 d$, because $\mathrm{disc}(O)_{1,2} > 0$, both factors are smaller than $f_1 f_2 d$ and so we have that p is always smaller than $f_1 f_2 d$. □

Remark 2.4.7. During the proof for Proposition 2.4.6 we showed that $t_1 t_2 - 2s \equiv f_1 f_2 d \bmod p$. This fact will be useful for what follows in Section 2.4.2.

Proposition 2.4.6 allows us to show interesting properties, including a lower bound on the size of $E_{\mathcal{O}}(p)$ (Corollary 2.4.8) and a bound on the minimal distance between two $\mathcal{O}$-oriented curves (Corollary 2.4.9).

Corollary 2.4.8. *When* $|\mathrm{disc}(\mathcal{O})| < p$, $\#E_{\mathcal{O}}(p) = \#S_{\mathcal{O}}(p)$.

Proof. If we assume that $\#E_{\mathcal{O}}(p) < \#S_{\mathcal{O}}(p)$, then there must be a curve E with two distinct $\mathcal{O}$-orientations ω_1, ω_2. Under the Deuring correspondence, this implies that there are two distinct quadratic orders $\mathcal{O}_1, \mathcal{O}_2$ isomorphic to $\mathcal{O}$ contained inside the maximal order $O = \mathrm{End}\, E$. By Proposition 2.4.6 with $f_1 = f_2 = 1$, p must be smaller than d which contradicts our assumption. □

Corollary 2.4.9. *Let* ℓ *be a prime different from* p*. If* ℓ *is inert in* $\mathcal{O}$ *of discriminant* d*, then the shortest chain of* ℓ*-isogenies between two curves of* $E_{\mathcal{O}}(p)$ *has degree larger than* p/d*.*

Proof. (sketch) Let us denote the two curves by E_1, E_2, and by O_1, O_2 their respective endomorphism rings. Let us take $\phi' : E_1 \to E_2$ the smallest chain of ℓ-isogenies connecting them. Let us write $\omega_i \in O_i$ such that $\mathcal{O} = \mathbb{Z}[\omega_i]$. Since ℓ is inert in $\mathcal{O}$, it can be shown that $\omega_1 = \omega_1$ and $\omega_2 = \hat{\phi}' \omega_2 \phi'$ are two elements in O_1 that are not commuting (otherwise, at least one of the isogenies composing ϕ' would be commuting with ω_1, which is impossible since $\ell \nmid p$ and is inert in K). Since $\mathrm{disc}(\mathbb{Z}[\omega_1]) = d$ and $\mathrm{disc}(\mathbb{Z}[\omega_2]) = \deg \phi'^2 d$, we obtain the desired bound by applying Proposition 2.4.6. □

2.4.2 The case of a maximal quadratic order.

In this section, we focus on the case where $\mathcal{O} = \mathcal{O}_K$ for a quadratic imaginary field K of discriminant d. Our main result is Proposition 2.4.19.

To improve the reader's understanding, we divide the proof of Proposition 2.4.19 into several lemmas and propositions. Next, we give a brief outline and some insights into the generic principle. Our starting point is the observation (already used to prove Corollary 2.4.8) that if $\#E_{\mathcal{O}}(p) < \#S_{\mathcal{O}}(p)$, then

there are some curves admitting several O-orientations. Similarly to Proposition 2.4.6, our result is obtained through the analysis of the quaternion orders obtained by combining together the different pairs of orientations. More concretely, we bound the number of these quaternion orders in two very different ways. The first one is a lower bound depending on $\#E_O(p)$ and $\#S_O(p)$ (Proposition 2.4.13) while the second one is an upper-bound (Proposition 2.4.18) that involves an explicit quantity that can be computed from d and p. The combination of these two bounds yields Proposition 2.4.19.

Here are some notations that we will use throughout this section. For any given $E \in E_O(p)$, we write $N_E - 1$ for the number of distinct O-orientations of E. We write $_1;\dots;\ _{N_E}$ for these N_E orientations, they induce the existence of endomorphisms $_1;\dots;\ _{N_E} \in \mathrm{End}\,E$ such that $(\ _i(O))_{1\ i\ N_E} = (\mathbb{Z}[\ _i])_{1\ i\ N_E}$. Since O is the maximal order of K, we can assume that $_i$ is either $_i\ ^1(\frac{p}{q})$ or $_i\ ^1((1+\frac{p}{q})=2)$ where q is the squarefree integer such that $K = \mathbb{Q}(\frac{p}{q})$. Let $I^2_{\neq}(N_E)$ be the set of pairs of distinct unordered elements inside $\{1;\dots;N_E\}$. We define an equivalence relation $_E$ on $I^2_{\neq}(N_E)$ as

$$(i;j)\ _E\ (l;m) \iff \langle 1;\ _i;\ _j;\ _i\ _j\rangle = \langle 1;\ _l;\ _m;\ _l\ _m\rangle:$$

Definition 2.4.10. The set of equivalence classes under $_E$ is denoted by K_E. We write $K_E = \#K_E$.

Intuitively, K_E is the set of distinct quaternion orders obtained by combining two embeddings of O inside $\mathrm{End}\,E$. By definition, we have $\#S_O(p) = \sum_{E \in E_O(p)} N_E$. The quantity we propose to study is

$$K_O(p) = \sum_{E \in E_O(p)} K_E \tag{2.4.2}$$

The link between $\#E_O(p)$ and $K_O(p)$. The number K_E is obviously related to N_E for every curve $E \in E_O(p)$. Intuitively, we would like to say that every pair $_i;\ _j$ generates a different quaternion order $O_{i;j}$ with $1 \le i < j \le N_E$ (thus proving that $K_E = N_E(N_E - 1)=2$). However, even if this seems to be the case with good probability, it is not true in full generality. The correct statement is given in Proposition 2.4.11. Fortunately, Proposition 2.4.11 still allows us to derive Corollary 2.4.12 that lower-bounds K_E by $CN_E(N_E - 1)$ for some constant factor C, which is enough for our purpose.

Proposition 2.4.11. *Let $\mathbb{Z}[\ _1]$, $\mathbb{Z}[\ _2]$, $\mathbb{Z}[\ _3]$ be three quadratic order of discriminant d (with $d > 10$) contained inside a maximal quaternion order O. If $d \equiv 3 \bmod 4$ or $d \equiv 0;1 \bmod p$, then $_3 \in O_{1;2} = \langle 1;\ _1;\ _2;\ _1\ _2\rangle$. When $d \equiv 0 \bmod p$, either $_3 \in O_{1;2}$ or the trace of $_1\ _2$ is equal to $4n(\ _1)$ and $_3$ is one of $(\mathrm{tr}(\ _1)=2 + \ _1\ \ _2)$. When $d \equiv 3 \bmod 4$ and $d \equiv 1 \bmod p$, either $_3 \in O_{1;2}$ or the trace of $_1\ _2$ is $(d - 1)=2$ and $_3$ is one of $((d - 1)=4 + \ _2\ _1\ _2);((d - 1)=4 + \ _1\ \ _1\ _2);((9 - d)=4\ \ _2 + \ _1\ _2);((9 - d)=4\ \ _1 + \ _1\ _2)$.*

Proof. Since all orders are isomorphic, we can assume that all $_i$ have the same trace and norm. Let us write $t = \mathrm{tr}(\ _i); n = n(\ _i)$ for any $i = 1;2;3$. The proof is based on the following claim: any $_3 \in O_{1;2}$ corresponds to a solution $x;y;z \in \mathbb{Z}[1=2]$ with $x \equiv y; y \equiv z \in \mathbb{Z}$ to the quadratic equation:

$$q = q(x^2 + y^2) + sxy + z^2(q^2 - s^2=4) \tag{2.4.3}$$

for some integers $s; q$ that we will de ne below. We exclude the trivial solutions $(1; 0; 0)$ and $(0; 1; 0)$.

As a consequence, our proof can be divided in two parts: rst, we prove the correspondence between the solutions to Eq. (2.4.3) and the $_3 \ 2 \ O_{1;2}$, then we nd the solutions of Eq. (2.4.3) to identify all the possible $_3$.

Proof of the claim. If we assume that $_3 \ 2 \ O_{1;2}$, then there exists $v; x; y; z \ 2$ Z such that $_3 = v + x \ _1 + y \ _2 + z \ _1 \ _2$. The trace of $_3$ implies the equation $t = 2v + t(x + y) + z \ \mathrm{tr}(\ _1 \ _2)$. Thus, we rewrite $_3 = t=2 + x(\ _1 \ t=2) + y(\ _2 \ t=2) + z(\ _1 \ _2 \ \mathrm{tr}(\ _1 \ _2)=2)$.

There are two di erent cases, depending on the value of $d \mod 4$. If $d \ 0 \mod 4$ then $d = 4q$ for some square-free $q \ 1 \mod 4$ and so we can assume w.l.o.g. that $t = 0$ and $_i = !_i$ with $!_i^2 = q$. Else $d = q$ for some square-free $q \ 3 \mod 4$ and we can take $t = 1$, $_i = (1 + !_i)=2$ with $!_i^2 = q$.

In both cases, let us write $s = \mathrm{tr}(!_1 !_2)$. If $d \ 0 \mod 4$, then we obtain the norm equation $q = n(\ _3) = q(x^2 + y^2) + sxy + z^2(q^2 \ s^2=4)$. When $d \ q \ 3 \mod 4$, we have $_3 = (1+!_3)=2 = 1=2+x!_1=2+y!_2=2+(z=4)(1+!_1+!_2+!_1!_2 \ (1 + s=2))$. Then, we obtain $!_3 = (x + z=2)!_1 + (y + z=2)!_2 + z=2(!_1!_2 \ s=2)$. Writing $x_2 = x + z=2$, $y_2 = y + z=2$, $z_2 = z=2$ and taking the norm, we obtain the equation $q = n(!_3) = q(x_2^2 + y_2^2) + sx_2y_2 + z_2^2(q^2 \ s^2=4)$.

In conclusion, we need to nd the solutions $x; y; z$ in $\mathbb{Z}[1=2]$ and $x \ y \ 2 \ \mathbb{Z}; y \ z \ 2 \ \mathbb{Z}$ to the quadratic equation $q = q(x^2 + y^2) + sxy + z^2(q^2 \ s^2=4)$ di erent from the trivial solutions $(1; 0; 0)$ and $(0; 1; 0)$. The end of the proof is dedicated to the enumeration of all possible solutions.

Finding the solutions of Eq. (2.4.3). Throughout this search, we will use heavily the fact that $jsj < 2q$ (which comes from $\mathrm{disc}(\mathbb{Z})[!_1 !_2] = s^2 \ 4q^2 < 0$). W.l.o.g. we can assume that $s \ 0$. We can directly remove the case $x = y = 0$ as it is clearly impossible to nd a z such that $q = z^2(q^2 \ s^2=4)$.

Our rst step is to nd the possible values of z. Let us rewrite our equation as $q = q(x^2+y^2)+sxy+z^2(q \ s=2)(q+s=2)$. With the bound $(q \ s=2)(q+s=2) \ q=2$ we get that we must have $z \ 2 \ f0; \ 1=2; \ 1g$. This implies that $s \ 0 \mod 2$ (otherwise $q(x^2+y^2)+sxy+z^2(q \ s=2)(q+s=2)$ would not be an integer). With that additional information, we can actually show that $q^2 \ s^2=4 \ q$. Thus, in fact we must have $z \ 2 \ f0; \ 1=2g$. We also have $q \ q(x^2 + y^2) + sxy$.

Now that we have greatly reduced the number of possible z, we can look at the values $x; y$. We distinguish two cases, depending on the sign of $x; y$.

Let us assume that $xy \ 0$. Then, we have $q \ q(x^2 + y^2) + sxy$. Thus, we must have $x^2 + y^2 \ 1$. Since we exclude $(x; y) \ 2 \ f(0; 0); (0; 1); (1; 0)g$, the only possibility respecting all our constraints is $(x; y; z) = (1=2; 1=2; \ 1=2)$. Thus, we obtain $q = q=2 + s=4 + q^2=4 \ s^2=16$ which leads to the equation $q^2 \ 2q+s(1 \ s=4) = 0$. The discriminant of the polynomial $X^2 \ 2X+s(1 \ s=4)$ is equal to $4 \ 4s + s^2 = (s \ 2)^2$. The two possible solutions are $s=2$ and $(4 \ s)=2$. The rst one is impossible by the bound $s < 2q$. Since $s \ 0$ we obtain $(4 \ s)=2 < 2$ and this is incompatible with the bound $d > 10$.

We have seen that we have no solutions to Eq. (2.4.3) when $xy \ 0$. Let us now consider the case $xy < 0$. W.l.o.g. we can assume that $x > 0$ and $y < 0$. Then, we have $q \ q(x^2 + y^2) \ sjxyj$, but the bound $s < 2q$ leads to the inequality $q(x^2 + y^2) \ sjxyj > q(x^2 + y^2) \ 2qjxyj = q(x + y)^2$. If we want to avoid a contradiction between these two bounds, then we must have $jx + yj < 1$. Since $x \ y \ 2 \ \mathbb{Z}$, the only possibility is $x = \ y$.

When $x = \ y$, we can rewrite Eq. (2.4.3) as

$$q = (q \quad s/2)(2x^2 + z^2(q + s/2)): \tag{2.4.4}$$

Let us study this new equation. Since we have only a few possibilities for z, we can simply see what happens with Eq. (2.4.4) for each value of z. Since the equation is in z^2 there are two cases: $z = 0$ and $z = \ 1/2$.

If $z = 0$, we get $q = x^2(2q \quad s)$ where $x \ 2 \ \mathbb{Z}$ and so the only solution is $x = 1$ and $s = q$ since q is square-free. However, looking at the discriminant of $h1; !_1; !_2; !_1!_2i$ with Proposition 2.4.5, we get that p must divide q as it divides the discriminant of any maximal order in $B_{p;1}$. In that case, we have the solution $(x; y; z) = \ (1; \ 1; 0)$.

If $z = \ 1/2$, the requirement $x \quad z \ 2 \ \mathbb{Z}$ implies that we can write $x = x'/2$ with $x' \quad 1 \mod 2$. Putting all this in Eq. (2.4.4), we get $q = (q \quad s/2)x'^2/2 + 1/4(q \quad s/2)(q + s/2)$ (we recall that $s' := s/2$ is in $\mathbb{Z}$). It is clear that we must have $q \quad s' \quad 0 \mod 2$ so let us write $q \quad s' = 2q$. Our equation becomes $q = q \quad + q_+ = q \ x'^2 + q_+q$ which implies that $q_+ \quad 0 \mod q$. Thus, we must have $q_+ = kq$ for some $k \ 2 \ \mathbb{Z}$ and Eq. (2.4.4) becomes $q = (k + 1)q \quad = q \ (x'^2 + kq \ )$ which can only be satis ed if $x' = 1$ and $q \quad = 1$. In that case, the only possible solution (up to signs) is $(x; y; z) = (1/2; \ 1/2; \ 1/2)$ when $s = 2q \quad 4$.

In summary, we have showed that our equations have the non-trivial solutions $(1; \ 1; 0)$ when $d \quad 0 \mod p$ and $s = q$ or $(1/2; \ 1/2; \ 1/2)$ when $d \quad 3 \mod 4$ and $s = 2q \quad 4$ and none otherwise. We conclude the proof by computing what are the corresponding values of $\ _3$, so we compute the concrete values $v; x; y; z \ 2 \ \mathbb{Z}$ such that $\ _3 = v + x \ _1 + y \ _2 + z \ _1 \ _2$ (note that these values $x; y; z$ are not directly the solutions to Eq. (2.4.3) when $d = 3 \mod 4$, see the proof of our claim at the beginning of the proof).

For the rst solution, selecting the value v to verify the trace equation we get that $\ _3 = \ \mathrm{tr}(\ _1) + \ _1 \quad \ _2$. It is easily veri ed that $\mathrm{tr}(\ _1 \ _2) = 4n(\ _1 \ _2)$ when $s = q$. Otherwise, $\ _3 = v + x \ _1 + y \ _2 + z \ _1 \ _2$ can only have a solution when $d \quad q \quad 3 \mod 4$ and $\mathrm{tr}(\ _1 \ _2) = 1/2 + 1/4(\mathrm{tr}(!_1!_2)) = (d \quad 1)/2$. By computing the discriminant of $Zh1; \ _1; \ _2; \ _1 \ _2i$ with Proposition 2.4.5 when $\mathrm{tr}(\ _1) = \mathrm{tr}(\ _2) = 1$ and $\mathrm{tr}(\ _1 \ _2) = (q \quad 1)/2$, we see that $\quad = d \quad 1$ and so p divides $d \quad 1$. This proves that $d \quad 1 \mod p$ is also a necessary condition for our equation to be satis ed.

The other possibilities for $\ _3$ can easily be found by taking $(x + z/2; y + z/2; z/2) = \ (1/2; \ 1/2; \ 1/2)$ and v be such that $\mathrm{tr}(\ _3) = 1$. □

Corollary 2.4.12. $K_E \quad \frac{N_E(N_E \quad 1)}{12}$.

Proof. From $_I \ 2 \ O_{i;j}$ or $\ _m \ 2 \ O_{i;j} \ )$ $(i; j) \ 6_E \ (I; m)$ for any $i; j; m; I$, we see from Proposition 2.4.11 that the cardinality of any equivalence class in $I^2_{\mathfrak{S}}(N_E)$ must be smaller than 6, as there at most four elements $_{i'}$ contained in $O_{i;j}$ and we must choose two among the possible i' to get a full quaternion order. This bound combined with $\#I^2_{\mathfrak{S}}(N_E) = N(N \quad 1)/2$ gives the result directly. □

The bound obtained in Corollary 2.4.12 is the key ingredient to the inequality between $\#E_{\mathcal{O}}(p)$, $\#S_{\mathcal{O}}(p)$ and $K_{\mathcal{O}}(p)$ in Proposition 2.4.13.

Proposition 2.4.13. $K_{\mathcal{O}}(p) \quad \frac{1}{12}\left(\frac{\#S_{\mathcal{O}}(p)^2}{\#E_{\mathcal{O}}(p)} \quad \#S_{\mathcal{O}}(p)\right)$

Proof. We have $\#S_O(p) = \sum_{E \in E_O(p)} N_E$. Using Corollary 2.4.12, we get $\sum_{E \in E_O(p)} K_E \quad (1/12) \sum_{E \in E_O(p)} (N_E^2 \quad N_E)$. Then, we can use the classical inequality $\sum_{i=1}^{n} x_i^2 \quad (1/n)(\sum_{i=1}^{n} x_i)^2$ to get the result. □

A generic upper-bound of $K_O(p)$. If $(i; j)$ is a representative of a class $k \in K_E$, we define t_k as the value of $\mathrm{tr}(\ _i\ _j)$ and O_k as the quaternion order equal to the image of $\langle 1; \ _i; \ _j; \ _i\ _j \rangle$ under the isomorphism between $B_{p;\infty}$ and $\mathrm{End}\, E \quad \mathbb{Q}$ (by definition of K_E, t_k and O_k are independent of a choice of $i; j$). The idea is to look at the embedding number of the different orders O_k for $k \in K_E$ and $E \in E_O(p)$ in order to rewrite $\sum_{E \in E_O(p)} K_E$. With the notation from Section 1.2.2, we write this number $e(O_k)$ for a given class k and we compute it in Proposition 2.4.15. Before proving this result, we need to understand a bit better the structure of the orders O_k. This is the purpose of Lemma 2.4.14.

Lemma 2.4.14. *Let E be a curve in $E_O(p)$ and $k \in K_E$. The order O_k is a Bass order.*

Proof. One of the several equivalent definitions of Bass orders inside $B_{p;\infty}$ is that they contain a maximal order inside a commutative subalgebra of $B_{p;\infty}$ [Voi21, Section 24.5]. Since O is the maximal order of K, and the property follows from the definition of O_k. □

With the knowledge that the O_k are Bass orders, we can use Proposition 1.2.10 and Proposition 1.2.21 to compute $e(O_k)$.

Proposition 2.4.15. *Let $D_k = \mathrm{disc}(O_k)/p$. The embedding number of O_k is*

$$e(O_k) = \prod_{\ell \in P_{D_k};(d/\ell)=1} (v_\ell(D_k) + 1) \prod_{\ell \in P_{D_k};(d/\ell)=0;\ell \neq p} 2.$$

Proof. If we show that when ℓ is a prime dividing D_k, $(O/\ell) = (d/\ell)$, then the result follows from Proposition 1.2.21 and Lemma 2.4.14. First, note that when $\ell = p$, the local embedding number $e_\ell(O)$ is always equal to 1 (it is a consequence of Propositions 1.2.10 and 1.2.21 and the fact that $(d/p) \neq 1$). Then, it suffices to prove the result for the cases where $\ell \neq p$ is either split or ramified in K. The two results $(d/\ell) = 1 \Rightarrow (O/\ell) = 1$ and $(O/\ell) = 0 \Rightarrow (d/\ell) = 0$ are easily implied by Proposition 1.2.10. To conclude, it suffices to show $(O/\ell) = 0 \Leftarrow (d/\ell) = 0$, as $(O/\ell) = 1 \Rightarrow (d/\ell) \in \{0; 1\}$. For that, we will show that ℓ divides the discriminant of every $\in O_k$. We recall that there exists $_i; \ _j$ with $O = \mathbb{Z}[\ _i] = \mathbb{Z}[\ _j]$ and $O_k = \langle 1; \ _i; \ _j; \ _i\ _j \rangle$. By assumption, the property is satisfied for $_i; \ _j$. We recall the value of $D_k = (d^2 \quad (\varepsilon \quad 2t_k)^2)/4p$ where $\varepsilon = \mathrm{tr}(\ _i)\,\mathrm{tr}(\ _j)$. From $\ell \mid D_k$ and $\ell \nmid d$, we get that ℓ must divide $2t_k \quad \varepsilon$ which implies that $\ell \mid \quad (\ _i\ _j)$. Then, using $\ell \mid d$ and $\ell \mid (2t_k \quad \varepsilon)$, we can conclude that $\ell \mid (x + y\ _i + z\ _j + w\ _1\ _2)$ for any $x; y; z; w \in \mathbb{Z}^4$. □

The value of $\mathrm{disc}(O_k)$ is $(d^2 \quad (\mathrm{tr}(\ _i)\,\mathrm{tr}(\ _j) \quad 2t_k)^2)/4$ by Proposition 2.4.5. It can be shown that $\mathrm{tr}(\ _i)\,\mathrm{tr}(\ _j)$ is a constant that is either 0 or 1 depending only on the value of $d \bmod 4$. Henceforth, we write this constant ε_d. Inspired by the formulation of Proposition 2.4.15, we define the functions

$$D : (t; d; p) \mapsto \frac{(d^2 \quad (\varepsilon_d \quad 2t)^2)}{4p}$$

and

$$e : (t;d;p) \mapsto \prod_{\ell^2 \mid D(t;d;p);(d/\ell)=1} (v_\ell(D(t;d;p)) + 1) \prod_{\ell^2 \mid D(t;d;p)(d/\ell)=0;\ell \neq p} 2:$$

Let us define $T_{\mathcal{O}}(p) = \{t_k | k \in K_E \text{ for } E \in E_{\mathcal{O}}(p)\}$. For each $t \in T_{\mathcal{O}}(p)$, the values $D(t;d;p)$ and $e(t;d;p)$ are well-defined, when p is prime and d is a fundamental discriminant coprime to p.

Proposition 2.4.16. *Let $\mathcal{O}$ be the maximal quadratic order of discriminant d. Then,*

$$\frac{1}{12} \sum_{t \in T_{\mathcal{O}}(p)} e(t;d;p) \quad K_{\mathcal{O}}(p) \quad \sum_{t \in T_{\mathcal{O}}(p)} e(t;d;p)$$

Proof. By definition, we have that for every class k, there exists $t \in T_{\mathcal{O}}(p)$ with $t = t_k$ and $e(\mathcal{O}_k) = e(t;d;p)$. Thus, each class k corresponds to at least one embedding of $\mathcal{O}_k$ inside a maximal order and so we must have $K_{\mathcal{O}}(p) \geq \sum_{t \in T_{\mathcal{O}}(p)} e(t;d;p)$.

The lower bound of $K_{\mathcal{O}}(p)$ is more delicate to obtain. For that, we will need to quantify the maximum number of embedding that corresponds to the same class k. Let us take an element $t \in T_{\mathcal{O}}(p)$. By definition of $T_{\mathcal{O}}(p)$, there exists a curve $E \in E_{\mathcal{O}}(p)$ and a class $k \in K_E$ with $t_k = t$. By definition of the embedding number, there exist $e(t;d;p)$ distinct maximal orders containing $\mathcal{O}_k$. Each of these maximal orders $\mathcal{O}'$ corresponds to the isomorphism class (up to Galois conjugacy) of supersingular curve E' under the Deuring Correspondence. By definition there also exists a class $k' \in K_{E'}$ such that $\mathcal{O}_{k'} = \mathcal{O}_k$.

We will provide an upper bound on the number of these $e(t;d;p)$ classes that are equal. Up to composition with the relevant isomorphisms, we can assume that all the orders $\mathcal{O}_{k'}$ are actually equal (and not simply isomorphic). Let us take $\mathcal{O}^1 \neq \mathcal{O}^2$, maximal orders with $\mathcal{O}_k \subset \mathcal{O}^i$ for $i = 1;2$ and assume that these two embeddings of $\mathcal{O}_k$ lead to the same class k' and curve E'. We must have $\mathcal{O}^1 = \mathcal{O}^2 = \mathrm{End}\, E'$, so let us write $\iota_i : \mathcal{O}^i \to \mathrm{End}\, E$ the isomorphisms. By definition of our equivalence relation, we must have $\iota_1(\mathcal{O}_k) = \iota_2(\mathcal{O}_k)$, which means that $\mathcal{O}_k$ is stable under the isomorphism $\iota_1^{-1} \circ \iota_2 : \mathcal{O}^2 \to \mathcal{O}^1$. This shows that our problem reduces to counting the number of isomorphisms that leave stable $\mathcal{O}_k$ but are not the identity. For that, it suffices to count the number of possible images of the two elements $\iota_i; \iota_j$ such that $k = (i;j)$. Proposition 2.4.11 tells us that there are at most two other elements of same norm and trace as $\iota_i; \iota_j$. Thus, we have $3 \times 4 = 12$ possible image pair for $\iota_i; \iota_j$, and so we can conclude that each class k corresponds to at most 12 distinct embeddings of $\mathcal{O}_k$ inside distinct maximal orders. This proves the result.

□

With Proposition 2.4.16, we have all the necessary ingredients to prove our generic upper-bound of $K_{\mathcal{O}}(p)$. We introduce in Definition 2.4.17, a final notation to simplify the formulation of Proposition 2.4.18.

Definition 2.4.17. The function $\sigma : \mathbb{N} \to \mathbb{N}$ is defined as $\sigma(N) = \prod_{\ell \in P_N} (v_\ell(N) + 1)$, and it computes the number of divisors of N.

Proposition 2.4.18. $K_{\mathcal{O}}(p) \quad \left\lfloor \frac{d+1}{4p} \right\rfloor \max_{0 \leq N \leq d^2/(4p)} \sigma(N)$.

Proof. It is clear from the de nition of the functions $;D;e$ that $(D(d;t;p))$ $e(d;t;p)$. Since 0 $D(t;d;p)$ $(d^2)=4p$ we get that

$$\sum_{t2T_O(p)} e(t;d;p) \quad \#T_O(p) \max_{0 \ N \ d^2=(4p)} (N):$$

Next, we prove that $\#T_O$ $d(d+1)=4pe$. If t 2 T_O, we must have that $D(d;t;p) = \mathrm{disc}(O_k)=p$ 2 N for some class k. The condition on the discriminant yields d^2 $("_d$ $2t)^2$ $0 \mod 4p$ and $d^2 > ("_d$ $2t)^2$. When $t > 0$, we have $2jtj$ $"_d > 0$ and so get the bound $(jdj + "_d)=2 > jtj$. There are two possible values of $t \mod 4p$ and combining that with $0 < jtj < (jdj + "_d)=2$ we obtain at most $b(d+1)=(4p)c$ possible values. Adding $t = 0$, we obtain the desired bound. The proof is concluded by Proposition 2.4.16. □

We obtain a generic lower bound on $\#E_O(p)$ in Proposition 2.4.19. It is a combination of Proposition 2.4.13 and Proposition 2.4.18.

Proposition 2.4.19. *$\#E_O(p)$ $\frac{AB}{A+B}$ $\frac{1}{2}\min(A;B)$ where $A = \#S_O(p)$ and $B = \frac{\#S_O(p)^2}{3(4p+d+1)} \frac{p}{\max_{0 \ N \ d^2=4p} (N)}$.*

Before proving Proposition 2.4.19, we prove a useful lemma.

Lemma 2.4.20. *For every 3 values $x;A;B > 0$ such that x A or x $(1$ $)B$ for every $0 <$ < 1 we have that x $\frac{AB}{A+B}$ $\frac{1}{2}\min(A;B)$.*

Proof. We are going to start with the intermediary result that x A or x $(1$ $)B$ for every $0 <$ < 1 implies that x $\max_{0< <1} \min(A;(1$ $)B)$. The function $7!$ $\min(A;(1$ $)B)$ is increasing on $]0; \ _m]$ and decreasing on $[\ _m;1[$ for the value $_m$ such that $_m A = (1$ $_m)B$. Thus, we get $_m = B=(A+B)$ and $\max_{0< <1} \min(A;(1$ $)B) = \ _m A = AB=(A+B)$.

To conclude it is easy to verify that

$$\frac{AB}{A+B} \quad \frac{1}{2}\min(A;B):$$

□

Proof. (Proposition 2.4.19) We will apply Lemma 2.4.20 with $x = \#E_O(p)$ and the values $A;B$ as in Proposition 2.4.19. Thus, we need to prove that either $\#E_O(p)$ A or $\#E_O(p)$ $(1$ $)B$ for any $0 <$ < 1.

Note that when $\#E_O(p) <$ $\#S_O(p)$ for some 1, we must have $K_O(p) > 0$ because there is at least one curve with two distinct orientations. Thus, Proposition 2.4.13 proves that

$$\#E_O(p) \quad \frac{\#S_O(p)^2 \quad \#S_O(p)\#E_O(p)}{12K_O(p)}:$$

For any 2 $[0;1]$, if $\#E_O(p) <$ $\#S_O(p)$, we have that

$$\#E_O(p) > (1 \quad)\frac{\#S_O(p)^2}{12K_O(p)}:$$

The proof is concluded by Proposition 2.4.18 and $d(d+1)=4pe$ $1 + (d+1)=4p$. □

Remark 2.4.21. Our bound becomes less and less tight when the size of d grows in comparison to p. Asymptotically, we have

$$\lim_{d \to \infty} \frac{\#S_{\mathcal{O}}(p)^2}{3(4p+d+1)} \frac{p}{\max_{0 \le N \le d^2/4p} \Delta(N)} = 0$$

which is very far from the expected $E_{\mathcal{O}}(p) = N_p$ when $d \to \infty$. However, when the value of d is polynomial in p, classical analysis on the Δ function detailed below shows that our bound will never be trivial even as p grows to infinity (see Eq. (2.4.6)). This is typically the case needed for isogeny-based cryptography as illustrated by our numerical application in Section 6.5.4 for a prime $p \approx 2^{400}$ and a discriminant d satisfying $p < d < p^2$.

Remark 2.4.22. Note that we can derive a family of upper bounds on the class number $h(\mathcal{O})$ from Proposition 2.4.19. Indeed, since we have the trivial bound $p/12 + 1 \le N_p \le \#E_{\mathcal{O}}(p)$, in the cases where $A > B$ (which will happen when d is much bigger than p as explained in Remark 2.4.21), we obtain $p/12 + 1 \le B/2$. When we replace $\#S_{\mathcal{O}}(p)$ by the correct value $ch(\mathcal{O})$ (with $c \in \{1/2, 1\}$) in the formula for B we obtain

$$h(\mathcal{O})^2 < \frac{(p+12)(4p+d+1)}{2c^2 p} \max_{0 \le N \le d^2/4p} \Delta(N). \tag{2.4.5}$$

Intuitively, the best bounds should be obtained when $d \approx p$. The estimates we provide on $\Delta(N)$ below does not allow us to conclude that this would lead to an improvement on the state of the art upper bounds on class numbers.

Remark 2.4.23. When p divides d, it might be possible to get better bounds. For instance, when d/p is a prime smaller than $p/4$, a lower bound was proven in [EHL+20, Theorem 3.9], using the fact that a curve in $E_{\mathcal{O}}(p)$ must be d/p-isogenous to its Galois conjugate E^p. Another possibility, is to exploit the fact that when $d \equiv 0 \bmod p$, the element $\ell_1 \ell_2/p$ is integral (see the proof of Proposition 2.4.11 for the definition of ℓ_1, ℓ_2) and so we may be able to consider superorders of the $\mathcal{O}_k$ (which might give a better bound since the discriminants are smaller). While this idea seems promising, it does not appear trivial to obtain the analogue of Proposition 2.4.11 and this is why we left the study of this special case open for future work.

Upper bound on the number of divisor functions. The number of division function Δ is well-studied and generic upper-bounds can be found in the literature. Since Wigert [Wig07], we know that $\Delta(N) = O(N^{\varepsilon})$ for any $\varepsilon > 0$. In 1983, Nicolas and Robin showed that

$$\Delta(N) \le 2^{1 \frac{\log(N)}{\log\log(N)}}, \text{ for any } N \ge 3 \tag{2.4.6}$$

where $\gamma_1 = 1.53793986\ldots$.

More recently De Konick and Letendre [DKL18] proved several new upper-bounds involving $\omega(N)$ the number of distinct prime factors of N. In particular they showed that for any composite $n \ge 2$

$$\Delta(N) \le \left(1 + \gamma_3 \frac{\log(N)}{\omega(N)\log(\omega(N))}\right)^{\omega(N)} \tag{2.4.7}$$

where $_3 = 1;1999953:::$

When $!(N)$ 74 they even prove that

$$(N) \quad 1 + \frac{\log(N)}{!(N)\log(!(N))}^{!(N)} \tag{2.4.8}$$

We will ued this last bound in the numerical application we propose in Section 6.5.4.

2.4.3 The case of non-maximal orders

In this section, we provide an upper-bound on $\#E_O(p)$ when O is not maximal. Most of the ideas exposed below are not fundamentally new, and have been considered for instance by Lubotzky, Phillips and Sarnak [LPS86], but we expose them here for completeness as simple statements matching our needs are rather hard to extract.

For the rest of this section, let us take $O = Z + fO_0$ where $f > 1$ is coprime to p and O_0 is a maximal quadratic order. The fundamental property underlying our result in Proposition 2.4.26 is summarized in Proposition 2.4.24. This is a consequence of [LB20, Lemma 5.4, Corollary 5.5] showed by Love and Boneh (they talk about optimal embeddings inside maximal orders rather than orientations of elliptic curves in their paper but the two notions are the same under the Deuring Correspondence) and the standard characterization of *horizontal, ascending and descending* isogenies (see [CK19, Onu21] for instance).

Proposition 2.4.24. *Let* O *be a quadratic order of conductor* f *and discriminant* d *such that* $E_O(p)$ *is not empty. Let* $\ell \neq p$ *be a prime number. For every curve* $E \in E_O(p)$*, among the* $\ell + 1$ *curves* ℓ*-isogenous to* E*, there are* $\ell - (d/\ell)$ *curves contained in* $E_{Z+\ell O}$*. If* ℓ *is coprime to* f*, the* $1 + (d/\ell)$ *remaining curves are in* $E_O(p)$ *and if not, then the final curve is contained in* $E_{O'}(p)$ *where* O' *is the quadratic order of discriminant* d/ℓ^2 *such that* $O = Z + \ell O'$*.*

Note that by ℓ-isogenous we mean connected by an ℓ-isogeny (that must be cyclic since ℓ is prime). From Proposition 2.4.24, we see that we can tie the generic case to the case of fundamental discriminant using the expansion properties of the isogeny graphs. For simplicity, we assume henceforth that $f = \ell^e$ for some prime ℓ and $e \in \mathbb{N}$ and O_0 is any maximal quadratic order. By Proposition 2.4.24, each coprime factor of the conductor can be treated independently . The exact bound in Proposition 2.4.26 depends on whether ℓ is split, rami ed or inert in K but the three cases can be treated in a similar manner. As a warm-up, we start with Proposition 2.4.25, to give a lower bound on the number of curves that are f-isogenous to a curve in an arbitrary subset E_0 of the isomorphism classes of supersingular curves over $\mathbb{F}_{p^2}$.

For any prime ℓ coprime to p, the graph of cyclic ℓ-isogenies on isomorphism classes of supersingular curves is Ramanujan with degree of regularity equal to $\ell + 1$. For ℓ^e-isogenies, we obtain an almost-Ramanujan graph with degree of regularity equal to $_1(\ell^e) = \ell^e(1+1/\ell)$. We write $A(\ell^e)$ for the adjacency matrix of the graph of cyclic ℓ^e-isogenies on supersingular curves. The matrices $A(\ell^r)$ for $r \in \mathbb{N}$ are related to the Brandt matrices $B(\ell^r)$ for $r \in \mathbb{N}$ under the relation $A(\ell^r) = B(\ell^r) \quad B(\ell^{r-2})$ and $B(\ell) = A(\ell)$ and $B(1) = A(1) = I$. The Brandt matrices $B(m)$ correspond to the action of the Hecke operator T_m on the space

of modular forms of weight 2 on $\Gamma_0(N)$ when m and N are coprime. The graph associated to the $B(\ell^r)$ are $\sum_{i=0}^{r} \ell^i$-regular. The matrices $A(\ell^r)$ and $B(\ell^r)$ are real symmetric positive and have N_p ordered real eigenvalues. The biggest eigenvalue is always equal to the degree of regularity k of the associated graph and the corresponding eigenspace is generated by the vector $(1/\sqrt{N_p})_{1 \leq i \leq N_p}$ of norm 1. The expansion of the graph can be measured by the size of the second eigenvalue. The graph is said to be *Ramanujan* when this value is smaller than $2\sqrt{k-1}$. A consequence of the Riemann hypothesis for function fields, proven by Deligne (see [Kat76] for instance) is that the second eigenvector of $B(\ell^r)$ is smaller than $(r+1)\sqrt{\ell^r}$. When $r = 1$, this proves that the graph of ℓ-isogenies is Ramanujan. For $r > 1$, the bound is not good enough to prove the same thing. This is why the other graphs are to be almost-Ramanujan. We will use $A(\ell^r) = B(\ell^r) - B(\ell^{r-2})$ to deduce results on the expansion of the graph of ℓ^e isogenies.

Proposition 2.4.25. *Let $e \geq 2$ and $\ell \geq 2$ different from p and let E_0 be a non-empty subset of isomorphism classes of supersingular elliptic curves. Let us write $C_0 = \#E_0$ and write E_{ℓ^e} for the set of isomorphism classes of curves ℓ^e-isogenous to a curve of E_0. We have the bound*

$$\#E_{\ell^e} \leq \frac{N_P}{1 + (N_P/C_0 - 1)\frac{(e+1)2\ell^e + (e-1)2\ell^{e-2}}{\lambda_1(\ell^e)^2}}.$$

Proof. Assume that we have an ordering $E_1, \ldots, E_{N_p}$ of all supersingular elliptic curves (up to isomorphism). Let us write $X = (x_i)_{1 \leq i \leq N_p} \in \mathbb{R}^{N_p}$, the vector such that $x_i = 1$ if $E_i \in E_0$ and 0 otherwise. If we write $Y = A(\ell^e)X$, then $\#E_{\ell^e}$ is equal to the number of non-zero entries of Y. A very classical bound tells us that

$$\#E_{\ell^e} \geq \frac{\|Y\|_1^2}{\|Y\|_2^2}.$$

We see easily that $\|Y\|_1 = \lambda_1(\ell^e)C_0$. To upper-bound $\|Y\|_2^2$, we are going to use the Ramanujan property. Let us write $f_1, \ldots, f_{N_p}$ the orthonormal basis of eigenvectors of $A(\ell^e)$. We already explained that $f_1 = (1/\sqrt{N_p})_{1 \leq i \leq N_p}$. We can write $X = z_1 f_1 + X^0$ where X^0 is orthogonal to f_1. The coefficient z_1 is equal to $\langle X, f_1 \rangle = C_0/\sqrt{N_p}$. We have $\|A(\ell^e)X\|_2^2 = \lambda_1(\ell^e)^2 z_1^2 + \|A(\ell^e)X^0\|^2$. We have the equality $A(\ell^e) = B(\ell^e) - B(\ell^{e-2})$. The vector X^0 lies in a space of dimension $N_p - 1$ where all the eigenvalues of $B(\ell^r)$ have absolute value smaller than $(r+1)\sqrt{\ell^r}$ for every $r \geq 1$. We have $\|A(\ell^e)X^0\|_2^2 \leq \|B(\ell^e)X^0\|_2^2 + \|B(\ell^{e-2})X^0\|_2^2$ by the triangular inequality. Thus, $\|A(\ell^e)X\|_2^2 \leq \lambda_1(f)^2 z_1^2 + ((e+1)2\ell^e + (e-1)2\ell^{e-2})\|X^0\|_2^2$. We can easily compute that $\|X^0\|_2^2 = \|X\|_2^2 - z_1^2$. Thus we obtain $\|Y\|_2^2 \leq \lambda_1(\ell^e)^2 C_0^2/N_p + ((e+1)2\ell^e + (e-1)2\ell^{e-2})(C_0 - C_0^2/N_p)$. Thus, we obtain

$$\#E_{\ell^e} \geq \frac{\lambda_1(\ell^e)^2 C_0^2}{\frac{\lambda_1(\ell^e)^2 C_0^2}{N_p} + ((e+1)2\ell^e + (e-1)2\ell^{e-2})(C_0 - C_0^2/N_p)}$$

$$\#E_{\ell^e} \geq \frac{N_P}{1 + (N_P/C_0 - 1)\frac{(e+1)2\ell^e + (e-1)2\ell^{e-2}}{\lambda_1(\ell^e)^2}}.$$

□

Proposition 2.4.26. *Let* $\mathcal{O} = \mathbb{Z} + \ell^e\mathcal{O}_0$ *for some* $\ell \geq 2$ $\mathbb{N}$ *coprime to* p *and a maximal quadratic order* $\mathcal{O}_0$*. Let us write* $C_0 = \#E_{\mathcal{O}_0}$.
If ℓ *is inert in* $\mathbb{K}$*:*

$$\#E_{\mathcal{O}}(p) \quad \frac{N_P}{1 + (N_P/C_0 \quad 1)\frac{(e+1)^2\ell^e + (e \quad 1)^2\ell^{e \quad 2}}{(\ell^e + \ell^{e \quad 1})^2}};$$

else if ℓ *is rami ed:*

$$\#E_{\mathcal{O}}(p) \quad \frac{N_P}{1 + (N_P/C_0 \quad 1)\frac{((e+1)^2\ell^e + e^2\ell^{e \quad 1})}{\ell^{2e}}};$$

else, ℓ *is split and:*

$$\#E_{\mathcal{O}}(p) \quad \frac{N_P}{1 + (N_P/C_0 \quad 1)\frac{\sum_{j=0}^{e}(2^{b(1+j)/2c} \quad 2^{b(1+j \quad 2)/2c})^2(e \quad j+1)^2\ell^{e \quad j+1}}{(\ell^e \quad \ell^{e \quad 1})^2}}:$$

Proof. The case ℓ inert is a simple combination of Proposition 2.4.24 and Proposition 2.4.25 since the set $E_{\mathbb{Z}+\ell^e\mathcal{O}}(p)$ is exactly the set of curves ℓ-isogenous to curves in $E_{\mathcal{O}}(p)$. When ℓ is rami ed or split, the situation is slightly more complicated. In fact, in both cases, the result is obtained by rewriting the reasoning used in the proof of Proposition 2.4.25. Indeed, we are going to show that $\#E_{\mathbb{Z}+\ell^e\mathcal{O}_0}$ is equal to the number of non-zero entries of a vector Y computed as MX where X is de ned as in the proof of Proposition 2.4.25 and M is a liner combination of the $A(\ell^i)$ for $i \in [0; e]$. For Proposition 2.4.25 (and ℓ inert) we can simply take $M = A(\ell^e)$. When ℓ is not inert, we need to remove some of the ℓ^e isogenies and this is why we have a more complicated expression for M.
For what remains, let us assume that the labelling of the N_P-isomorphism classes of supersingular curves is such that $E_1, \ldots, E_{C_0}$ are the C_0 curves contained in $E \in E_{\mathcal{O}_0}(p)$. X_i is the vector of $\mathbb{N}^{N_p}$ such that $(X_i)_j = 1$ if $j = i$ and 0 otherwise and $X = \sum_{i=1}^{C_0} X_i$.
When ℓ is rami ed, Proposition 2.4.24 implies that there exists a permutation $\quad$ of $[1; C_0]$ such that E_i and $E_{\quad(i)}$ are ℓ-isogenous and $\quad^2$ is the identity. To get the curves of $E_{\mathcal{O}}(p)$, we need to exclude all the ℓ^e-isogenies that can be written as $\quad\phi'_i$ where ϕ'_i is the ℓ-isogeny between E_i and $E_{\quad(i)}$. $A(\ell^e)X_i$ gives all the curves that are ℓ^e-isogenous to E_i. To remove the ones that are obtained through the wrong isogenies we can subtract by $A(\ell^{e \quad 1})X_{\quad(i)}$ but with that we have also subtracted the curves that are $\ell^{e \quad 2}$-isogenous to E_i. So we need to compensate by adding $A(\ell^{e \quad 2})X_i$ and iterating this reasoning, we end up with the $\sum_{j=0}^{e}(\quad 1)^j A(\ell^{e \quad j})X_{\quad j(i)}$. Thus, we get $M = \sum_{j=0}^{e}(\quad 1)^j A(\ell^{e \quad j})$ after summing this formula for all $i \in [1; C_0]$. The lower bound on the number of zeroes of $Y = MX$ is given by $\|Y\|_1^2/\|Y\|_2^2$. A simple counting gives that $\|Y\|_1^2 = (\sum_{j=0}^{e}(\quad 1)^j \quad_1(\ell^{e_i}))^2 = \ell^{2e}$. To lower bound $\|Y\|_2^2$, we see that M has the same eigenvector f_1 (see the notations of the proof of Proposition 2.4.25) for the eigenvalue ℓ^{2e}. Thus, we can decompose $X = z_1 f_1 + X'$ with $z_1 = C_0/\sqrt{N_p}$. Once again we replace each $A(\ell^r)$ by $B(\ell^r) \quad B(\ell^{r \quad 2})$ and $B(\ell) = A(\ell)$ and $B(1) = A(1)$. Interestingly, a lot of terms cancel out in M and we end with $B(\ell^e) \quad B(\ell^{e \quad 1})$. As in the proof of Proposition 2.4.25, we conclude with the bound on the eigenvalues of the $B(\ell^r)$ and the triangular inequality. This is how we get $\|MX'\|_2^2 \quad ((e+1)^2\ell^e + e^2\ell^{e \quad 1})\|X'\|_2^2$. The proof is concluded in a

similar way to Proposition 2.4.25.
When ℓ is split, we have by Proposition 2.4.24 that there exists two permutations σ_l, σ_r such that E_i is ℓ-isogenous to $E_{\sigma_l(i)}$ and $E_{\sigma_r(i)}$ and $\sigma_l \circ \sigma_r = \sigma_r \circ \sigma_l$ is the identity. A similar reasoning proves that we can take

$$M = \sum_{j=0}^{\infty} (-1)^j 2^{b(1+j)/2c} A(\ell^{e-j}).$$

Once again, we use our relation between A and B matrices to get a sum on the $B(\ell^{e-j})$. We have $\ell^e(1 - 1/\ell) = \sum_{j=0}^{e} (-1)^j 2^{b(1+j)/2c} \circ_1(\ell^{e-j})$ and the final result follows from the same ideas as before.

□

Chapter 3

Resolution of norm equations in quaternion lattices

In this chapter, we study the problem of finding elements of given norm inside some families of lattices in $B_{p,\infty}$. Overall, we will cover orders whose Gorenstein closure are Eichler orders and their ideals.

The basis for all the content of this chapter have been laid out by Kohel, Lauter, Petit and Tignol in their seminal paper "On the quaternion ℓ-isogeny path problem" [KLPT14]. In particular, they introduced all the building blocks RepresentInteger, StrongApproximation, IdealModConstraint, EquivalentPrimeIdeal and the KLPT algorithm. Our contribution is to show how to reuse and modify these building blocks to expand upon the range of lattices that we are able to treat. The content of this chapter is a mix between some results presented in our publications [DFKL+20, DFLW22, Ler21].

The problem originally targeted by [KLPT14] is called the *Quaternion N-isogeny path problem*.

Problem 3.0.1. *Let $N \in \mathbb{N}$. Given a maximal order $\mathcal{O}$ and I, a left $\mathcal{O}$-ideal. Find $J \subset I$ of norm in N.*

In Section 2.2, we have already mentioned the importance of the KLPT algorithm and we will provide in Chapter 4, and later in Chapter 5, new reasons behind our interest in these algorithms. Motivated by these applications, our goal is to obtain practical and efficient algorithms. Most of the proofs and analysis below are relying on plausible heuristics that have been verified experimentally. When a result bears the mention (UPHA), it means that it holds *under plausible heuristic assumptions*. The Generalized Riemann Hypothesis (GRH) will often be among those assumptions but it is not the only one.

We imitate the formulation of Problem 3.0.1 and index our algorithms by a set N to indicate that it will look for solutions whose norm is contained in N. In all our algorithms, we will always strive to minimize as much as possible the norm of the output without impacting too much the performances. This principle might be worth keeping in mind to understand some of the choices that we are going to make.

If $c \in \mathbb{N}$, we define $cN = \{cn; n \in N\}$. For simplicity, in our algorithms, we will also assume that the set N is such that if $n \in N$, then all dn are also contained in N. This will be true for the sets that we will use in practice. One example of such set is $S = \{\ell^e; e \in \mathbb{N}\}$. Another is $D(M)$, the sets of divisor of M. When $n_1 \in N$, we write $N/n_1 = \{n/n_1; n \in N \text{ and } n/n_1 \in \mathbb{N}\}$.

Remark 3.0.2. In fact, the authors of [KLPT14] only considered the case where $N = S$, but variants where the elements can have powersmooth norm were later introduced by [GPS17]. As the method of resolution is quite similar for any N, we consider the generic case.

In Sections 3.1 and 3.2, we mostly present the results from [KLPT14] leading to the KLPT algorithm. We introduce our contributions in Sections 3.3 and 3.4.

3.1 The case of special extremal orders

One notion that will be of great importance to us in this chapter and the next is the one of *special extremal order* from [KLPT14]. We will keep the notations introduced in Definition 3.1.1 throughout this chapter. We use the basis $\langle 1, i, j, k \rangle$ for $B_{p,\infty}$ where $i^2 = -a$ for some integer $a > 0$, $j^2 = -p$ and $k = ij = -ji$.

Definition 3.1.1. Let $\mathcal{O}$ be a maximal order in $B_{p,\infty}$. We say that $\mathcal{O}$ is a $\mathcal{S}$*-special extremal* if the unique two-sided ideal of norm p in $\mathcal{O}$ is equal to $\mathcal{O}j_{\mathcal{O}}$ for some $j_{\mathcal{O}} \in \mathcal{O}$ of norm p and there is a quadratic order $\mathbb{Z}[\omega] \hookrightarrow \mathcal{O}$ of discriminant Δ such that the orthogonal complement of $\mathbb{Z}[\omega]$ (with respect to the inner product of $B_{p,\infty}$ defined in Eq. (1.2.4)) is $\mathbb{Z}[\omega]j_{\mathcal{O}}$. For each type of special extremal orders, we consider the canonical representation as the one satisfying $j_{\mathcal{O}} = j$ and $\mathbb{Z}[\omega] \subset \mathbb{Q}(i)$. The generator of $\mathbb{Z}[\omega]$ is written ω and the norm form of $\mathbb{Z}[\omega]$ is noted $f_{\mathbb{Z}[\omega]}$. The *special extremal* order of $B_{p,\infty}$ is a $\mathcal{S}$-special extremal order with smallest possible Δ (arbitrarily chosen among all the at most 2 types of $\mathcal{S}$-special extremal order [CPV20, Theorem 26]).

Remark 3.1.2. We will denote by $\mathcal{O}_0$ the special extremal order of $B_{p,\infty}$ and q is the square-free integer such that $\mathbb{Q}(\sqrt{-q}) = \mathbb{Q} \cap \mathcal{O}$. In fact, we have already encountered one of those orders in the beginning of Section 2.1.3. Indeed, when $p = 7 \bmod 12$, the smallest possible Δ is 4 and we can take the canonical $\mathcal{S}$-special extremal order to be

$$\mathcal{O}_0 = \langle 1, i, \frac{i+j}{2}, \frac{1+k}{2} i \rangle \tag{3.1.1}$$

and we have $q = 1$. We saw that this maximal order corresponded to the supersingular curve E_0 of j-invariant 1728. Examples of special extremal orders for other primes are given in [KLPT14]. Under GRH, a result of Ankeny [Ank52] proves that $q = O(\log(p)^2)$. We will use this estimate for our complexity analysis throughout this chapter. We call the *special-extremal curve* over $\mathbb{F}_{p^2}$ to be the curve (up to isomorphism) whose endomorphism ring is isomorphic to the special extremal order of $B_{p,\infty}$. Note that this isomorphism class is well-defined since it is defined over $\mathbb{F}_p$ because the two-sided ideal of norm p is principal.

Lemma 3.1.3. *The suborder* $\langle 1, \omega, j, j\omega \rangle$ *has index* $\mathrm{disc}(\mathbb{Z}[\omega])$ *in* $\mathcal{O}_0$ *and its norm form is given by*

$$f_0 : (t, x, y, z) \mapsto f_{\mathbb{Z}[\omega]}(t, x) + p f_{\mathbb{Z}[\omega]}(y, z). \tag{3.1.2}$$

The idea from [KLPT14] is that it is relatively easy to solve norm equations in the suborder $\langle 1, \omega, j, j\omega \rangle + I \subset O_0$ given by the norm form f_0 above. Their algorithm RepresentInteger to perform that task is presented later in this section, but we need first to find solutions to the binary quadratic equations of the form $f_O(t, x) = m$. In Section 3.1.1, we present Cornacchia's algorithm to solve that problem.

3.1.1 Cornacchia's algorithm

We present, as Algorithm 1, Cornacchia's algorithm [Cor08] to solve quadratic equations of the form $t^2 + qx^2 = m$ in $\mathbb{Z}$. When there is a solution for a given m, q, Cornacchia will find it. The efficiency mainly depends on the hardness of a square root computation $\bmod m$. This operation basically comes down to factoring m. In particular, when m is prime or a near-prime, the factorization is easy to find and so Cornacchia will be efficient. To preserve the efficiency, we assume that Cornacchia starts by checking if m is prime or near-prime with an algorithm TrialFactorization. When m satisfies the condition, TrialFactorization outputs the factorization of m and the computation proceeds, otherwise, Cornacchia outputs $?$.

Algorithm 1 Cornacchia

Input: $q, m \in \mathbb{N}$.
Output: $?$ or $t, x \in \mathbb{Z}$ such that $t^2 + qx^2 = m$.

fac ← TrialFactorization(m).
if fac $= ?$ then
  Return $?$.
end if
Determine if q is a square $\bmod m$ using fac.
if q is not a square. then
  Return $?$.
end if
Compute r_2 a square root of $q \bmod m$ using fac.
Set $r_1 = m$ and $i = 1$.
repeat
  $i \leftarrow i + 1$
  $r_i \leftarrow r_{i-2} \bmod r_{i-1}$.
until $r_i^2 < m < r_{i-1}^2$:
if $m - r_i^2 \not\equiv 0 \bmod q$ then
  Try again with another square root of $q \bmod m$ and if it is not possible return $?$.
end if
$t \leftarrow r_i$, $x \leftarrow \sqrt{(m - t^2)/q}$
if $x \notin \mathbb{Z}$ then
  Try again with another square root of $d \bmod m$ and if it is not possible, return $?$.
end if
return t, x

If q is fixed, the probability that there exists a solution for some random m is

proportional to $1/h(\mathcal{O}_{\mathbb{Q}(\sqrt{-pq})})$. The expected running time is $O(\text{poly}(\log(qm)))$.

3.1.2 Representing integers by the norm form of the special extremal order.

Once we have Cornacchia to solve efficiently norm equations of the form $f_{\mathcal{O}}(t,x) = m$, we can devise easily an efficient algorithm to find solutions to $f_0(t,x,y,z) = M$. The generic idea used both in RepresentInteger and StrongApproximation is to sample y,z in some randomized way and try to see if we can find t,x such that $f_{\mathcal{O}}(t,x) = M - pf_{\mathcal{O}}(y,z)$ with Cornacchia. The algorithm RepresentInteger was introduced by the authors of [KLPT14] and uses this idea quite straightforwardly to find elements of a given norm.

Algorithm 2 RepresentInteger$_N$

Input: N ∈ ℤ such that there exists $M \in N\mathbb{Z}$ with $M > p$
Output: $\gamma = t + xi + yj + zij$ with $n(\gamma) \in N\mathbb{Z}$.

1: Set $M = fg$.
2: Select the smallest $M \in N\mathbb{Z}$ or M such that $M > p$.
3: Set $m = \left\lfloor \sqrt{\frac{M}{p(1+q)}} \right\rfloor$ and sample random integers $y,z \in [-m,m]^2$. Set $M' = M - pf_{\mathcal{O}}(y,z)$.
4: If Cornacchia$(f_{\mathcal{O}}, M') = ?$ go back to Step 3 or Step 2 if all pairs y,z have been tried. Otherwise, set $t,x =$ Cornacchia$(f_{\mathcal{O}}, M')$.
5: return $\gamma = x + \beta y + j(z + \beta t)$.

We formulate a statement on the running time of RepresentInteger in Lemma 3.1.4. We remind the reader that $D(M)$ is the set of divisors of M.

Lemma 3.1.4. *(UPHA) Let $M > p$ be an integer and $N = D(M)$. The number of possible outputs to* RepresentInteger$_N$ *is in* $\left(\frac{M}{p\log(M)h(\mathcal{O})}\right)$ *and the running time is in* $O(\text{poly}(\log(M)))$.

Proof. We use the notations of Algorithm 2. Let us assume that the value M has been selected in Step 2. There are (M/p) possible pairs y,z. Let us take a random one. Assuming a good distribution of integers of the form $M - pf_{\mathcal{O}}(y,z)$, the near-primality condition on M' will be satisfied with probability roughly $1/\log(M)$. Then, assuming again a good distribution, under GRH, the probability that there is a solution to the equation $f_{\mathcal{O}}(t,x) = M'$ is $1/h(\mathcal{O})$. This justifies the estimate on the number of solutions and also on the running time by the estimate on the value of d and $M > p$ and the expected running time of Cornacchia. □

We present below a second algorithm StrongApproximation that solves norm equations inside $\mathcal{O}_0$ but with an additional constraint. More precisely, this algorithm finds the strong approximation mod N in N of some $\gamma_0 \in \mathcal{O}_0 \setminus j\mathcal{O}_0$, i.e., an element $\gamma \in \mathcal{O}_0$ with $\gamma = \gamma_0 \in N\mathcal{O}_0$ and $n(\gamma) \in N\mathbb{Z}$ (we give more explanations on the link with the strong approximation in Remark 3.2.2). We will explain later the motivation behind this algorithm.

Remark 3.1.5. StrongApproximation can be modified to output elements of smaller norm following [PS18]. We choose to exclude this improvement from Algorithm 3 for clarity of exposition. The idea from [PS18] is to take a good pair

Algorithm 3 StrongApproximation$_N$

Input: A prime number N, two values $C, D \in \mathbb{Z}$.
Output: $\alpha = \alpha_0 + N\alpha_1$ with $\alpha_0 = j(C + iD)$, $\alpha_1 \in \mathcal{O}_0$ such that $n(\alpha) \in N$.

1: Select $M \in N$ such that $M \equiv pN^4$ and $M/p(C^2 + qD^2)$ is a quadratic residue mod N. Compute the square root β of this element.
2: Select a random pair y, z such that $M \mid pf_{\mathcal{O}}(C + Ny, D + Nz) - \beta^2$ mod N^2. This can be done by solving a linear equation mod N.
3: Set $M' = \frac{M \mid pf_{\mathcal{O}}(C+Ny, D+Nz) - \beta^2}{N^2}$ and determine if the equation $f_{\mathcal{O}}(t, x) = M'$ has a solution (and find its solution if so) using Cornacchia. If no solution exists, go back to Step 2.
4: **return** $\alpha = j(C + Di) + N(t + ix + j(y + iz))$.

y, z instead of sampling it at random. We define good solutions as the ones corresponding to small value of $pf_{\mathcal{O}}(C + Ny, D + Nz)$. In [PS18], it is shown that good solutions correspond to vectors that are close to some lattice. Looking at the determinant of this lattice, we can prove that there exists a solution of approximate size pN^3 (instead of pN^4). This in turns lets us take a smaller $M \in N$ during Step 1. By enumerating short vectors in increasing order, we can make StrongApproximation deterministic. Henceforth, when we write StrongApproximation, we implicitly mean the version of Algorithm 3 modified with the ideas from [PS18].

Lemma 3.1.6. *(UPHA) For any* $\varepsilon > 0$, *there exists* $\kappa = O(\log\log(p) + \log\log(N) + \log(\varepsilon))$ *such that if* c *is bigger than* $\log(p) + 3\log(N) + \kappa$ *and* M *is a random integer with* $c < \log(M) < c + 1$, *then* StrongApproximation$_{D(M)}$ *(modified as in Remark 3.1.5) will succeed with probability higher than* $1 - \varepsilon$. *The expected running time is in* $O(\text{poly}(\log(pN)))$.

Proof. Let us assume that the integer M has been selected in Step 1. As in the proof of Lemma 3.1.4, the probability that, for a given pair y, z, the end of the computation succeeds is in $O(1/h(\mathcal{O})\log(M))$ assuming that M' is well-distributed. Under the heuristics in [PS18], there exists $\kappa = O(\log\log(p) + \log\log(N) + \log(\varepsilon))$ such that if c is bigger than $\log(p) + 3\log(N) + \kappa$ and M is such that $c < \log(M) < c + 1$, then there are enough pairs y, z such that $M > pf(C+Ny, D+Nz)$ to rerandomize M' enough to find one for which Cornacchia will succeed. By our estimates, we will have to try for $O(h(\mathcal{O})\log(M))$ of them, and we can conclude with the complexity of Cornacchia. □

Remark 3.1.7. The formulation of Lemma 3.1.6 targets the case where $N = D(M)$ because it is hard to state a formal result for a generic set N. Lemma 3.1.6 implies that if N is infinite, StrongApproximation will succeed with probability 1. For any $\varepsilon > 0$, when N is finite but contains an element M as in Lemma 3.1.6, the probability of success should be higher than $1 - \varepsilon$ under the assumption that this element M will behave as a random element of the same size would. For most of the algorithms that follows, we will give results with a formulation similar to Lemma 3.1.6.

Strong Approximation step for composite numbers. Here, we explain how to extend the StrongApproximation algorithm to the case where N is not

a single prime but a product of several coprime factors $\prod_{i=1}^{k} N_i$. In fact, it suffices to follow the method described in Algorithm 3. What will change when $N = \prod_{i=1}^{k} N_i$ is simply the running time due to potential failure. There are exactly two possibilities for these failures: first, the equation of Step 2 of Algorithm 3 may require to compute the inverse of non-invertible elements. A random element in $\mathbb{Z}/N\mathbb{Z}$ has a probability at least $c/\log(N)$ to be invertible, so this will cause a logarithmic slowdown in the worst case. In practice, we will consider cases where the N_i are large prime numbers of roughly the same sizes. In these cases, the probability of being invertible will be overwhelming.

The second concern, however, is more problematic, as we can estimate its probability of happening to be roughly in $1 - 2^{-k}$ (we recall that k is the number of coprime factors of N). During Step 1 of Algorithm 3, a value $M \geq N$ must be chosen to ensure that $M/(p(C^2 + D^2))$ is a quadratic residue mod N. The proportion of quadratic residues mod N is $1/2^k$ and this explains the probability estimate we give above. The set N will influence this probability in practice. In [KLPT14], when $N = \ell$ and $k = 1$, the authors explained a method to select a power ℓ^{e_1} to ensure that the quadratic residuosity constraint is always satisfied.

When one of the failures above happens, we will need to rerandomize the computation either by selecting another y, z in Step 2 or, when applying StrongApproximation in a bigger algorithm, by changing the inputs.

3.1.3 Finding solutions in the full order

Most of the time, the fact that RepresentInteger and StrongApproximation produces solutions contained in the suborder $\mathbb{Z}[i, j]$ is perfectly fine. However, we are going to argue later that this can also be problematic depending on the application. In this section, we introduce two new variants FullRepresentInteger and FullStrongApproximation to find solutions in the full order $\mathcal{O}_0$ that were presented in our article [DFLW22]. For simplicity, we assume for this section that we take $p \equiv 7 \bmod 12$ so we have $\mathcal{O}_0$ defined as in Eq. (3.1.1) and we have $\beta = i$ and $q = 1$. We can rewrite the norm form f_0 as

$$f_0 : (t, x, y, z) \mapsto t^2 + x^2 + p(y^2 + z^2) \tag{3.1.3}$$

Let us write g_0, the norm form of the full order $\mathcal{O}_0$. Lemma 3.1.8 below shows how we can relate integers represented by g_0 to the integers represented by f_0.

Lemma 3.1.8. *If M is represented by g_0, then $4M$ is represented by f_0 with (x', y', z', t') such that $t' \equiv z' \bmod 2$ and $x' \equiv y' \bmod 2$.*

Proof. By definition of $\mathcal{O}_0$, we have $g_0 : (t, x, y, z) \mapsto (t + z/2)^2 + (x + y/2)^2 + p((y/2)^2 + (z/2)^2)$. If we have $M = g_0(t, x, y, z)$, we have that $4M = (2t + z)^2 + (2x + y)^2 + p(y^2 + z^2)$ and conversely. Thus, an integer M is represented by f_0 if and only if $4M$ is represented by f_0 and the solution x', y', z', t' satisfies $t' \equiv z' \bmod 2$ and $x' \equiv y' \bmod 2$. □

From Lemma 3.1.8 and RepresentInteger (resp. StrongApproximation), we derive FullRepresentInteger (resp. FullStrongApproximation). In the case of FullRepresentInteger, we include another change in the way the values y', z' are sampled. Instead of selecting both at random in $[-m, m]$ with $m = \lfloor \sqrt{2M/pc} \rfloor$, we start by taking y' at random inside $[-m', m']$ with $m' = \lfloor \sqrt{4M/pc} \rfloor$ before

sampling z' inside $[-m'', m'']$ with $m'' = b\sqrt{4M/p - z'^2}c$. The reason behind this modification will be given in Section 5.6. Intuitively, the idea is to reach a wider range of solutions.

Algorithm 4 FullRepresentInteger$_N$

Input: $N \in \mathbb{Z}$ such that there exists $M \in N$ with $M > p$
Output: $\gamma = x + yi + z\frac{i+j}{2} + t\frac{1+k}{2}$ with $n(\gamma)$ dividing M.

1: Set $M = \{\}$.
2: Select the smallest $M \in N$ not in M such that $M > p$.
3: Set $m' = b\sqrt{\frac{4M}{p}}c$ and sample a random integer $y' \in [-m', m']$.
4: Set $m'' = b\sqrt{\frac{4M}{p} - y'^2}c$ and take a random z' inside $[-m'', m'']$. Set $M' = 4M - pf(y', z')$.
5: If Cornacchia(f_O, M') = ?, go back to Step 3 (or Step 2 if all pairs y', z' have been tried). Otherwise, set t', x' = Cornacchia(f_O, M').
6: If $t' \not\equiv z' \bmod 2$ or $x' \not\equiv y' \bmod 2$, then go back to Step 3 or Step 2 if all pairs y', z' have been tried.
7: Set $\gamma = (x + iy + jz + kt)/2$ and repeat $\gamma = \gamma/2$ while $\gamma/2 \in O_0$ and $n(\gamma) \in 4N$.
8: return γ.

Just as RepresentInteger is heuristically believed to return well-distributed solutions in $Z[i, j]$, the variant FullRepresentInteger is believed to return well-distributed solutions in O_0, thanks to Lemma 3.1.8. A rigorous study of the distribution of the output appears to be hard because of the Cornacchia subroutine, whose success depends on the factorization pattern of its input. This question is further investigated in Section 5.6 in the context of the security of the signature scheme introduced in Chapter 5, with heuristics and experimental evidences.

The running time of FullRepresentInteger is essentially the same as the running time of RepresentInteger, divided by the success probability of the condition $\gamma/2 \in O_0$. Heuristically, this constant is 2/3: the solutions $(x', y', z', t') \bmod 2$ of the equation $x'^2 + y'^2 + p(z'^2 + t'^2) = 0 \bmod 4$ are $(0,0,0,0)$, $(1,1,1,1)$, $(1,0,0,1)$, $(0,1,1,0)$, $(1,0,1,0)$, and $(0,1,0,1)$. Among these 6, there are 2 that do not lead to $\gamma/2 \in O_0$: the solutions $(1,0,1,0)$ and $(0,1,0,1)$. The heuristic analysis is corroborated by the behavior of our implementation.

Remark 3.1.9. One might wonder why we do not propose to swap x' and y' when the constraint is not satisfied. Undeniably, this would be a good way to ensure that each set of values x', y', z', t' leads to a solution. However, this introduces a distinguishable bias, precisely of the kind investigated in Section 5.6.

The StrongApproximation algorithm can also be modified to find solutions in the full order O_0 with Lemma 3.1.8. In Algorithm 5, we present FullStrongApproximation as a generic reduction to StrongApproximation. Thanks to Lemma 3.1.8, properties of the distribution of the output of FullStrongApproximation directly follow from properties of the distribution of StrongApproximation. As in the case of FullRepresentInteger, we expect the running time of FullStrongApproximation to be equal to the running time of StrongApproximation multiplied by 3/2.

Algorithm 5 FullStrongApproximation$_N$

Input: A prime number N, two values $C, D \in \mathbb{Z}$.
Output: $\beta \in O_0$ such that $\beta = \beta_0 + N\beta_1$ with $\beta_0 = jC + kD$, $\beta_1 \in O_0$, and $n(\beta) \in N$.

1: Let $4N = \{4n \mid n \in N\}$.
2: Set β_0 = StrongApproximation$_{4N}(N, C, D)$.
3: If $\beta_0 \notin 2O_0$, go back to Step 2.
4: **return** $\beta = \beta_0/2$.

Lemmas 3.1.4 and 3.1.6 remain true for FullRepresentInteger and FullStrongApproximation. In all the algorithms that we introduce in the next sections, we will use FullRepresentInteger and FullStrongApproximation. Most of the time it will not change much compared to their simpler variant, but it will be crucial in some cases.

3.2 Ideals in the special extremal orders

In this section, we present the main contribution of [KLPT14]: the KLPT$_N$ algorithm. This algorithm finds a solution to the quaternion N isogeny path problem for any O_0-ideal. We show with Lemma 3.2.1 how solving this problem is actually equivalent to solving a norm equation inside I with the map from Definition 1.2.14.

Lemma 3.2.1. *For any integral ideal I, the map*

$$\theta_I(\alpha) = I\frac{\overline{\alpha}}{n(I)}$$

is a surjection from $I \setminus \{0\}$ to the set of ideals J equivalent to I. For $\alpha \in$, we have $\theta_I(\alpha) = \theta_I(\beta)$ if and only if $\alpha = \beta$ where $\beta \in O_R(I)$.

Proof. (sketch) This map is well-defined, as proven in [KLPT14]. We see that it is a surjection by identifying $\overline{I}J$ with a principal ideal $O_R(I)\overline{\alpha}$. Then, it is clear that $\alpha \in I$ and $J = \theta_I(\alpha)$. Finally, one can verify that $O_R(I)\beta_1 = O_R(I)\beta_2$ if and only if $\beta_1 = \beta_2$ where $\beta \in O_R(I)$. □

Following Lemma 3.2.1, we define $q_I : \alpha \mapsto n(\alpha)/n(I)$. As a consequence of Lemma 3.2.1, KLPT$_N$ consists in finding $\alpha \in I$ with $q_I(\alpha) \in N$ before returning $\theta_I(\alpha)$. For what remains of this section, we focus on how to find such an element α.

A common framework. The KLPT algorithm shares a common structure with the other norm equation algorithms that we are going to introduce in Sections 3.3 and 3.4. In the hope that it might provide some insights on these algorithms and help the reader understand how they work and why they were designed in that way, we describe below an informal framework and will try to present our algorithms through that framework.

Let us consider a generic lattice O_0. The goal is to find β of norm contained in N. The framework is parameterized by two integers N_1, N_2, whose exact values will depend on . The algorithms can be decomposed as follows:

1. Find satisfying a set of conditions and having a norm in N_1N.
2. Find $C; D$ 2 Z such that $j(C + D\,!)$ 2 .
3. Compute = StrongApproximation$_{N_1N=n(\ )}(N_2; C; D)$.
4. Output $(j(C + D\,!) + N_2\ )$.

The goal of the \conditions" on the element in the rst step of our framework is to ensure that the second step will always have a solution. The formulation of this second step might seem mysterious at rst glance. One may wonder why we look for an element of this speci c form. The answer to this question is to be found in the StrongApproximation algorithm. The generic principle (explained in the beginning of Section 3.1.2) behind the formulation of RepresentInteger and StrongApproximation is basically the best method we know to nd elements of small given norm in O_0. When we look at StrongApproximation speci cally, it becomes clear that we need to look for $C; D$ such that $j(C + D\,!)$ 2 . The same reasoning is given with more details in [KLPT14]. We always solve the second step in our framework using linear algebra modN_2, as we are going to explain in Section 3.2.2. When N_2 is composite, we will decompose it in sub-operations modulo the di erent factors before using CRT to put everything together. By our assumptions on the form of the set N, we have that $N_1N=n(\ )$ is not empty since $n(\ )$ 2 N_1N.

Remark 3.2.2. The authors of [KLPT14] observed that nding an element could be seen as an e ective realization of the *strong approximation theorem* (this result is basically a generalization of the chinese remainder theorem, see [Voi18, chapter 28] for more details) in the sense that the element generates the lattice almost everywhere (i.e, at all primes coprime to the norm of). Of course, the explanations provided in [KLPT14] are targeted at the case where is an ideal as in Section 3.2, but in fact the strong approximation theorem justi es that we can extend their approach to a larger set of lattices in $B_{p;\ 1}$.

3.2.1 Reducing to the prime-norm case

One thing that we did not mention before, but that will have its importance in practice, is that we will not apply the framework described above on the lattice directly but rather on an \equivalent" lattice. The goal of this initial step is to reduce the values $N_1; N_2$ (whose size will directly impact the size of the nal output, as we can see with Lemma 3.1.6 for instance).

When is an O_0 ideal, this means that we want to replace I with an equivalent ideal L of small prime norm. The fact that the norm is prime is not absolutely necessary, but it makes everything simpler at a minimal cost. To achieve that, we will present two algorithms EquivalentPrimeIdeal and RandomEquivalentPrimeIdeal. The rst one will nd the equivalent ideal of smallest possible prime norm, while the other one will select a random one among a set of potential good candidates, this is useful to rerandomize the computations in case something fails.

With Lemma 3.2.1 we see that we need to nd an element with $q_I(\ )$ equal to N where N is prime and as small as possible. The idea is to use a Minkowski reduced basis $_1; _2; _3; _4$ of I. A Minkowski reduced basis is constituted of

the elements $_i$ attaining the successive minima of q_I and when O is a maximal order, they verify

$$p^2 \quad 16q_I(\ _1)q_I(\ _2)q_I(\ _3)q_I(\ _4) \quad 4p^2: \tag{3.2.1}$$

It can be shown that $q_I(\sum_{i=1}^{4} c_i\ _i)$ is small when the c_i are small. Since we are dealing with lattices of dimension 4, a Minkowski reduced basis can always be e ciently computed.

Note that EquivalentPrimeIdeal takes an integral ideal in input, but there is no assumption on its left order O, it need not be a special extremal order or even a maximal order.

Algorithm 6 EquivalentPrimeIdeal$_b(I)$

Input: I a left O-ideal.
Output: J I of smallest possible prime norm N.
1: Compute a Minkowski-reduced basis $_1; _2; _3; _4$ of I.
2: Use it to nd the element of the smallest prime norm q_I.
3: **return** $J = {}_L(\ )$.

The randomized version RandomEquivalentPrimeIdeal simply computes as a combination $\sum_{i=1}^{4} c_i\ _i$ where the coe cients are random in $[\ b; b]$ for some small bound b.

The authors of [GPS17] showed that EquivalentPrimeIdeal has the same output distribution when given equivalent ideals in input. Thus, we sometimes abuse notations by giving a class C 2 Cl(O) as input to EquivalentPrimeIdeal.

It is important for us to understand the size of the norm N of the outputs of EquivalentPrimeIdeal. When O is a maximal order, we can see that we should have N $p\bar{p}$ with Eq. (3.2.1). This yields Lemma 3.2.3, and this result holds under heuristic assumptions on the distribution of primes represented by some quadratic forms (see [KLPT14] for more details). We stress that this approximation is quite tight in practice when O is a random maximal order, as illustrated in the experimental results of [KLPT14]. However, for some rare speci c cases, the estimate will be o by a big margin, we come back on this case in Section 3.5.

Lemma 3.2.3. *(UPHA) for any* $\ > 0$*, there exists* $\ _0 = O(\log\log(p) + \log(\ ))$ *such that for a random class* C 2 Cl(O_0)*, the norm* N *of* EquivalentPrimeIdeal(C) *veri es* $\log(p)/2 \quad _0 < \log(N) < \log(p)/2 + \ _0$ *with probability higher than* $1 \quad 2 \quad .$

Remark 3.2.4. By taking $\ = \log(p)$, we get that there exists a minimal value $_0$ for which Lemma 3.2.3 holds with overwhelming probability and p uniquely determines this value. Henceforth, we x such a value $_0$. We will use it in Chapter 5.

Lemma 3.2.5. EquivalentPrimeIdeal *and* RandomEquivalentPrimeIdeal *terminates in expected* $O(\text{poly}(\log(pn(I)C)))$ *where* C *is a bound on the coe cients of a basis for* O.

3.2.2 The linear algebra step

We explain here how to perform the second step of the framework we outlined above. For now, let us assume that we replaced I with the equivalent L of norm N and that we have a suitable element γ. In Section 3.2.3, we will give a necessary condition on γ with Lemma 3.2.6 and explain how to find this element γ. Our goal is to find $C, D \in \mathbb{Z}$ such that $\gamma(C + \iota D) \in L$. Since $NO \subset L$, it is natural for us to work mod N to find these coefficients L. Since L is a $\mathbb{Z}$-module, we can scale by an invertible scalar without changing anything. In the end, we see that we actually look for a solution $(C : D) \in \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$.

To find a representative $(C : D)$ with the correct property we use more concretely the fact that $O_0/NO_0 \simeq \mathrm{M}_2(\mathbb{Z}/N\mathbb{Z})$ (which can be seen for instance from the fact that $O_0 \otimes \mathbb{Z}_N \simeq \mathrm{M}_2(\mathbb{Z}_N)$) and so the solution we look for will be in the kernel of the matrix generated by $\gamma, \gamma\iota$ and a basis of $L \bmod NO_0$.

We note IdealModConstraint the algorithm taking in input L, γ and outputting the correct $(C : D) \in \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$. We do not give a detailed description of this algorithm, as it is fairly simple to derive from our explanations above.

3.2.3 Putting everything together

We now have all the building blocks for the full KLPT algorithm from [KLPT14]. With Lemma 3.2.6, we understand what "conditions" we require on the element γ.

Lemma 3.2.6. *[KLPT14] Let L be a random O_0 ideal of norm N and $\gamma \in O_0$. When $\gcd(n(\gamma), N^2) = N$, there exist $C, D \in \mathbb{Z}$ such that $\gamma(C + D\iota) \in L$ with probability at least $1 - 4(N - 1)/(N + 1)$.*

Proof. (sketch) The proof is quite similar to what we have seen in Proposition 2.3.12. The map $(C : D) \mapsto O_0 \to \gamma(C + \iota D)/O_0N$ gives an action of $(\gamma O/N\gamma O)^\times$ up to scalar multiplication (which is in bijection with $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$) on the ideals of the form J/NO_0 where J is a cyclic ideal of norm N in O_0. This action has either one orbit and so this proves the result, or it has two orbits: one of size 2, one of size $N - 1$. The former happens when N is inert in O and the latter if N splits. The probability that $O_0 = NO_0$ and $L : N_0$ are in the same orbit is $1 - 4(N - 1)/(N + 1)$. □

In practice, we can expect values of N to be polynomial in p. In that case, we can consider the probability given in Lemma 3.2.6 to be overwhelming. In full generality, we can add the constraint that N is inert in O to increase the probability of success to 1. The condition on γ from Lemma 3.2.6 is quite easy to satisfy with the RepresentInteger algorithm.

We are now ready to give the detailed description of the KLPT algorithm from [KLPT14]. Following Section 3.2.1, the first step is to replace I by an equivalent ideal L of prime norm N. Then, we apply our framework with $N_1 = N_2 = N$ and Lemma 3.2.6 with the content of Section 3.2.2 to get Algorithm 7.

Proposition 3.2.7. *(UPHA) For any $\varepsilon > 0$, there exist $\eta_1, \eta_2 = O(\log\log(p) + \log(\varepsilon))$ such that if $c_1 > 1/2\log(p) + \eta_1$ and $c_2 > 5/2\log(p) + \eta_2$ and M_i is a random integer with $c_i < \log(M_i) < c_i + 1$ for $i = 1, 2$ then $\mathrm{KLPT}_{D(M_1M_2)}$ will succeed with probability higher than $1 - \varepsilon$ on an ideal whose class is uniformly distributed in $\mathrm{Cl}(O_0)$. The expected running time is in $O(\mathrm{poly}(\log(pn(I))))$.*

Algorithm 7 $\mathrm{KLPT}_N(I)$

Input: I a left O_0-ideal.
Output: $J \sim I$ of norm in N.

1: Compute L = EquivalentPrimeIdeal(I) with $N = n(L)$ and $\in I$ such that $L = {}_I(\ )$.
2: Compute $\ $ = FullRepresentInteger$_{N\ N}$.
3: Compute $(C_0 : D_0)$ = IdealModConstraint($L;\ $).
4: Compute $\ $ = FullStrongApproximation$_{N\ N=n(\ )}(N; C_0; D_0)$) and set $\ $ = $\ $. If the computation fails, go back to Step 2.
5: **return** $J = {}_L(\ )$.

Proof. By Lemma 3.2.3, the integer N will be contained in $\log(p)/2 \ {}_0 < \log(N) < \log(p)/2 + {}_0$ for some ${}_0 = O(\log\log(p))$. If we take ${}_1 = {}_0$, $\log(M) > c_1 > 1/2\log(p) + {}_0$ implies that $M_1 N > p$. Thus, by Lemma 3.1.4, we can find a solution $\ $ with FullRepresentInteger. By Lemma 3.2.6, an element $(C_0 : D_0) \in \mathrm{P}^1(\mathbb{Z}/N\mathbb{Z})$ can be found with IdealModConstraint. Then, by Lemma 3.1.6, we can take ${}_2$ to be the $\ $ from Lemma 3.1.6, so that FullStrongApproximation will succeed with probability bigger $1 \ \ 2 \ \ $.

By Lemma 2.2.5, I has a basis whose coefficients have size $O(\log(n(I)))$. Hence, the expected running time follows from Proposition 2.2.4 and Lemmas 3.1.4, 3.1.6 and 3.2.5. □

Remark 3.2.8. A result of [GPS17] proves that the outputs of EquivalentPrimeIdeal and KLPT only depend on the equivalence class of the input (in fact, this is only true with a minor tweak to the original algorithm of [KLPT14]). Hence, we will sometimes abuse notations and use both algorithms as if they took inputs in $\mathrm{Cl}(O_0)$. Moreover, we will assume henceforth that KLPT is deterministic (which can always be achieved by fixing an order on the random choices).

As explained in Remark 3.1.7, Proposition 3.2.7 can be used to deduce meaningful results on KLPT_N for more complicated sets N.

Extending KLPT to any maximal order. The KLPT algorithm presented as Algorithm 7 specifically targets O_0-ideals. In fact, we gave a formulation that works efficiently for all $\ $-special extremal orders when $\ $ is not too big. However, this represents only a minority of maximal orders, as most of them do not contain a quadratic order of small discriminant. For a generic maximal order, we have the upper-bound $p^{2=3}$ for the norm of the smallest non-trivial endomorphism $\ !$ and this bound is tight. This might seem like a big obstacle but the authors of [KLPT14] observed that an efficient algorithm in a special case was sufficient as all the maximal orders are in the same genus. Let us take the ideal $I = I(O_1; O_2)$, where neither $O_1; O_2$ are special extremal. To find $J \sim I$, it suffices to find $J_i \sim I_i = I(O_0; O_i)$ for $i = 1; 2$ and take $J = J_1\overline{J_2}$. The I_i exists because all maximal orders are connected, and the J_i can be computed with KLPT. This idea provides a way to solve the quaternion N-isogeny path problem for any maximal order with at most two executions of the KLPT_N algorithm. While this method is satisfying in itself, we are going to see later that it is not generic enough for one of our cryptographic application. This

is partly what motivated the study of Eichler orders and their ideals that we introduce in Section 3.3.

Remark 3.2.9. In his recent article [Wes22], Benjamin Wesolowski introduced a version of KLPT for which he managed to prove termination in polynomial time assuming only GRH. This version is a lot less e cient than the one we presented, we do not explain it in full details here. Nonetheless, it is important to know that (up to increasing a lot the size of the outputs) we have a provable version of KLPT. Note that there are no provable versions of all the other algorithms we introduce in the remaining of this chapter. However, given that they are built mostly on the same building blocks as KLPT, it is quite probable that provable versions could be derived similarly. This is not something that we have investigated in full details, though.

3.3 Eichler orders and their ideals

In this section, we will show another way to solve Problem 3.0.1 for the O_1-ideal I, where O_1 is a generic maximal order, using algorithms to solve norm equations in Eichler orders and their ideals. Our idea is in fact quite simple. Since we know how to solve norm equations in O_0 and O_0-ideals, we will consider the order $O_1 \cap O_0$ and the ideal $I \cap O_0$. By de nition, we get an Eichler order and the ideal of an Eichler order (see Lemma 2.3.9) in this manner. Thus, we will derive a GenericKLPT algorithm from an algorithm IdealEichlerNorm to solve norm equations in ideals of Eichler orders.

Adaptation of the algorithmic framework to the new case. Now that we have given an example with KLPT of an algorithm following our framework, the formulation of all the other algorithms will appear more natural. To get our desired algorithm for Eichler orders and their ideals, we just need to understand the conditions on . With Proposition 3.3.1, we cover the case of Eichler orders.

Proposition 3.3.1. *Let $\mathbb{Z} + I$ be an Eichler order of level N contained in O_0 and N inert in O. If $_1; _2 \in O_0$ are such that* $\gcd(n(_i); N) = 1$, *then there exist $C; D \in \mathbb{Z}$ with $_1 j(C + !D)\ _2 \in (\mathbb{Z} + I)$.*

Proof. We can get the result by looking at the map from Proposition 2.3.12 with $L = O_0$ and $= _2$. When we apply Proposition 2.3.12 to the class C_0 of principal ideals of O_0, we see that each of the classes of $\mathrm{Cl}_{\mathbb{Z}+I}(C_0)$ contains an ideal of the form $O_0\overline{_2}j(C + D!)$ for some $C; D \in \mathbb{Z}$. In particular, there exist $C_0; D_0$ such that $O_0\ _1\ _{\mathbb{Z}+I}\ O_0\overline{_2}j(C_0 + !D_0)$. In that case, the results from Section 2.3.1 prove that we must have $\overline{_1 j(C_0 + !D_0)}\ _2 \in \mathbb{Z} + I$, and this proves the result. □

Proposition 3.3.1 implies that we can design an algorithm EichlerModConstraint very similar to IdealModConstraint that takes $I; _1; _2$ in input and outputs $(C : D) \in \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$ such that $_1 j(C + !D)\ _2 \in \mathbb{Z} + I$. This algorithm consists mainly of linear algebra mod N. Henceforth, we assume that we have such an algorithm EichlerModConstraint.

We deduce from Proposition 3.3.1 a useful corollary on EichlerModConstraint, for two values $_2; \frac{0}{2}$ contained in the same Eichler class, which will be useful for us at some point in Chapter 5.

Corollary 3.3.2. *Let I be an O_0-ideal of norm N and $\gamma_1, \gamma_2, \gamma_2' \in O_0$ all with norm coprime to N. Let L be an ideal of norm coprime to L such that $\gamma_2, \gamma_2' \in L$. If $\gamma_L(\gamma_2) \in \mathbb{Z}+I \Leftrightarrow \gamma_L(\gamma_2')$, then* EichlerModConstraint$(I, \gamma_1, \gamma_2) =$ EichlerModConstraint(I, γ_1, γ_2').

Proof. When $\gamma_L(\gamma_2) \in \mathbb{Z}+I \Leftrightarrow \gamma_L(\gamma_2')$, the results from Section 2.3.1 prove that the two maps γ, γ' related to γ_2, γ_2' in Proposition 2.3.12 are the same, and so the proof of Proposition 3.3.1 allow us to conclude. □

Norm equations in $O_1 \cap O_0$. In this paragraph, we assume for simplicity that the ideal $I = I(O_0, O_1)$ has prime norm N inert in O. Unlike the O_0-ideal case and what is explained in Section 3.2.1, it seems hard to replace I with another equivalent ideal in our case. We will explain in Section 3.5 that most of the algorithms involved in EichlerNorm can still be made to work when N is not prime. The main obstacle is in fact FullStrongApproximation and we discussed in Section 3.1.2 what are the effects of a composite modulus. If N is not inert in O, the algorithm will still work with overwhelming probability $1 - 4(N-1)/(N+1)$ as in Lemma 3.2.6.

Under our assumption on N, we have that I satisfies the constraint of Proposition 3.3.1. We have $\mathbb{Z} + I = O_0 \cap O_1$ and when we apply Proposition 3.3.1 to our framework with $N_1 = 1$ and $N_2 = N$, we get algorithm EichlerNorm.

Algorithm 8 EichlerNorm$_N(I)$

Input: I a cyclic O_0-ideal of norm N inert in O.
Output: $\gamma \in \mathbb{Z} + I$ of norm in N.

1: Compute $(C : D)$ = EichlerModConstraint$(I, 1, 1)$.
2: Compute γ = FullStrongApproximation$_N(N, C, D)$.
3: return γ.

Proposition 3.3.3. *(UPHA) For any $\varepsilon > 0$, there exists $\alpha = O(\log\log(p) + \log\log(N) + \log(\varepsilon))$ such that if c is bigger than $\log(p) + 3\log(N) + \alpha$ and M is a random integer with $c < \log(M) < c + 1$, then* EichlerNorm$_{D(M)}$ *will succeed with probability higher than $1 - 2\varepsilon$. The expected running time is in $O(\text{poly}(\log(pN)))$.*

Proof. Our result follows from Lemma 3.1.6 and Proposition 3.3.1. □

Solving norm equations in a generic maximal order. With EichlerNorm, we have an algorithm to solve norm equations in Eichler orders of the form $\mathbb{Z} + I$ where I is an O_0-ideal of norm N. We can deduce an algorithm SpecialEichlerNorm to solve norm equations efficiently in any maximal order O. The idea is to find $O_1 = O$ such that $I(O_0, O_1)$ satisfies the constraints on the input of EichlerNorm, so we can solve in $O_0 \cap O_1$ and then transport the output in O using the isomorphism between O and O_1. Motivated by our application of SpecialEichlerNorm to more complicated algorithms in Chapter 4, we require that the output of SpecialEichlerNorm$_N$ satisfies the following additional constraint: given the additional input K, a left O-ideal of norm ℓ coprime to all the elements of N, we need that $\gamma \in \mathbb{Z} + K$ (see Step 2 in Algorithm 8). A justification for this constraint will be provided in Section 4.2.2.

Algorithm 9 SpecialEichlerNorm$_N(O; K)$

Input: O a maximal order and K a left O-ideal of norm ℓ.
Output: $\gamma \in O$ r $(\mathbb{Z} + K)$ of norm in N coprime to ℓ.

1: Compute $I = I(O_0; O)$.
2: Set L = RandomEquivalentPrimeIdeal(I), $N = n(L)$ and compute γ s.t. $L = I\ \ .$
3: Compute $K' = \ \ ^{1}K$
4: Compute $(C : D)$ = EichlerModConstraint$(L; 1; 1)$.
5: Enumerate all possible solutions of γ = FullStrongApproximation$_N(N; C; D)$ until $\gamma \notin \mathbb{Z} + K'$. If it fails, go back to Step 2.
6: **return** $\gamma = \ \ ^{1}$.

Proposition 3.3.4. *(UPHA) Let ℓ be a small prime.* SpecialEichlerNorm *is correct. For any $\ > 0$, there exists $\ = O(\log\log(p) + \log(\ ) + \log(\ell))$ such that if $c > 5/2\log(p) + \ $ and M is a random integer coprime to ℓ and $c < \log(M) < c + 1$, then* SpecialEichlerNorm$_{D(M)}$ *will succeed with probability higher than $1 \ \ 2\ $ on a uniformly random maximal order O and random cyclic O-ideal of norm ℓ. The expected running time is in $O(\text{poly}(\log(p)))$.*

Proof. Under the estimate of Lemma 3.2.3, we can expect N to be around $\sqrt{p}$. Thus, the size estimate follows from the same reasons as in Proposition 3.3.3. For the running time and probability of success, we need to look at the constraint $\gamma \notin \mathbb{Z} + K'$. Now, we introduce the following heuristic assumption: the output of FullStrongApproximation satisfies $\gamma \notin \mathbb{Z} + K'$ with probability approximately $[O_0 : \mathbb{Z} + K']^{\ 1} = 1/\ell$ (which is the probability one would get by drawing uniformly in a large enough ball of O_0). Even though the precise distribution of γ appears difficult to analyze, this heuristic is plausible since the algorithm FullStrongApproximation seems to constrain possible values of γ only locally at N and M, both coprime to ℓ. Thus, when we combine the probability of success of FullStrongApproximation from Lemma 3.1.6, we get the result.

Correctness follows from the fact that $\gamma \in \mathbb{Z} + L = O_0 \cap O_R(L)$, where $L = I\ \ $, we can show that $O = \ \ O_R(L)\ \ ^{1}$. Since $\gamma \notin \mathbb{Z} + K'$, then $\ \ ^{1}\gamma \notin \mathbb{Z} + \ \ K\ \ ^{1} = \mathbb{Z} + K$. □

Remark 3.3.5. Note that the new heuristic introduced in the proof of Proposition 3.3.3 would not have held if we had used StrongApproximation instead of FullStrongApproximation. Indeed, as we have explained, the solutions of StrongApproximation lie in $\mathbb{Z}\langle 1; i; j; k\rangle$ which is contained in the Eichler order $(\mathbb{Z} + O_0\langle 1 + i; 2i\rangle)$. Thus, in the execution of SpecialEichlerNorm, when $K \cap L = O_0\langle 1 + i; 2i\rangle \cap L$, the condition $\gamma \notin \mathbb{Z} + K$ can never be satisfied. This is why it is important to use our new variant FullStrongApproximation to ensure that the elements are not contained in any specific suborder of O_0.

Norm equations in ideals of Eichler orders. In this paragraph, we show how to solve norm equations in lattices of the form $(\mathbb{Z} + I) \cap J$ where $I; J$ are O_0 cyclic ideals of coprime norm with the algorithm IdealEichlerNorm. The fact that the norms are coprime is in fact essential, as will become clear in the proof of Proposition 3.3.6. Intuitively, this is because we need to deal with the two constraints (the solution needs to be in $\mathbb{Z} + I$ and in J) independently. When

the norms are coprime, the local-global principle tells us that this is possible. As for EichlerNorm, we assume that I has norm N_I, a prime number inert in O. We need not make the same assumption for J as we can use EquivalentPrimeIdeal to replace J by an equivalent ideal L of prime norm N (we also require that $\gcd(N; N_I) = 1$). With our framework, the combination of Proposition 3.3.1 and Lemma 3.2.6 into Proposition 3.3.6 will underlie the formulation of IdealEichlerNorm. We will see that we get IdealEichlerNorm by combining KLPT with EichlerNorm. Intuitively, it is quite natural under the bijection of Lemma 2.3.9 that identifies the $\mathbb{Z} + I$-ideals of norm N with the O_0-ideals of norm N.

Proposition 3.3.6. *Let $I; J$ be two O_0 ideals of norm $N_I; N_J$ with $\gcd(N_I; N_J) = 1$. Let $L = {}_J(\ )$ and $N := n(L)$ coprime to N_I, and let $\ 2\ O_0$ be such that $\gcd(n(\ ); N_I N^2) = N$. There exist $C; D\ 2\ \mathbb{Z}$ with $\ j(C + !D)\ \ 2\ (\mathbb{Z} + I) \setminus J$.*

Proof. By Lemma 3.2.6, there exist $C_0; D_0\ 2\ \mathbb{Z}$ such that $\ j(C_0 + !D_0)\ 2\ L$ and it is easy to see that any $C; D = C_0; D_0 \mod N$ work as well since $NO_0 \quad L$. By Proposition 3.3.1, there exist $C_1; D_1\ 2\ \mathbb{Z}$ with $\ j(C_1 + !D_1)\ \ 2\ \mathbb{Z} + I$ because $n(\ ) = NN_J$ which is coprime to N_I. Similarly, any $C; D = C_1; D_1 \mod N_1$ will preserve this property. So if we take elements $C; D\ 2\ \mathbb{Z}$ satisfying the two constraints $\mod N; N_1$, we get that $\ j(C + !D)\ 2\ L$ and so $\ j(C + !D)\ \ 2\ J$ by definition of the map. We also have $\ j(C + !D)\ \ 2\ \mathbb{Z} + I$, and this proves the result. □

Concretely, the elements $C; D$ in Proposition 3.3.6 will be found using IdealModConstraint, EichlerModConstraint and CRT to combine the results $\mod N$ and $\mod N_I$. In our framework, we take as in Proposition 3.3.6 and $N_1 = N$ and $N_2 = NN_I$.

Algorithm 10 $\text{IdealEichlerNorm}_N(I; J)$

Input: $I; J$ be two O_0 ideals of norm $N_I; N_J$ with $\gcd(N_I; N_J) = 1$.
Output: $\ 2\ (\mathbb{Z} + I) \setminus J$ and $n(\ )\ 2\ n(J)N$.

1: Compute $L = \text{EquivalentPrimeIdeal}(J)$, $L = {}_J(\ )$ for $\ 2\ J$. Set $N = n(L)$.
2: Compute $\ = \text{FullRepresentInteger}_{NN}$.
3: Compute $(C_0 : D_0) = \text{IdealModConstraint}(L; \ )$.
4: Compute $(C_1 : D_1) = \text{EichlerModConstraint}(I; \ ; \ )$.
5: Compute $C = \text{CRT}_{N;N_I}(C_0; C_1)$ and $D = \text{CRT}_{N;N_I}(D_0; D_1)$.
6: Compute $\ = \text{StrongApproximation}_{NN=n(\ )}(NN_I; C; D)$.
7: return .

Proposition 3.3.7. *(UPHA) For any $\ > 0$, there exist ${}_1; {}_2 = O(\log\log(p) + \log(\ ))$ such that if $c_1 > 1{=}2\log(p) + {}_1$ and $c_2 > 5{=}2\log(p) + 3\log(N_I) + {}_2$ and M_i is a random integer with $c_i < \log(M_i) < c_i + 1$ for $i = 1; 2$, then $\text{EichlerNorm}_{D(M_1M_2)}$ will succeed with probability higher than $1 \quad 2$ when J is in a random class of $\text{Cl}(O_0)$. The expected running time is in $O(\text{poly}(\log(pN_IN_J)))$.*

Proof. We obtain the result by combining Proposition 3.3.6 with Lemmas 3.1.4 and 3.1.6 and the estimates from Lemma 3.2.3 for the size of N. □

A Generic KLPT algorithm. We arrive at the main goal of this section: a new generic version of KLPT for all maximal orders. Our algorithm Generic-KLPT$_N$ takes in input a maximal order O, an O-ideal I, and nds J I with $n(J)$ 2 N. The idea is to combine IdealEichlerNorm with SpecialEichlerNorm to nd 2 J. More precisely, we will nd K of prime norm connecting O_0 and $O_1 = O$. Then, we will apply IdealEichlerNorm to nd 2 $(\mathbb{Z} + K) \setminus [K]$ I of norm in N. By the results of Section 2.3.1, we get that 2 I, and so the output is simply $_I(\ )$. The reasoning that we just outlined is not exactly formal because I is not an O_1-ideal, but it works if we use the isomorphic copy of I in O_1. The relevant isomorphism is $7!$ ^{1} where is s.t. $K = I(O_0; O_1)$.

Algorithm 11 GenericKLPT$_N(O_1; I)$

Input: O_1 a maximal order and I a left O_1-ideal.
Output: J I of norm in N.

1: Compute $I_0 = I(O_0; O_1)$.
2: Set K = RandomEquivalentPrimeIdeal(I_0), compute s.t. $K = I_0$.
3: Compute $I' =$ ^{1}I and $L = [K]$ I'.
4: Compute = IdealEichlerNorm$_N(K; L)$.
5: Compute = ^{1}.
6: **return** $_I(\ )$.

Proposition 3.3.8. *(UPHA) For any* > 0*, there exist* $_1$; $_2 = O(\log\log(p) + \log(\ ))$ *such that if* $c_1 > 1/2\log(p) +$ $_1$ *and* $c_2 > 4\log(p) +$ $_2$ *and* M_i *is a random integer with* $c_i < \log(M_i) < c_i + 1$ *for* $i = 1, 2$*, then* EichlerNorm$_{D(M_1M_2)}$ *will succeed with probability higher than* 1 2 *when* O_1 *is a random maximal order and* I *is in a random class of* Cl(O)*. The expected running time is in* $O(\text{poly}(\log(pn(I)C)))$ *where* C *is a bound on the coe cients of a basis for* O_1*.*

Proof. The size estimates and expected running time are a consequence of Lemmas 3.2.3 and 3.2.5 and Proposition 3.3.7. For correctness, since 2 $(\mathbb{Z}+K) \setminus L$, 2 $\mathbb{Z} + I_0 = O_0 \setminus O_1$ and 2 $[K]$ $L = I$ and so the output is correct. □

As for KLPT (see Remark 3.2.8), we assume that GenericKLPT is deterministic. In Chapter 5, we will introduce SigningKLPT, a variant of GenericKLPT adapted to the signature scheme SQISign. In particular, the distribution of output is crucial for the security of the protocol, and we will explain there why the generic method from Kohel et al. in [KLPT14] is not su cient for this application.

Remark 3.3.9. Proposition 3.3.8 shows that our new method succeeds in nding an ideal of norm smaller than the solution proposed in [KLPT14] that we outlined in the end of Section 3.2.3. Indeed, as mentioned above, their output is a concatenation of two solutions obtained from KLPT, thus their output is of norm roughly equal to p^6. Finding a smaller solution was not the primary motivation, but it is nice nonetheless.

Norm equation in all Eichler orders and ideals. We advertised an algorithm to solve norm equations in Eichler orders, but so far we have only treated the case of orders of the form $O_0 \setminus O_1$. Let us take a generic Eichler order $O = O_1 \setminus O_2 = \mathbb{Z} + I(O_1; O_2)$. It can be shown that $O \setminus O_0 = (\mathbb{Z} +$

$I(O_0;O_1)I(O_1;O_2)$). The algorithm EichlerNorm can be modified to work on an input ideal whose norm is not a prime inert in O. The efficiency will be reduced (depending mainly on the number of factors of the ideal in input) and there may be some failures with constant probability but since we can rerandomize by replacing $I(O_0;O_1)$ by equivalent ideals as in Step 2 of SpecialEichlerNorm or GenericKLPT, we can increase the probability of success to be overwhelming (if there are big enough elements in N). Since we have no concrete applications of this algorithm, we do not describe it in full details.

3.4 Norm Equations in non-Gorenstein suborders of Eichler orders

In this section, we treat the case of non-Gorenstein orders whose Gorenstein closure is an Eichler order. Concretely, this means lattices of the form $\mathbb{Z} + DI$ and $(\mathbb{Z} + DI) \setminus J$ where $I;J$ are cyclic integral ideals of maximal orders and $\gcd(n(I);n(J);D) = 1$. Our motivation is the resolution of norm equations inside $\mathbb{Z} + DO$ for any maximal order $O \quad B_{p;1}$. As in Section 3.3, we will restrict the resolution to the suborder $(\mathbb{Z} + DO) \setminus O_0$. Since $O \setminus O_0 = \mathbb{Z} + I$ where $I = I(O_0;O)$, we end up solving inside $\mathbb{Z} + DI = (\mathbb{Z} + DO) \setminus (\mathbb{Z} + I)$. We introduce SuborderEichlerNorm to do that. We apply SuborderEichlerNorm to get GeneratingFamily, an algorithm that computes a generating family of $\mathbb{Z} + DO$ (see Definition 3.4.3). Finally, we show that SuborderEichlerNorm can be extended to treat ideals of the form $(\mathbb{Z} + DI) \setminus J$, and we introduce IdealSuborderEichlerNorm. Both these algorithms will prove useful for the content of Chapter 4 where we will use these algorithms in our new isogeny representation.

Norm equations inside $\mathbb{Z} + DI$. Next, we explain our method for the case $= \mathbb{Z} + DI$. As before, we assume that I has a prime norm N inert in O. This time, we need to satisfy more conditions than a simple constraint on its norm. We will introduce the necessary condition in Proposition 3.4.1. It proves to be slightly inconvenient, and will impact the size of the final solution, but we managed to find a way to keep some control on the norm of while ensuring that the linear algebra step always has a solution.

Proposition 3.4.1. *Let I be an integral left O_0-ideal of prime norm N and let D be a distinct prime number. If $\in O_0$ can be written as $j(C_2 + !D_2) + D_2$ with $_2 \in O_0$ and $n(\ )$ is coprime to N, then there exist $C_1;D_1 \in \mathbb{Z}$ such that $j(C_1 + !D_1) \in \mathbb{Z} + DI$.*

Proof. If $n(\ )$ is coprime to N, we know from Proposition 3.3.1 that there exist $C_0;D_0$ such that $j(C_0 + !D_0) \in \mathbb{Z} + I$. Then, if we set $C_2' = D_2'C_2(D_2)^{-1} \bmod D$ for any D_2', it is easy to verify that $j(C_2' + !D_2') \in \mathbb{Z} + DO_0$. Hence, if $C_1;D_1$ satisfy $C_1;D_1 = C_0;D_0 \bmod N$, $C_1;D_1 = C_2';D_2' \bmod D$ and $\gcd(N;D) = 1$, we have that $j(C_1 + !D_1) \in \mathbb{Z} + DO_0 \setminus (\mathbb{Z} + I) = \mathbb{Z} + DI$. □

With Proposition 3.4.1, we see that we must take $N_1 = 1$ and $N_2 = ND$ and that we must also apply a strong approximation mod D to compute a whose norm is in N. When we apply these ideas to the framework described above, we obtain SuborderEichlerNorm.

Algorithm 12 SuborderEichlerNorm$_N(D; I)$

Input: I a left O_0-ideal of norm prime N coprime to D.
Output: $\alpha \in \mathbb{Z} + DI$ of norm contained in N.

1: Select a random class $(C_2 : D_2) \in \mathbb{P}^1(\mathbb{Z}/D\mathbb{Z})$.
2: Compute γ_2 = FullStrongApproximation$_N(D; C_2; D_2)$). If the computation fails, go back to Step 1.
3: Compute $(C_0 : D_0)$ = EichlerModConstraint$(I, \gamma_2, 1)$.
4: Sample a random D_2' in $\mathbb{Z}/D\mathbb{Z}$, compute $C_2' = D_2' C_2 (D_2)^{-1} \bmod D$.
5: Compute $C_1 = \mathrm{CRT}_{N;D}(C_0; C_2')$, $D_1 = \mathrm{CRT}_{N;D}(D_0; D_2')$.
6: Compute γ_1 = FullStrongApproximation$_{N=n(\gamma_1)}(ND; C_1; D_1)$). If it fails, go back to step 1.
7: **return** $\alpha = \gamma_1\gamma_2$.

Proposition 3.4.2. *For any $\varepsilon > 0$, there exists $\eta_1, \eta_2 = O(\log\log(pND) + \log(1/\varepsilon))$ such that if $c_1 > \log(p) + 3\log(D) + \eta_1$ and $c_2 > \log(p) + 3\log(DN) + \eta_2$ and M_i is a random integer with $c_i < \log(M_i) < c_i + 1$ for $i = 1, 2$, then* SuborderEichlerNorm$_{D(M_1M_2)}$ *will succeed with probability higher than $1 - \varepsilon$ when I is in a random class of $\mathrm{Cl}(O_0)$. The expected running time is in $O(\mathrm{poly}(\log(pDN)))$.*

Proof. The result follows from Proposition 3.4.1 and Lemma 3.1.6. We can verify that $j(C_2 + D_2\ell)j(C_2' + \ell D_2') \in \mathbb{Z} + DO_0$. Since $j(C_2 + D_2\ell)j(C_2' + \ell D_2') \in DO_0$ this proves that $\alpha \in \mathbb{Z} + DO_0$. □

A generating family of $\mathbb{Z}+DO$. In this paragraph, we show how to compute a *generating family* of the order $\mathbb{Z} + DO_1$ where O_1 is a maximal order of $B_{p,\infty}$. This algorithm will be useful in Chapter 4.

Definition 3.4.3. A *generating family* $\alpha_1, \ldots, \alpha_n$ for an order O is a set of elements in O such that any element $\alpha \in O$ can be written as a linear combination of 1 and $\prod_{j \in I} \alpha_j$ for all $I \subset \{1, \ldots, n\}$. In that case, we write $O = \mathrm{Order}(\alpha_1, \ldots, \alpha_n)$.

We present below GeneratingFamily$_N$ that takes in input a maximal order O_1 and a prime D, outputs a generating family of $\mathbb{Z} + DO_1$ with elements having norm in N. The idea behind this algorithm is quite straightforward: apply SuborderEichlerNorm on I, for various ideals I connecting O_0 and orders isomorphic to O_1. This gives a way to sample elements in $\mathbb{Z} + DO_1$, and it suffices to iterate this method until we obtain a generating family from this set. Experimental results show that after taking a few elements in that manner (for instance, no more than ten for parameters of cryptographic sizes, i.e, of a few hundred bits), we can extract a generating family of size three. We formulate this more precisely as Conjecture 1.

Conjecture 1. Let O_1 be a maximal order in $B_{p,\infty}$. Let I_1, I_2, I_3 be random O_0-ideals of prime norms with $\gamma_i O_R(I_i) \gamma_i^{-1} = O$ for some $\gamma_i \in B_{p,\infty}$. If $\alpha_1, \alpha_2, \alpha_3$ are random outputs of SuborderEichlerNorm$(D; I_i)$ for $i = 1, 2, 3$, then $\mathbb{Z} + DO_1 = \mathrm{Order}(\alpha_1\gamma_1\alpha_1\gamma_1^{-1}, \alpha_2\gamma_2\alpha_2\gamma_2^{-1}, \alpha_3\gamma_3\alpha_3\gamma_3^{-1})$ with probability $1/c$ where $c = O(\mathrm{poly}(\log(pD)))$.

Algorithm 13 GeneratingFamily$_N$ $(O_1; D)$

Input: A maximal order O_1 and a prime D.
Output: A generating family $_1; \ldots; _3$ for $\mathbb{Z} + DO_1$ where each $_j$ has norm in N.

1: Set $L = \{\}$ and $I_0 =$ ConnectingIdeal$(O_0; O_1)$.
2: **while** There do not exist $_1; _2; _3 \in L$ s.t $\mathbb{Z} + DO_1 =$ Order$(_1; _2; _3)$ **do**
3: $I =$ RandomEquivalentPrimeIdeal(I_0) and $I = I_0$.
4: Compute $=$ SuborderEichlerNorm$_N(D; I)$.
5: $L = L \cup \{^{-1}\}$.
6: **end while**
7: **return** $_1; _2; _3$.

Proposition 3.4.4. *(UPHA) Assuming Conjecture 1, for any > 0, there exist $_1; _2 = O(\log\log(pD) + \log(\ ))$ such that if $c_1 > \log(p) + 3\log(D) + _1$ and $c_2 > 5/2\log(p) + 3\log(D) + _2$ and M_i is a random integer with $c_i < \log(M_i) < c_i + 1$ for $i = 1, 2$, then* GeneratingFamily$_{D(M_1M_2)}$ *will succeed with probability higher than $1 - 2^{-}$ when O_1 is a random maximal order in $B_{p;\infty}$. The expected running time is in $O(\text{poly}(\log(pDC)))$ where all the coefficients of a basis for O_1 are smaller than C.*

Proof. The result follows from Lemmas 3.2.3 and 3.2.5, Propositions 2.2.4 and 3.4.2 and Conjecture 1.

□

Norm equations inside $(\mathbb{Z}+DI)\setminus J$. For simplicity in this paragraph, we assume that both I and J have inert prime norm $N_I; N_J$ and $\gcd(N_I; N_J; D) = 1$. We obtain Proposition 3.4.5 as a combination of Lemma 3.2.6 and Proposition 3.4.1. IdealSuborderEichlerNorm follows from our framework if we take $N_1 = N_J$, $N_2 = N_I N_J D$. This yields Algorithm 14.

Proposition 3.4.5. *Let $I; J$ be two cyclic integral left O_0-ideals of prime norm $N_I; N_J$ inert in J, and let D be a distinct prime number. If $\in O_0$ can be written as $j(C_2 + !D_2) + D_2$ with $_2 \in O_0$ and $\gcd(n(\ ); N_I N_J^2) = N_J$, then there exist $C_1; D_1 \in \mathbb{Z}$ such that $j(C_1 + !D_1) \in (\mathbb{Z} + DI)\setminus J$.*

Proposition 3.4.6. *(UPHA) For any > 0, there exist $_1; _2 = O(\log\log(pN_I N_J D) + \log(\ ))$ such that if $c_1 > \log(p) + 3\log(D) - \log(N_J) + _1$ and $c_2 > \log(p) + 3\log(DN_I N_J) + _2$ and M_i is a random integer with $c_i < \log(M_i) < c_i + 1$ for $i = 1, 2$, then* SuborderEichlerNorm$_{D(M_1M_2)}$ *will succeed with probability higher than $1 - 2^{-}$ when I is in a random class of $\mathrm{Cl}(O_0)$. The expected running time is in $O(\text{poly}(\log(pDN_I N_J)))$.*

Proof. The result follows from Proposition 3.4.5 and Lemma 3.1.6. □

3.5 Failures

Most of the size estimates we have presented on the various algorithms introduced in the sections above are based on the heuristic that, for a given class

Algorithm 14 IdealSuborderEichlerNorm$_N(D; I; J)$

Input: An integer D, two left $\mathcal{O}_0$-ideals $I; J$ of inert prime norm $N_I; N_J$ and $\gcd(N_I; N_J; D) = 1$.

Output: $\alpha \in \mathbb{Z} + DI \setminus J$ of norm $N_J N$.

1: Select a random class $(C_2 : D_2) \in \mathbb{P}^1(\mathbb{Z}/D\mathbb{Z})$.
2: Compute γ = FullStrongApproximation$_{N N_J}(D; C_2; D_2)$). If the computation fails go back to Step 1.
3: Compute $(C_0 : D_0)$ = EichlerModConstraint($I; \gamma; 1$).
4: Compute $(C_3 : D_3)$ = IdealModConstraint($J; \gamma$).
5: Sample a random $D_2' \in \mathbb{Z}/D\mathbb{Z}$, compute $C_2' = D_2' C_2 (D_2)^{-1} \bmod D$.
6: Compute $C_1 = \mathrm{CRT}_{N_I; D; N_J}(C_0; C_2'; C_3)$, $D_1 = \mathrm{CRT}_{N_I; D; N_J}(D_0; D_2'; D_3)$.
7: Compute β = FullStrongApproximation$_{N_J N = n(\beta)}(N_I D N_J; C_1; D_1)$). If it fails, go back to step 1.
8: **return** $\alpha = \gamma\beta$.

of ideals, the norm N of the output of EquivalentPrimeIdeal satisfies the bound in Lemma 3.2.3. This will be true with overwhelming probability for a random class, but there are some identifiable counter examples. In fact, by looking at Eq. (3.2.1), it is easy to see that the bad situation will happen when the shortest element $\alpha \in I$ is such that $q_I(\alpha)$ is not prime and is significantly smaller than $p\bar{p}$. Even if the bad ideal classes are rare, we will sometimes encounter them, and this is why we cannot simply dismiss them as too unlikely to happen. In any case, it is always better to be able to handle all the extreme situations. The simplest way to deal with this problem is to increase the size of the elements of N. Indeed, as it is clear from Eq. (3.2.1), the size of N is at most p. Thus, if N contains big enough elements, our algorithms will always be able to succeed.

However, in a number of our applications, this solution will not be acceptable as the size of the elements in N will be tightly constrained. There is a way to handle the problem without increasing the size of the output in most of the cases. In all our algorithm, the size constraint actually comes from the FullStrongApproximation algorithm. As we explained already, we want the input N of FullStrongApproximation to be prime to improve efficiency. However, we also explained that it could be run with an input N that is not prime. Thus, when there exists an equivalent ideal of norm $M \neq p\bar{p}$, we can run FullStrongApproximation with this non-prime M. With good probability, the computation will still succeed in a relatively efficient time (because M will not have too many factors) and obtain a solution whose size will be smaller than usual. There may be some very bad cases, where M has a lot of factors and so FullStrongApproximation will be very inefficient, but those situations will just not happen in practice.

The only case, where this fix will not be enough is in SpecialEichlerNorm because of the additional constraint that $\alpha \in \mathbb{Z} + K$. Indeed, if L is the ideal equivalent to $I(\mathcal{O}_0; \mathcal{O})$ of the smallest possible norm and it happens that $L \subset K$, then the constraint will never be satisfied because the output of FullStrongApproximation will satisfy $\alpha \in \mathbb{Z} + L \subset \mathbb{Z} + K$ by design of the algorithm. If $n(L)$ is coprime to ℓ, this will not happen, but it can occur when $\ell | n(J)$. In summary, SpecialEichlerNorm$_N$ cannot terminate on input $\mathcal{O}; K$ with an output of size roughly $p^{5/4}$ when $\mathcal{O}$ is connected to $\mathcal{O}_0$ with an ideal of very small

norm included in K. Without any further context, it is hard to see how we can overcome the problem, and so we wait for a concrete application of SpecialEichlerNorm to explain how we hope to deal with this obstacle.

Chapter 4

Isogeny representation: algorithmic aspects

In this chapter, we explore ways to represent, manipulate and compute isogenies. A good part of this chapter is made of new contributions, constituted of a mix of results from our articles [BDFLS20, DFKL+20, DFLW22, Ler21].

We begin with the definition of our main topic of interest: isogeny representations. We recall that L_{isog} is the language of isogenous curves (see Definition 1.1.27). Informally, an isogeny representation is a string s_ϕ associated to an isogeny $\phi : E_1 \to E_2$ of degree D. This string can be used as input to two algorithms: one that can verify that the element D, E_1, E_2 is in L_{isog} and one that can compute $\phi(P)$ for some point $P \in E_1$.

We call the former a *verification algorithm* and the latter an *evaluation algorithm*. We can regroup isogeny representations in *families of representations* by looking at the associated verification and evaluation algorithms. Thus, to a family XX of representations we associate two algorithms XXVerification and XXEvaluation.

Since an isogeny of degree $D_1 D_2$ can always be decomposed as a D_1-isogeny and a D_2-isogeny, we focus naturally on the case where $D = \ell$ and ℓ is a prime. In a few occurrences, we will also target the case $D = \ell^e$ for efficiency reasons.

In what follows, we will introduce several families of representations and their verification and evaluations algorithms. This is motivated by isogeny-based cryptography. As we will see in Part II, a number of algorithms introduced in this chapter are going to be useful.

If we go back to Definition 1.1.11, isogenies are rational maps between elliptic curves. As such, the most natural representation of an isogeny ϕ are the rational functions $f_1, f_2 \in K(x,y)$, such that $\phi : (x,y) \mapsto (f_1(x,y), f_2(x,y))$. However, we will see that this representation is far from being the most convenient. It can be shown that the degree of the polynomials involved in the expression of f_1, f_2 scales linearly with the degree ℓ of the isogeny. Very quickly, this representation of isogenies will be too big to handle efficiently. Nonetheless, we consider the rational maps f_1, f_2 as the canonical representation for ϕ and will refer to these when simply saying \an isogeny".

Let us make here a brief summary of the other family isogeny representations that we will consider and of their respective interests. First, we have the kernel

representation that we will treat in Section 4.1. Based on the Velu formul , this is the representation that is most often used in practice and it is de nitely the most e cient to treat isogenies of smooth degree. However, in the generic case, the computational complexity is exponential in the degree. The ideal representation will be the focus of Section 4.2. This representation is based on the results of the Deuring correspondence and it is the most adapted to handle isogenies of arbitrary degree since we have some heuristic polynomial-time algorithms in the degree to perform all the necessary computations. Finally, we will introduce in Section 4.3, the suborder representation. This representation is also based on the Deuring correspondence but in a di erent way. Like the ideal representation, we have some heuristic polynomial-time algorithms to handle most of the task even though they are less e cient for the suborder representation. The interest of this third representation lies in the fact that we believe it is not equivalent to the ideal representation and the gap between these two representations can be used to build cryptography. This will be the focus of Chapter 7.

Through this chapter, unless speci ed otherwise, the isogenies we consider are cyclic separable isogenies.

4.1 The kernel representation

For the result of this section, ` 6= p is a prime number and $q = p^k$ for some $k \geq 1$.

In this section, we state all time complexities in terms of algebraic operations over $\mathbb{F}_q$. In fact, most of our algorithm are algebraic algorithms in the sense of [BCS97], and can further be lifted to algorithms de ned over $\mathbb{Z}[1/2]$ and in some cases over $\mathbb{Z}$. In other words, the algorithms are agnostic to the choice of p and q in $\mathbb{F}_q$, except for sometimes requiring q to be odd; and the algorithms can also be applied to more general rings, as long as all necessary divisions can be carried out.

We have already explicited in Section 1.1 the strong link between an isogeny and its kernel through the correspondence between cyclic `-isogenies and cyclic subgroups of order `. The domain E and a generator P of $\ker \phi$ is what we call a *kernel representation* for ϕ. In this section, we will look at what operations can be done using the kernel representation and how to perform them e ciently.

Let E be an elliptic curve over the nite eld $\mathbb{F}_q$, and let P be a point in $E(\mathbb{F}_q)$ of order `. The point P generates the cyclic subgroup $G \subset E(\mathbb{F}_q)$. And we write E' for the curve E/G and $\phi : E \to E'$ for the isogeny whose kernel is $\langle P \rangle$. This isogeny is de ned over $\mathbb{F}_q$.

Velu introduced formul for ϕ and E' (see [Vel71] and [Koh96, §2.4]): for E de ned by $y^2 = x^3 + a_2x^2 + a_4x + a_6$ and ` 3, we have

$$\phi : (x, y) \mapsto \left(\frac{v_G(x)}{u_G(x)^2}, y \frac{v_G(x)}{u_G(x)^3} \right)$$

where

$$u_G(X) = \prod_{s=1}^{(\ell-1)/2} (X - x([s]P)),$$

$$v_G(X) = 4(X^3 + a_2X^2 + a_4X + a_6)(u_G'(X)^2 - u_G''(X)u_G(X)) - 2(3X^2 + 2a_2X + a_4)u_G'(X)u_G(X) + (\ell X - \sum_{s=1}^{\ell-1} x([s]P))u_G(X)^2,$$

$$w_G(X) = u_G'(X)v_G(X) - 2u_G(X)v_G'(X).$$

Definition 4.1.1. The polynomial $\_G$ in the Velu formula is the *kernel polynomial* of $'$.

With the Velu formul , we see that we can very naturally design an algorithm KernelToIsogeny that takes a generator P of G and computes the isogeny $'$ from $\_G$; $\_G$ and $\_G$ (we will explain later that, for our applications, computing the kernel polynomial is enough). We are not going to give a detailed description of this algorithm because it is quite straightforward from the Velu formul . We can easily verify that the degree of the polynomials $\_G$; $\_G$ and $\_G$ is in $O(\ell)$. Thus, when using e cient algorithms to compute a polynomial and its derivatives from its roots, we get that KernelToIsogeny takes $\tilde{\Theta}(\ell)$ operations over $\mathbb{F}_q$. From the output of KernelToIsogeny, $'$ can be evaluated in $\tilde{\Theta}(\ell)$. To compute the de ning equation of E^0, we can evaluate $'(Q)$ for a few Q outside G, possibly after extending $\mathbb{F}_q$, and then interpolate a curve equation through the resulting points. Alternatively, Velu gives further formulas for the de ning equation. In any case, this computation can be done in $\tilde{\Theta}(\ell)$-operations over $\mathbb{F}_q$. In conclusion, there is an algorithm KernelVeri cation that takes in input a triple $\ell; E; E^0$ (where $E; E^0$ are de ned over $\mathbb{F}_q$) and a non-trivial point $P \in E[\ell](\mathbb{F}_q)$ and outputs 1 if and only $(\ell; E; E^0) \in L_{\text{isog}}$ in $\tilde{\Theta}(\ell)$-operations over $\mathbb{F}_q$. The Velu formlu also give directly an algorithm to evaluate the isogeny $'$ on any point of E. This proves that the kernel representation is a family of isogeny representation.

Conversely, given the isogeny $'$, we can compute a generator of $\ker '$. We call this algorithm IsogenyToKernel and similarly to KernelToIsogeny we do not include a detailed description. The idea is quite simple: compute a basis $\langle P_1; P_2 \rangle$ of $E[\ell]$ and compute $'(P_1); '(P_2)$. Then compute the discrete logarithm of $'(P_2)$ in the subgroup generated by $'(P_1)$ (or the converse if it failed) to nd a linear combination of $P_1; P_2$ of order ℓ that is sent to 0_{E^0} by $'$. The complexity of IsogenyToKernel is $\tilde{\Theta}(\ell)$ operations over $\mathbb{F}_q$.

Up to this point, everything that we said on the kernel representation is pretty standard in the literature. The main contribution of this section is to improve upon the linear complexity of the algorithm we just outlined. We achieve a quadratic speed-up with algorithms of complexity $\tilde{\Theta}(\sqrt{\ell})$ that we will describe below. Before getting to that, we give some additional information on the kernel representation.

E cient isogeny computation. Let us go back to the case of an isogeny $'$ of abstract degree D for a time. Neither the algorithms KernelToIsogeny, IsogenyToKernel we described above nor the improvements that we will present later have a polynomial complexity in D when D is prime. We are going to introduce in Sections 4.2 and 4.3 two isogeny representations that let us manipulate e ciently isogenies of arbitrary degree when the endomorphism ring of E is known, but we are not aware of any generic algorithm when we have no information on $\mathrm{End}(E)$. Note that contrary to the two aforementioned isogeny representations, the kernel representation does not require having any information on $\mathrm{End}(E)$. One might wonder if there is a case where we can hope to have e cient algorithms for the kernel representation. Fortunately, the answer to that question is yes. Indeed, we have mentioned several times already that any isogeny of composite degree $D_1 D_2$ could be seen as a D_1-isogeny composed

with a D_2-isogeny. This implies that if D is a B-smooth integer, we can get algorithms of complexity $\tilde{O}(B \log(D))$ over the field of definition for the D-torsion by treating independently each isogeny of prime degree involved in the decomposition of the isogeny φ'. This approach gives a polynomial algorithm when the smoothness bound $B = O(\text{poly}(\log(D)))$. Since we only have an exponential algorithm to treat prime degrees, we cannot hope to do better than that.

Remark 4.1.2. We made the choice of counting the number of operations over $\mathbb{F}_q$ to simplify our analysis in the next sections. However, to get a more practical estimate, we would also need to consider the degree of the extension $\mathbb{F}_q/\mathbb{F}_p$. Indeed, the cost (in operations over $\mathbb{F}_p$) of operations over $\mathbb{F}_q$ is polynomial in $[\mathbb{F}_q : \mathbb{F}_p]$. Even when D is smooth, we can see from Proposition 1.1.23 that we cannot exclude that $\mathbb{F}_q/\mathbb{F}_p$ has large degree. For a generic p, the only way to ensure that $[\mathbb{F}_q : \mathbb{F}_p] = O(\text{poly}(\log(D)))$ is if D is powersmooth and each prime power factor of D is smaller than $B' = O(\text{poly}(\log(D)))$. For some specific choices of p and degree D, we can get that $[\mathbb{F}_q : \mathbb{F}_p]$ is small enough to get efficient algorithms when D is smooth but not powersmooth. These restrictions on the degrees that can be handled efficiently from the kernel representation will have a big impact on the choices of parameter for the protocols in isogeny-based cryptography. Most of the time, we need to use very specific values of p and D. However, with the efficient isogeny representations from Sections 4.2 and 4.3, we will be able to use big prime degrees in some cases. In fact, the protocols that we present in Part II are the first examples of schemes in isogeny-based cryptography that are making use explicitly of big prime degrees instead of smooth ones.

Compact kernel representation. We have discussed efficiency at length, but compactness is another issue. Fortunately, the prospects are slightly better in that department for the kernel representation. Indeed, we show with Lemma 4.1.3 that the kernel representation can always be compacted in a string of polynomial size even if the kernel points are defined over a big extension of $\mathbb{F}_p$. Of course, Lemma 4.1.3 says nothing about the efficiency of the algorithm required to compute that compressed representation. The compression technique underlying Lemma 4.1.3 is quite standard in isogeny-based cryptography [ZSP+18, NR19, AJK+16, CJL+17, PDJ20]. In Section 4.1.5, we will see a concrete and efficient algorithm to compress an isogeny of degree 2^e based on this principle.

Lemma 4.1.3. *A cyclic isogeny of degree D can be represented as a string of size $O(\log(pD))$.*

Proof. We will be making use of the kernel representation to compress the cyclic isogeny φ' of degree D. First, one needs the starting curve E, which can be described in $O(\log(p))$ using the j-invariant. Let P, Q be a basis of $E[D]$. A generator of the kernel of φ', can always be expressed as a linear combination of P, Q whose coefficients x, y are smaller than D. In the end, it suffices to publish $j(E), x, y$ to obtain a representation of φ' of size $O(\log(pN))$. If the basis P, Q can be computed canonically from $j(E)$, anyone will be able to find $\ker \varphi'$ from E, x, y □

In the rest of this section, we show how to get faster algorithms, in particular to evaluate the kernel representation. The results that we present are the main

contributions of our joint publication [BDFLS20] with Bernstein, De Feo and Smith. In our cryptographic applications, the full computation of the isogeny involved is often not necessary. First, we often work in x-only coordinates for e ciency and size reasons, which means that focusing on the kernel polynomial $_G$ will be enough. Second, most of the time, we simply need to be able to evaluate our isogeny on a few well-chosen points. For this reason, we will focus mainly on evaluating the kernel polynomial as e ciently as possible. Our main result on that matter is introduced in Section 4.1.3, and the two sections Sections 4.1.1 and 4.1.2 can be seen as preliminaries to review related works. In Section 4.1.4, we will explain concretely how to use this result for our purpose.

4.1.1 A generalization of the problem

Our algorithmic problem falls naturally into a more general framework: the e cient evaluation of polynomials and rational functions over $\mathbb{F}_q$ whose roots are values of a function from a cyclic group to $\mathbb{F}_q$.

Fix a cyclic group G (which we will write additively), a generator P of G, and a function $f : G \to \mathbb{F}_q$. For each nite subset S of $\mathbb{Z}$, we de ne a polynomial

$$h_S(X) = \prod_{s \in S} (X - f([s]P));$$

where $[s]P$ denotes the s-th scalar multiple of P. The kernel polynomial $_G(x)$ above is an example of this, with $f = x$ and $S = \{1, \ldots, (\ell - 1)/2\}$. Another example is the cyclotomic polynomial $_n$, where f embeds $\mathbb{Z}/n\mathbb{Z}$ in the roots of unity of $\mathbb{F}_q$, and $_n(X) = h_S(X)$ where $S = \{i \mid 0 < i < n, \gcd(i, n) = 1\}$. More generally, if f maps $i \mapsto {}^i$ for some , then $h_S(X)$ is a polynomial whose roots are various powers of ; similarly, if f maps $i \mapsto i$ for some , then $h_S(X)$ is a polynomial whose roots are various integer multiples of .

Given f and S, then, we want to compute $h_S( ) = \prod_{s \in S}(- f([s]P))$ for any in $\mathbb{F}_q$. One can always directly compute $h_S( )$ in $O(\#S)$ $\mathbb{F}_q$-operations; this is the standard way to compute $_G( )$. But if S has enough additive structure, and if f is su ciently compatible with the group structure on G, then we can compute $h_S( )$ in $\tilde{O}(\sqrt{\#S})$ $\mathbb{F}_q$-operations. Pollard [Pol74] and Strassen [Str76] were the rst to introduce this kind of ideas for deterministic factorization. Those ideas were later reused in various contexts: we can mention the recent generalization of Bostan [Bos20] for q-holonomic sequences.

Our main theoretical result on this subject is Theorem 4.1.9, which shows how to achieve this quasi-square-root complexity for a large class of S when f is the x-coordinate function on an elliptic curve. We apply this to the special case of e cient ℓ-isogeny computation in Section 4.1.4.

We focus on asymptotic exponents, in particular improving ℓ-isogeny evaluation from cost $\tilde{O}(\ell)$ to cost $\tilde{O}(\sqrt{\ell})$. However, this analysis hides polylogarithmic factors that can dominate the asymptotic improvement for small ℓ. At the end of Section 4.1.4, we discuss brie y concrete e ciency and compare with the conventional method.

4.1.2 Evaluation of polynomials whose roots are powers

In this section, we introduce the standard algorithm by Pollard to evaluate e ciently a polynomial whose roots (with multiplicity) form a geometric pro-

gression.

Fix $\alpha \in (\mathbb{Z}/n\mathbb{Z})^*$, we will consider this as an implicit parameter of our algorithm below. Define $G = \mathbb{Z}$ and define $h_S(X) = \prod_{s \in S}(X - \alpha^s) \in (\mathbb{Z}/n\mathbb{Z})[X]$ for $S = \{1, 2, 3, \ldots, \ell\}$. Given $\beta \in \mathbb{Z}/n\mathbb{Z}$, one can straightforwardly evaluate $h_S(\beta)$ for this S using $O(\ell)$ algebraic operations in $\mathbb{Z}/n\mathbb{Z}$ and we will show how to accomplish the same result using only $\tilde{O}(\sqrt{\ell})$ operations.

Outline of the algorithm. Compute $b = b\sqrt{\ell}c$, and define $I = \{1, 2, 3, \ldots, b\}$ and $J = \{0, b, 2b, \ldots, (b-1)b\}$. A multipoint evaluation can be used on the polynomial $h_I(X) = \prod_{i=1}^{b}(X - \alpha^i)$ to compute $h_I(\beta\alpha^j)$ for all $j \in J$. Multiply α^{jb} by $h_I(\beta\alpha^j)$ to obtain $\prod_{i=1}^{b}(\beta\alpha^{i+j})$ for each j, and then multiply across $j \in J$ to obtain $\prod_{s=1}^{b^2}(\beta\alpha^s)$. Finally, multiply by $\prod_{s=b^2+1}^{\ell}(\beta\alpha^s)$ to obtain the desired $h_S(\beta)$. One can also view the product $\prod_{s=1}^{b^2}(\beta\alpha^s)$ as the resultant of two degree-b polynomials. Specifically, $\prod_j h_I(\beta\alpha^j)$ is the resultant of $\prod_j (X - \beta\alpha^j)$ and h_I; and $\prod_j \alpha^{jb} h_I(\beta\alpha^j)$ is the resultant of $\prod_j (\alpha^j X - \beta)$ and h_I. Generic resultant algorithms can also be used to evaluate $h_S(\beta)$.

Structures in S and f. We highlight two structures exploited in the above computation of $\prod_{s=1}^{\ell}(\beta\alpha^s)$. First, the set $S = \{1, 2, \ldots, \ell\}$ has enough additive structure to allow most of it to be decomposed as $I + J$, where I and J are much smaller sets. Second, the map $s \mapsto \alpha^s$ is a group homomorphism, allowing each α^{i+j} to be computed as the product of α^i and α^j.

We now formalize the statement regarding additive structure, focusing on the $\mathbb{F}_q$ case that we will need later in the paper. First, some terminology: we say that sets of integers I and J have *no common differences* if $i_1 - i_2 \neq j_1 - j_2$ for all $i_1 \neq i_2$ in I and all $j_1 \neq j_2$ in J. If I and J have no common differences, then the map $I \times J \to I + J$ sending (i, j) to $i + j$ is a bijection.

Lemma 4.1.4. *Let q be a prime power. Let α be an element of $\mathbb{F}_q$. Define $h_S(X) = \prod_{s \in S}(X - \alpha^s) \in \mathbb{F}_q[X]$ for each finite subset S of $\mathbb{Z}$. Let I and J be finite subsets of $\mathbb{Z}$ with no common differences. Then*

$$h_{I+J}(X) = \mathrm{Res}_Z(h_I(Z), H_J(X, Z))$$

where $\mathrm{Res}_Z(\ ,\ )$ is the bivariate resultant, and

$$H_J(X, Z) := \prod_{j \in J} (X - \alpha^j Z).$$

Proof. $\mathrm{Res}_Z(h_I(Z), H_J(X, Z)) = \prod_{i \in I} \prod_{j \in J}(X - \alpha^{i+j}) = \prod_{(i,j) \in I \times J}(X - \alpha^{i+j}) = h_{I+J}(X)$ since the map $I \times J \to I + J$ sending (i, j) to $i + j$ is bijective. □

We now give a detailed description of the algorithm outlined above as Algorithm 15.

Proposition 4.1.5. *Let q be a prime power. Let α be an element of $\mathbb{F}_q$. Let I, J be finite subsets of $\mathbb{Z}$ with no common differences. Let K be a finite subset of $\mathbb{Z}$ disjoint from $I + J$. Given β in $\mathbb{F}_q$, Algorithm 15 outputs $\prod_{s \in S}(\beta\alpha^s)$ using $\tilde{O}(\max(\#I, \#J, \#K))$ $\mathbb{F}_q$-operations, where $S = (I + J) \cup K$.*

Algorithm 15 Computing $h_S(\ ) = \prod_{s \in S}(\ - \alpha^s)$

Input: a prime power q; $\alpha \in \mathbf{F}_q$; finite subsets $I, J, K \subseteq \mathbf{Z}$ such that I and J have no common differences and $(I + J) \setminus K = \{s\}$; α^s for each $s \in I \cup J \cup K$ in $\mathbf{F}_q$

Output: $h_S(\ )$ where $h_S(X) = \prod_{s \in S}(X - \alpha^s)$ and $S = (I + J) \cup K$

$h_I \leftarrow \prod_{i \in I}(Z - \alpha^i) \in \mathbf{F}_q[Z]$

$H_J \leftarrow \prod_{j \in J}(\ - \alpha^j Z) \in \mathbf{F}_q[Z]$

$h_{I+J} \leftarrow \prod \mathrm{Res}_Z(h_I, H_J) \in \mathbf{F}_q$

$h_K \leftarrow \prod_{k \in K}(\ - \alpha^k) \in \mathbf{F}_q$

return $h_{I+J}\ h_K$

Proof. Since $S \setminus K = I + J$, we have $h_S(\ ) = h_{I+J}(\ )\ h_K(\ )$, and Lemma 4.1.4 shows that $h_{I+J}(\ ) = \mathrm{Res}_Z(h_I(Z), H_J(\ , Z))$. Line 1 computes $h_I(Z)$ in $\tilde{O}(\#I)$ $\mathbf{F}_q$-operations; Line 2 computes $H_J(\ , Z)$ in $\tilde{O}(\#J)$ $\mathbf{F}_q$-operations; Line 3 computes $h_{I+J}(\ )$ in $\tilde{O}(\max(\#I, \#J))$ $\mathbf{F}_q$-operations; and Line 4 computes $h_K(\ )$ in $\tilde{O}(\#K)$ $\mathbf{F}_q$-operations. The total is $\tilde{O}(\max(\#I, \#J, \#K))$ $\mathbf{F}_q$-operations. □

4.1.3 Evaluation of the polynomial whose roots are in an elliptic curve torsion subgroup.

The technique in §4.1.2 for evaluating polynomials whose roots are powers is well known. Our main theoretical contribution is to adapt this to polynomials whose roots are functions of more interesting groups: in particular, functions of elliptic-curve torsion points. The most important such function is the x-coordinate. The main complication here is that, unlike in §4.1.2, the function x is not a homomorphism.

The elliptic setting. Let $E/\mathbf{F}_q$ be an elliptic curve, let $P \in E(\mathbf{F}_q)$, and define $G = \langle P \rangle$. Let S be a finite subset of $\mathbf{Z}$. We want to evaluate

$$h_S(X) = \prod_{s \in S}(X - f([s]P)); \qquad \text{where} \qquad f : Q \mapsto \begin{cases} 0 & \text{if } Q = 0; \\ x(Q) & \text{if } Q \neq 0; \end{cases}$$

at some α in $\mathbf{F}_q$. Here $x : E \to E/\langle 1 \rangle = \mathbf{P}^1$ is the usual map to the x-line.

Adapting Algorithm 15 to this setting is not a simple matter of replacing the multiplicative group with an elliptic curve. Indeed, Algorithm 15 explicitly uses the homomorphic nature of $f : s \mapsto \alpha^s$ to represent the roots α^s as $\alpha^i \alpha^j$ where $s = i + j$. This presents an obstacle when moving to elliptic curves: $x([i + j]P)$ is not a rational function of $x([i]P)$ and $x([j]P)$, so we cannot apply the same trick of decomposing most of S as $I + J$ before taking a resultant of polynomials encoding $f(I)$ and $f(J)$. The isogenies context is different: we need a product of functions of $x([i + j]P)$.

Fortunately, even if the x-map is not homomorphic, there is an algebraic relation between $x(P)$, $x(Q)$, $x(P + Q)$, and $x(P - Q)$, which we will review below. The introduction of the difference $x(P - Q)$ as well as the sum $x(P + Q)$ requires us to replace the decomposition of most of S as $I + J$ with a decomposition involving $I + J$ and $I - J$, which we will formalize in the next paragraph.

Biquadratic relations on x-coordinates. Lemma 4.1.6 recalls the general relationship between $x(P)$, $x(Q)$, $x(P+Q)$, and $x(P-Q)$. Example 1 gives explicit formul for the case that is most useful in isogeny-based cryptography.

Lemma 4.1.6. *Let q be a prime power. Let $E/\mathbb{F}_q$ be an elliptic curve. There exist biquadratic polynomials F_0, F_1, and F_2 in $\mathbb{F}_q[X_1, X_2]$ such that*

$$(X - x(P+Q))(X - x(P-Q)) = X^2 + \frac{F_1(x(P), x(Q))}{F_0(x(P), x(Q))}X + \frac{F_2(x(P), x(Q))}{F_0(x(P), x(Q))}$$

for all P and Q in E such that $0 \notin \{P, Q, P+Q, P-Q\}$.

Proof. The existence of F_0, F_1, and F_2 is classical (see e.g. [Cas91, p. 132] for the F_i for Weierstrass models); indeed, the existence of such biquadratic systems is a general phenomenon for theta functions of level 2 on abelian varieties (see e.g. [Mum66, §3]). □

Example 1 (Biquadratics for Montgomery models). If E is de ned by an a ne equation $By^2 = x(x^2 + Ax + 1)$, then the polynomials of Lemma 4.1.6 are

$$F_0(X_1, X_2) = (X_1 - X_2)^2,$$
$$F_1(X_1, X_2) = -2((X_1X_2 + 1)(X_1 + X_2) + 2AX_1X_2),$$
$$F_2(X_1, X_2) = (X_1X_2 - 1)^2.$$

The symmetric triquadratic polynomial $(X_0X_1 - 1)^2 + (X_0X_2 - 1)^2 + (X_1X_2 - 1)^2 - 2X_0X_1X_2(X_0 + X_1 + X_2 + 2A) - 2$ is $X_0^2F_0(X_1, X_2) + X_0F_1(X_1, X_2) + F_2(X_1, X_2)$.

Montgomery curves $By^2 = x(x^2 + Ax + 1)$, and the remarkably simple formula $(X_1X_2 - 1)^2/(X_1 - X_2)^2$ for the product $x(P+Q)x(P-Q)$ on these curves, were introduced by Montgomery in [Mon87, Section 10.3.1].

An extended index system. In §4.1.2, we represented most of S as $I + J$; requiring I and J to have no common di erences ensured this representation had no redundancy. Now we will represent most elements of S as elements of $(I + J) \cup (I - J)$, so we need a stronger restriction on I and J to avoid redundancy.

De nition 4.1.7. Let I and J be nite sets of integers.

1. We say that (I, J) is an *index system* if the maps $I \times J \to \mathbb{Z}$ de ned by $(i, j) \mapsto i + j$ and $(i, j) \mapsto i - j$ are both injective and have disjoint images.
2. If S is a nite subset of $\mathbb{Z}$, then we say that an index system (I, J) is an *index system for S* if $I + J$ and $I - J$ are both contained in S.

If (I, J) is an index system, then the sets $I + J$ and $I - J$ are both in bijection with $I \times J$. We write $I \pm J$ for the union of $I + J$ and $I - J$.

Example 2. Let m be an odd positive integer, and consider the set

$$S = \{1, 3, 5, \ldots, m\}$$

in arithmetic progression. Let

$$I := \{2b(2i + 1) \mid 0 \le i < b'\} \quad \text{and} \quad J := \{2j + 1 \mid 0 \le j < b\}$$

where $b = \lfloor \sqrt{m+1}/2c \rfloor$; $b' = \lfloor (m+1)/4bc \rfloor$ if $b > 0$; and $b' = 0$ if $b = 0$. Then (I, J) is an index system for S, and $S \setminus (I \pm J) = K$ where $K = \{4bb' + 1, \ldots, m - 2, m\}$. If $b > 0$ then $\#I = b' \cdot b + 2$, $\#J = b$, and $\#K \le 2b - 1$.

The elliptic resultant. We are now ready to adapt the results of §4.1.2 to the setting of the kernel polynomial. Our main tool is Lemma 4.1.8, which expresses $h_{I\ J}$ as a resultant of smaller polynomials.

Lemma 4.1.8. *Let q be a prime power. Let $E/\mathbb{F}_q$ be an elliptic curve. Let P be an element of $E(\mathbb{F}_q)$. Let n be the order of P. Let $(I;J)$ be an index system such that I, J, $I+J$, and $I-J$ do not contain any elements of $n\mathbb{Z}$. Then*

$$h_{I\ J}(X) = \frac{1}{\Delta_{I;J}} \operatorname{Res}_Z(h_I(Z); E_J(X;Z))$$

where

$$E_J(X;Z) := \prod_{j\in J} F_0(Z;x([j]P))X^2 + F_1(Z;x([j]P))X + F_2(Z;x([j]P))$$

and $\Delta_{I;J} := \operatorname{Res}_Z(h_I(Z); D_J(Z))$ *where* $D_J(Z) := \prod_{j\in J} F_0(Z;x([j]P))$.

Proof. Since $(I;J)$ is an index system, $I+J$ and $I-J$ are disjoint, and therefore we have $h_{I\ J}(X) = h_{I+J}(X)\ h_{I\ J}(X)$. Expanding and regrouping terms, we get

$$\begin{aligned} h_{I\ J}(X) &= \prod_{(i;j)\in I\ J} (X - x([i+j]P))(X - x([i-j]P)) \\ &= \prod_{i\in I}\prod_{j\in J} X^2 + \frac{F_1(x([i]P);x([j]P))}{F_0(x([i]P);x([j]P))}X + \frac{F_2(x([i]P);x([j]P))}{F_0(x([i]P);x([j]P))} \end{aligned}$$

by Lemma 4.1.6. Factoring out the denominator, we find

$$h_{I\ J}(X) = \frac{\prod_{i\in I} E_J(X;x([i]P))}{\prod_{i\in I}\prod_{j\in J} F_0(x([i]P);x([j]P))} = \frac{\prod_{i\in I} E_J(X;x([i]P))}{\prod_{i\in I} D_J(x([i]P))};$$

and finally $\prod_{i\in I} E_J(X;x([i]P)) = \operatorname{Res}_Z(h_I(Z);E_J(X;Z))$ and $\prod_{i\in I} D_J(x([i]P)) = \operatorname{Res}_Z(h_I(Z);D_J(Z)) = \Delta_{I;J}$, which yields the result. □

We introduce as Algorithm 16, our efficient method to evaluate the polynomial h_S on a point . Theorem 4.1.9 proves its correctness and run time.

Theorem 4.1.9. *Let q be a prime power. Let $E/\mathbb{F}_q$ be an elliptic curve. Let P be an element of $E(\mathbb{F}_q)$. Let n be the order of P. Let $(I;J)$ be an index system for a finite set $S \subset \mathbb{Z}$. Assume that I, J, and S contain no elements of $n\mathbb{Z}$. Given in $\mathbb{F}_q$, Algorithm 16 computes*

$$h_S(\ ) = \prod_{s\in S} x([s]P)$$

in $\tilde{O}(\max(\#I;\#J;\#K))$ $\mathbb{F}_q$-operations, where $K = S \setminus (I - J)$.

In particular, if $\#I$, $\#J$, and $\#K$ are in $\tilde{O}(\sqrt{\#S})$, then Algorithm 16 computes $h_S(\ )$ in $\tilde{O}(\sqrt{\#S})$ $\mathbb{F}_q$-operations. The $\tilde{O}$ is uniform in q. Instead of taking P and various $x([s]P)$ as parameters, Algorithm 16 can take P as an input, at the cost of computing the relevant multiples of P.

Algorithm 16 Computing $h_S(\ ) = \prod_{s \in S} (\ - x([s]P))$ for $P \in E(\mathbb{F}_q)$

Input: a prime power q; an elliptic curve $E/\mathbb{F}_q$; $P \in E(\mathbb{F}_q)$; a finite subset $S \subset \mathbb{Z}$; an index system (I, J) for S such that $S \setminus n\mathbb{Z} = I \setminus n\mathbb{Z} = J \setminus n\mathbb{Z} = \{\}$, where n is the order of P; $x([s]P)$ for each $s \in I \cup J \cup K$ in $\mathbb{F}_q$

Output: $h_S(\ )$ where $h_S(X) = \prod_{s \in S}(X - x([s]P))$

1: $h_I \leftarrow \prod_{i \in I}(Z - x([i]P)) \in \mathbb{F}_q[Z]$
2: $D_J \leftarrow \prod_{j \in J} F_0(Z, x([j]P)) \in \mathbb{F}_q[Z]$
3: $\_{I,J} \leftarrow \mathrm{Res}_Z(h_I, D_J) \in \mathbb{F}_q$
4: $E_J \leftarrow \prod_{j \in J} F_0(Z, x([j]P))^2 + F_1(Z, x([j]P)) + F_2(Z, x([j]P)) \in \mathbb{F}_q[Z]$
5: $R \leftarrow \mathrm{Res}_Z(h_I, E_J) \in \mathbb{F}_q$
6: $h_K \leftarrow \prod_{k \in S \setminus n(I \cup J)}(\ - x([k]P)) \in \mathbb{F}_q$
7: **return** $h_K \cdot R / \_{I,J}$.

Proof. The proof follows that of Proposition 4.1.5. Since $S \setminus nK = I \cup J$, we have $h_S(\ ) = h_{I \cup J}(\ ) \cdot h_K(\ )$. Using the notation of Lemma 4.1.8: Line 1 computes $h_I(Z)$ in $\tilde{\mathcal{O}}(\#I)$ $\mathbb{F}_q$-operations; Line 2 computes $D_J(Z)$ in $\tilde{\mathcal{O}}(\#J)$ $\mathbb{F}_q$-operations; Line 3 computes $\_{I,J}$ in $\tilde{\mathcal{O}}(\max(\#I, \#J))$ $\mathbb{F}_q$-operations; Line 4 computes $E_J(\ , Z)$ in $\tilde{\mathcal{O}}(\#J)$ $\mathbb{F}_q$-operations; Line 5 computes $\mathrm{Res}_Z(h_I(Z), E_J(\ , Z))$, which is the same as $\_{I,J} h_{I \cup J}(\ )$ by Lemma 4.1.8, in $\tilde{\mathcal{O}}(\max(\#I, \#J))$ $\mathbb{F}_q$-operations; Line 6 computes $h_K(\ )$ in $\tilde{\mathcal{O}}(\#K)$ $\mathbb{F}_q$-operations; returns $h_S(\ ) = h_K(\ ) \cdot h_{I \cup J}(\ )$. The total number of $\mathbb{F}_q$-operations is in $\tilde{\mathcal{O}}(\max(\#I, \#J, \#K))$. □

4.1.4 Application to isogeny computations

Coming back to our problem of evaluating kernel polynomials for isogenies of prime degree ℓ, we have:

$$\_G(X) = h_S(X) = \prod_{s \in S} (X - x([s]P)) \quad \text{where} \quad S = \{1, 3, \ldots, \ell - 2\}$$

(the set S may be replaced by any set of representatives of $((\mathbb{Z}/\ell\mathbb{Z}) \setminus \{0\})/\{\pm 1\}$). Following Example 2, let $I = \{2b(2i+1) \mid 0 \le i < b'\}$ and $J = \{1, 3, \ldots, 2b - 1\}$ with $b = \lfloor\sqrt{\ell - 1}/2\rfloor$ and (for $b > 0$) $b' = \lfloor(\ell - 1)/4b\rfloor$; then (I, J) is an index system for S, and Algorithm 16 computes $h_S(\ ) = \_G(\ )$ for any $\ $ in $\mathbb{F}_q$ in $\tilde{\mathcal{O}}(\sqrt{\ell})$ $\mathbb{F}_q$-operations.

Below, we give more explanations as to why evaluating $\_G$ on a few selected points is sufficient for our purpose, which can be divided in two main tasks: evaluating isogenies and computing the codomain. We focus on the x-coordinates in the Montgomery setting as it is the most standard for concrete instantiation.

Evaluating isogenies. Let $E/\mathbb{F}_q : y^2 = x(x^2 + Ax + 1)$ be an elliptic curve in Montgomery form, and let P be a point of prime order $\ell \ge 2$ in $E(\mathbb{F}_q)$. Costello and Hisil give explicit formulæ in [CH17] for a quotient isogeny $\varphi : E \to E'$ with kernel $G = \langle P \rangle$ such that $E'/\mathbb{F}_q : y^2 = x(x^2 + A'x + 1)$ is a Montgomery curve:

$$\varphi : (X, Y) \mapsto (\varphi_x(X), c_0 Y \varphi'_x(X))$$

where $c_0 = \prod_{0<s<\ell/2} x([s]P)$ and

$$\varphi_x(X) = X \prod_{0<s<\ell} \frac{x([s]P)X - 1}{X - x([s]P)}. \tag{4.1.1}$$

See [Ren18] for generalizations and a different proof, and see the earlier paper [MS16] for analogous Edwards-coordinate formulas.

Our main goal is to evaluate φ on the level of x-coordinates: that is, to compute $\varphi_x(\alpha)$ given $\alpha = x(Q)$ for Q in $E(\mathbb{F}_q)$.

When the y-coordinate of $\varphi(Q)$ is also required, one needs to compute $\varphi'_x(\alpha)$ together with $\varphi_x(\alpha)$. This can be done with simple adaptations of the technique presented above in Algorithm 16 (see [BDFLS20] for more details).

To compute $\varphi_x(\alpha)$, we rewrite Eq. (4.1.1) as

$$\varphi_x(X) = \frac{X^\ell \cdot h_S(1/X)^2}{h_S(X)^2} \quad \text{where} \quad S = \{1, 3, \ldots, \ell - 2\}.$$

Computing $\varphi_x(\alpha)$ thus reduces to two applications of Algorithm 4.1.9, using (for example) the index system (I, J) for S in Example 2. The constant $\Delta_{I,J}$ appears with the same multiplicity in the numerator and denominator, so we need not compute it. All divisions in the computation are by nonzero field elements except in the following cases, which can be handled separately: if $Q = 0$ then $\varphi(Q) = 0$; if $Q \neq 0$ but $h_S(\alpha) = 0$ for $\alpha = x(Q)$ then $\varphi(Q) = 0$; if $Q = (0, 0)$ then $\varphi(Q) = (0, 0)$.

Computing codomain curves Our other main task is to determine the coefficient A' in the defining equation of E'.

One approach is as follows. We can now efficiently compute $\varphi(Q)$ for any Q in $E(\mathbb{F}_q)$. Changing the base ring from $\mathbb{F}_q$ to $R = \mathbb{F}_q[\,]/(\,^2 + A\, + 1)$ (losing a small constant factor in the cost of evaluation) gives us $\varphi(Q)$ for any Q in $E(R)$. In particular, $Q = (\ , 0)$ is a point in $E[2](R)$, and computing $\varphi(Q) = (\ ', 0)$ reveals $A' = -(\ ' + 1/\ ')$. An alternative — at the expense of taking a square root, which is no longer a q-independent algebraic computation — is to find a point $(\ , 0)$ in $E(\mathbb{F}_{q^2})$ with $\neq 0$. Sometimes is in $\mathbb{F}_q$, and then extending to $\mathbb{F}_{q^2}$ is unnecessary.

Another approach is to use explicit formulas for A'. The formulas from [CH17] give $A' = c_0^2(A\, - 3\ )$ where $c_0^2 = \prod_{0<s<\ell} x([s]P)$ and $\ = \sum_{0<s<\ell}(x([s]P) - 1/x([s]P))$. As pointed out in [MR18] in the context of CSIDH, one can instead transform to twisted Edwards form and use the formulas from [MS16], obtaining $A' = 2(1 + d)/(1 - d)$ where

$$d = \frac{A - 2}{A + 2}^\ell \prod_{s \in S} \frac{x([s]P) - 1}{x([s]P) + 1}^8 = \frac{A - 2}{A + 2}^\ell \cdot \frac{h_S(1)}{h_S(-1)}^8.$$

We can thus compute A' using $\tilde{O}(\sqrt{\ell})$ operations: every task we need can be performed by some evaluations of h_S and some (asymptotically negligible) operations.

Concrete efficiency. Asymptotic improvements rarely translate in a concrete improvement for all size of inputs. Thus, the question is: at what point the new algorithm outperforms the old one? Our new method is no exception. We have made various implementations in different languages to try and determine when our improved method to evaluate the Velu formulæ becomes better than the conventional one in the case of the small primes used in the CSIDH scheme [CLM+18], where we have $\mathbb{F}_q = \mathbb{F}_p$ for a 512-bits prime p. While the precise cutoff depends on various things, we can estimate that the new method is worth using for primes degrees bigger than roughly 100 over a prime characteristic of cryptographic sizes. We will see in Part II that this is well within the range of degrees required by some of our protocols. More analysis and benchmarks can be found in [BDFLS20]. In the implementations that we present for the protocols of Part II, we used our new algorithm to deal with the isogeny computations for degrees over 100.

4.1.5 A compressed representation

In this section, we give an example of the compression/decompression mechanism underlying Lemma 4.1.3 in the case of a degree contained in $2^{\cdot}$. We will use these algorithms for the protocols in Part II. We will provide another example with the G-CGL hash function introduced in Section 6.2.1

Discrete logarithm algorithms. In Compression, Decompression and later in the algorithms of Sections 4.2 and 4.3, we will need to solve some discrete logarithms. We write DLP_D for the algorithm taking $Q, P \in E[D]$ with $Q \in \langle P \rangle$ and that outputs the scalar a such that $Q = [a]P$. We will also use the bi-dimensional version BiDLP_D that takes $Q, P_1, P_2 \in E[D]$ with P_1, P_2 a basis of $E[D]$ and outputs a, b such that $Q = [a]P_1 + [b]P_2$. We write xDLP and xBiDLP, the variant that takes the x-coordinates instead. The complexity is in $O(\sqrt{D})$ operations over the field of definition of $E[D]$. The worst case is when D is prime. The complexity of the composite case reduces to the complexity of each coprime component with the classical Pohlig-Hellman method.

We put ourselves in the setting where the 2^f-torsion is defined over $\mathbb{F}_{p^2}$ and e is equal to vf. In that case, if φ is the isogeny to be compressed, we can decompose $\varphi = \varphi_v \circ \cdots \circ \varphi_1$. We highlight that the strategy inspiring Algorithms 17 and 18 is quite classical and this is why we introduce Compression, Decompression without a formal proof of correctness. We present a slightly faster version than what is usual by using a less compact representation. The idea is to add a few bits of information for every φ_j. We choose arbitrarily to take 4 for the number of these bits, but we stress that this choice can be adjusted. For the sake of explanation, let us assume that each φ_j has degree exactly 2^f. We use a canonical way to sample pseudo-random points on any supersingular curve E. This means that anyone knowing a curve E can agree on an ordered list of points $P_1^E, P_2^E, P_3^E, \ldots$ on $E(\mathbb{F}_{p^2})$. This is classical for key compression, and we refer to the sources mentioned in the beginning of the section for more details. We keep this notation for Algorithms 17 and 18.

We can easily derive variants $\mathrm{Compression}_{\cdot}$, $\mathrm{Decompression}_{\cdot}$ for isogenies of any prime power degree (up to considering fields extensions of $\mathbb{F}_{p^2}$ if necessary). By concatenating the output of these variants of Compression and Decompression, we get an algorithm that works for any arbitrary degree. If the degree

Algorithm 17 Compression$_2$ $(E;\ )$

Input: A curve E and an isogeny $= _v \quad _1$ of degree 2^{fv} of domain E.
Output: A bit string of size $(f + 4)(v \quad 1) + f$ representing .

1: Compute a canonical basis of the torsion $E[2^f]$ and encode in S_1 an integer of f bits using DLP$_{2^f}$, the kernel of $_1$. This also determines Q_1 a point orthogonal to the kernel. $Q_2 \quad _1(Q_1)$
2: **for** $j \ 2\ [2; v]$ **do**
3: Write E_j for the codomain of $_{j \ 1}$, $k \quad 1$.
4: Deterministically generate a sequence $R_1; R_2; \quad 2\ E_j[2^f]$ until we reach a k for which R_k is orthogonal to Q_j. If $k < 2^4 \quad 1$, set s_j to be the binary representation of k. Else, set $s_j = 1111$.
5: Use DLP$_{2^f}$ to compute the f-bit integer S_j such that $\ker _j = \langle R_j + [S_j]Q_j \rangle$.
6: $Q_{j+1} \quad _j(Q_j)$.
7: **end for**
8: **return** $S = S_1 \| s_2 \| S_2 \| \ \| s_v \| S_v$.

Algorithm 18 Decompression$_2$ $(E; S)$

Input: The curve E, a bit string S of size $(f + 4)(v \quad 1) + f$ representing .
Output: An isogeny $= _v \quad _1$ of degree 2^{fg}.

1: Parse S as $S_1 \| s_2 \| S_2 \| \ \| s_v \| S_v$ where each S_j has f bits and s_j has 4 bits.
2: Compute canonically a basis of the torsion $E[2^f]$ and nd R_1 using S_1. De ne $_1$ as the isogeny of kernel R_1 and determine Q_1 a point orthogonal to the kernel.
3: $Q_2 \quad _1(Q_1)$.
4: **for** $j \ 2\ [2; v]$ **do**
5: Write E_j for the codomain of $_{j \ 1}$, $k \quad 1$.
6: If $s_j \neq 1111$, parse s_j as an integer k and recover R_j. Else, $k \quad 16$ and compute $R_{15}; R_{16}; \ldots$ until R_j is orthogonal to Q_j.
7: Parse S_j as an integer and compute $_j = \text{KernelToIsogeny}_{R_j + [S_j]Q_j}$.
8: $Q_{j+1} \quad _j(Q_j)$.
9: **end for**
10: **return** $= _v \quad _1$.

is smooth or powersmooth (with smoothness bound in $O(\text{poly}(\log(p)))$), then there is always a choice of exponent f such that the eld of de nition of the ℓ^f-torsion is small for every supersingular, and so we get an algorithm of complexity $O(\text{poly}(\log(p)))$.

4.2 The ideal representation

In this section, we introduce *the ideal representation*, an isogeny representation naturally derived from the Deuring correspondence. This representation appears to be much more suited to handle a generic isogeny than the canonical representation or the kernel representation.

The important algorithmic building blocks to handle the ideal representation were introduced in [KLPT14, GPS17, EHL+18]. Our main contribution in this

section are the practical algorithms to perform the translation from the ideal representation that we present in Section 4.2.2.

Remark 4.2.1. In this section and the next, we will build upon the heuristic algorithms from Chapter 3. As a result, all our results are only holding under heuristic assumptions. As mentioned in Chapter 3, Wesolowski [Wes22] introduced a version of the KLPT algorithm and managed to prove its expected polynomial termination assuming only the GRH. The tools developed for KLPT could be reused to derive provable versions of most of the other algorithms from Chapter 3. Using these variants instead, it should be possible to prove most of the results of this section and the next under the generalized Riemann hypothesis only.

Let us fix an element $x = (D; E_1; E_2) \in L_{\text{isog}}$ and assume that φ is a corresponding isogeny of degree D. We define *the ideal representation* for the isogeny φ as the associated kernel ideal I_φ (see Definition 2.1.1). With Lemma 2.2.5, we have shown that the ideal representation can be as compact as the kernel representation as we can represent any ideal of degree D with a basis whose coefficients over the canonical basis of $B_{p;1}$ have size $O(\log(pD))$.

In Section 4.2.1, we see how to perform the generic translation from the ideal representation to the canonical representation and the converse. In Section 4.2.2, we present fast algorithms targeting the special case of ideal to isogeny translation when the degree is in ℓ^e for some small prime ℓ. In Section 4.2.3, we explain how to perform the verification of an ideal representation. Finally, in Section 4.2.4, we explain how to use the ideal representation to evaluate any isogeny in polynomial time over the field of definition of the point to be evaluated.

4.2.1 Ideal to isogeny translation, the generic case

We introduce two generic algorithms to make the translation for an isogeny/ideal of degree/norm ℓ^e. As for the kernel representation, the algorithms we present work over $\mathbb{F}_q$ the field of definition of the ℓ^e-torsion. Galbraith, Petit and Silva in [GPS17] introduced the first explicit formulation of these algorithms. They are quite naturally derived from Definition 2.1.1 and Definition 2.1.3 of kernel ideals and kernel of an ideal. In fact, these algorithms work by computing the kernel of φ first, so we give a detailed description of the two sub-algorithms IdealToKernel and KernelToIdeal from which we can derive IdealToIsogeny and IsogenyToIdeal by composition with KernelToIsogeny and IsogenyToKernel. We target an arbitrary degree D.

In both IdealToKernel and KernelToIdeal, we assume that an explicit basis $\theta_1, \theta_2, \theta_3, \theta_4$ is known for $\mathrm{End}(E_1)$ where each θ_i can be evaluated on any point P of $E(\mathbb{F}_q)$ in $O(\log(p\ell^e))$ operations over $\mathbb{F}_q$. We also assume that we have a way to compute a concrete isomorphism between $\mathrm{End}(E_1)$ and any isomorphic maximal order. We will come back to these assumptions a bit later.

Lemma 4.2.2. *The algorithms* $\mathrm{KernelToIdeal}_D$ *and* $\mathrm{IdealToKernel}_D$ *are correct and terminate in* $O(\sqrt{p\overline{D}})$ *operations over the field of definition of* $E_1[D]$.

Proof. By our assumption on the endomorphism ring of E_1, the bottleneck operation in both algorithms is the discrete logarithm computations, which have $O(\sqrt{p\overline{D}})$ complexity.

Algorithm 19 IdealToKernel$_D$

Input: A curve E_1, a cyclic ideal of maximal orders $I \subset B_{p,1}$.
Output: A generator P_I of $E_1[I]$

1: Compute $D = n(I)$ and $O_1 = O_L(I)$.
2: Compute an explicit isomorphism $\theta : \mathrm{End}(E_1) \to O_1$.
3: Compute a generator $\gamma \in I$ such that $I = O_1\langle \gamma, D\rangle$.
4: Compute a basis P, Q of $E_1[D]$.
5: Compute $R, S = \theta^{-1}(\gamma)(P, Q)$.
6: **if** The order of $R < D$ **then**
7: Swap P with Q and R with S.
8: **end if**
9: Compute $a = \mathrm{DLP}_D(R, S)$.
10: **return** $P - [a]Q$.

Algorithm 20 KernelToIdeal$_D$

Input: A point P of order D in $E_1[D]$.
Output: The ideal $I(\langle P \rangle)$.

1: Compute $\theta : \mathrm{End}(E_1) \to B_{p,1}$ and set $O_1 = \theta(\mathrm{End}(E_1))$.
2: Compute a basis $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ of $\mathrm{End}(E_1)$ where each α_i has its norm coprime to D.
3: Find i, j such that $\alpha_i(P), \alpha_j(P)$ is a basis of $E_1[D]$.
4: Take $k \neq i, j$ and compute $a, b = \mathrm{BiDLP}_D(\alpha_k(P), \alpha_i(P), \alpha_j(P))$.
5: Compute $\gamma = (\alpha_k - a\alpha_i - b\alpha_j)$.
6: **return** $O_1\langle \gamma, D\rangle$.

For the correctness of IdealToKernel$_D$, since γ is a generator of I, we have that $\theta^{-1}(E_1[D])$ is a cyclic subgroup of order D of E_1. Since P, Q is a basis of $E_1[D]$, at least one of them is a generator of this cyclic subgroup, and so we ensure that there is a solution to the DLP. By definition of the kernel of an ideal, we get that $P - [a]Q$ is a generator of this group.

For the correctness of KernelToIdeal$_D$, we have that $\mathrm{End}(E_1) \otimes \mathbb{Z}_\ell$ is the endomorphism ring of the Tate module $T_\ell(E_1)$ and is isomorphic to $\mathrm{M}_2(\mathbb{Z}_\ell)$ for any prime $\ell \neq p$. So we get that the action of $\mathrm{End}(E_1)$ on $E_1[D] = \mathbb{Z}/D\mathbb{Z}^2$ is the same as $\mathrm{M}_2(\mathbb{Z}/D\mathbb{Z})$. Thus, we must have two endomorphisms α_i, α_j in the basis of $\mathrm{End}(E_1)$ sending P to points of order D that are not in the same subgroup of order D, which implies that they form a basis of $E_1[D]$. This is why the bi-dimensional DLP will be correct and find two coefficients a, b such that $\alpha_k(P) = a\alpha_i(P) + b\alpha_j(P)$. This proves that $\gamma = (\alpha_k - a\alpha_i - b\alpha_j)$ will be in $I(\langle P \rangle)$ from Definition 2.1.1. To see that γ must generate this ideal, it suffices to argue that $\langle P \rangle = \ker(\theta) \cap E_1[D]$. Since we have already exhibited that $\langle P \rangle \subset \ker(\theta)$, if there is another point $Q \in \ker(\theta) \cap E_1[D]$, we must have that there exists $d \mid D$ such that $E_1[d] \subset \ker(\theta)$ and this is not possible for a combination of a basis of $\mathrm{End}(E_1)$ whose coefficients are not all zero in $\mathbb{Z}/d\mathbb{Z}$ (otherwise the basis could not generate any matrix in $\mathrm{M}_2(\mathbb{Z}/d\mathbb{Z})$). This proves that the output is correct. □

Remark 4.2.3. Seeing Lemma 4.2.2, one might think that contrary to what we announced, the ideal representation is not better than the kernel representation

since the complexity remains exponential in D. In fact, the inefficiency of KernelToIdeal and IdealToKernel stems directly from the fact that we want to translate to or from the kernel representation. In Section 4.2.3, we will show that we can do a lot better because we avoid entirely to compute the kernel of φ.

4.2.2 Ideal to isogeny, efficient algorithms for the prime power case

This section is dedicated to devise an efficient method to perform the ideal to isogeny translation when the degree is ℓ^e. We are going to present two different algorithms to perform that task. The first one, IdealToIsogenyFromKLPT, was first introduced in essence by Eisentraeger et al. in [EHL+18] as an efficient generalization of the IdealToIsogeny algorithm from Section 4.2.1. The version we present is enhanced with several tricks to improve the efficiency and was presented in our article [DFKL+20]. The second one, IdealToIsogenyFromEichler, works slightly differently. It was a contribution of our paper [DFLW22] and was introduced as an improvement of IdealToIsogenyFromKLPT. The two approaches have their interest and this is why we present them both, but the second one appears to be faster in practice, which we will justify informally in this section, and later experimentally in Chapter 5, where we provide a detailed cost analysis between the two methods and concrete efficiency in the context of the signature scheme presented there.

The IdealToIsogeny algorithm from Section 4.2.1 is not efficient when the degree is ℓ^e for some large exponent e because the ℓ^e-torsion might be defined on an extension of $\mathbb{F}_{p^2}$ of very large degree. To avoid the expensive operations over huge field extensions, we will divide the computation into steps of size ℓ^f where f is the biggest exponent such that the ℓ^f-torsion is defined over $\mathbb{F}_{p^2}$ (we assume $f > 0$) so we only need to perform operations in $\mathbb{F}_{p^2}$.

In Part II, we are going to take $\ell = 2$ but we state our algorithms in full generality. Nonetheless, it might be worth remembering that we target values of ℓ p for which a $O(\text{poly}(\ell))$ complexity is considered acceptable. Let I be the ideal of norm ℓ^e that we want to translate. The corresponding isogeny φ_I admits the decomposition $\varphi_g \circ \cdots \circ \varphi_1$ under the factorization $I = I_1 I_2 \cdots I_g$, where each I_i of norm ℓ^f corresponds to φ_i under the Deuring correspondence.

Given what we have said so far, the task might seem easy at first glance. If we iterate several times IdealToIsogeny to translate each of the I_i, then with the ℓ^f-torsion defined over $\mathbb{F}_{p^2}$ and a $\tilde{O}(f\sqrt{p})$ complexity algorithm, it seems that we have everything we need already. And this is true except for a small caveat: for IdealToKernel, we made the seemingly reasonable assumption that we have a way to evaluate efficiently the endomorphisms of the domain. In reality, evaluating endomorphism of a random supersingular curve is not something easy, and the two methods we present below are heavily dependent on the way we choose to do this task.

On representing and evaluating endomorphism rings. In Chapter 2, we introduced the Deuring correspondence and we gave as a concrete example the curve E_0 of j-invariant 1728, which is supersingular and has endomorphism ring isomorphic to the maximal order $\mathcal{O}_0 = \langle 1, i, \frac{i+j}{2}, \frac{1+k}{2} \rangle$ in $B_{p,\infty}$ when $p \equiv 3 \bmod 4$. We also gave concrete formulas for the endomorphisms ι, $2 \in \text{End}(E_0)$

corresponding to $i; j$ under the isomorphism $_0 : O_0 \; ! \; \mathrm{End}(E_0)$. From there, we can derive explicitly the image of any endomorphism under $_0$ by decomposing it rst on the basis of O_0 before using the addition formulas, the scalar multiplications and $\; ;$.

There is only a slight issue in what we described brie y: how do we translate the division by two through our isomorphism? Scalar multiplication is a well-de ned operation over any elliptic curve, but there is no division operation. If we say that the division by two is the inverse of the multiplication by two, we can easily evaluate it on any point of order N coprime to 2 by replacing the division by 2 with the scalar multiplication by the inverse of 2 mod N. Treating the points of even order is more complicated. Since [2] has a kernel of size 4, there are always four preimage under [2] of any point of even order, and they can be found by computing the roots of the second division polynomial. Fortunately, we don't need to lift the ambiguity to evaluate endomorphisms. If we want to evaluate the endomorphism $=2$ on the point P when we know how to evaluate the endomorphism $\;$ on any point, it su ces to have any point Q such that $P = [2]Q$. Then $(=2)(P) = \;(Q)$. Since the second division polynomial has small degree, nding the roots can be done easily and the point Q will be de ned on a quadratic extension in the worst case.

This nishes explaining how we can evaluate any endomorphism of $\mathrm{End}(E_0)$ quite e ciently. We can nd an analog of the curve E_0 for any prime p. Interestingly enough, as we can conjecture from our example, all the instances of such supersingular curves have an endomorphism ring that is isomorphic to one of the special extremal orders from Chapter 3. So let us assume for the rest of this paragraph that we take the curve E_0 to be the special extremal curve in $\mathbb{F}_{p^2}$. The di culty we encountered with the division by 2 forebodes the complications we will encounter for a generic supersingular curve E_1 where we don't have nice endomorphisms with simple expressions. A solution was outlined in [EHL+18] using an isogeny $' : E_0 \; ! \; E_1$ of degree N. Let us write $I_{'}$, the corresponding ideal and $O_1 = O_R(I) = \mathrm{End}(E_1)$. The idea is to use the fact that $NO_1 \quad O_0$. This containment corresponds to the embedding of $_0 = \hat{} \quad '$ in $\mathrm{End}(E_0)$ for any $\; 2 \; \mathrm{End}(E_1)$ (we have already used this embedding in Section 2.3.2). Using the reverse embedding of $NO_0 \; , ! \; O_1$, we get $[N^2] \; = \; ' \quad _0 \quad \hat{}$ and so

$$= \frac{' \quad _0 \quad \hat{}}{N^2}: \tag{4.2.1}$$

Thus, the idea is to evaluate $\;$ by rst evaluating $_0$ with the explicit isomorphism $_0$, translate this to E_1 using $'$ and $\hat{}$ and then apply the division by N^2.

The method we just outlined will successfully allow us to evaluate any endomorphism in any supersingular curve (when we know its endomorphism ring). However, if we want something that is e cient, there is still some work to do. The main operations are translation from $I_{'}$ to $'$, evaluations of $'$ and $\hat{}$, and the division by N^2. This implies two things: the degree N needs to enable fast N-isogenies evaluations and translation, and we need e cient division by N. These two constraints imply that we need to be very careful in the choice of N. More precisely, for e cient isogeny evaluation and translation, the best choice is to have N smooth as we explained in Section 4.1. For the division, it is important that N is either small (so that we can easily compute a preimage under the scalar multiplication by N) or that it is coprime to the order of

the point we need to evaluate. Apart from special cases, we cannot hope for a small N and so we will always target the second possibility. Without further assumptions, when O_1 is a random maximal order, we cannot say anything on the norm of $I(O_0; O_1)$. This is where the algorithms from Chapter 3 come into play. Using KLPT, we can find a $J \sim I$ with some control over its norm. This means that we will be able to compute $O_1' = \mathrm{End}(E_1)$ where $n(I(O_0; O_1'))$ will allow for efficient evaluation of the endomorphisms on any points of E_1 (under the constraint we stated above). This is exactly the idea that we apply to get IdealToIsogenyFromKLPT. The IdealToIsogenyFromEichler algorithm will follow from slightly different ideas that we explain a bit later.

The discussion above motivates the introduction of a smooth integer T coprime to ℓ such that T-isogeny computations are efficient over the supersingular curves in $\mathbb{F}_{p^2}$ and $T > p^3$ so that $\mathrm{KLPT}_{D(T)}$ will succeed with overwhelming probability (we recall that $\mathrm{KLPT}_{D(T)}$ will look for ideals whose norm divides T). If we want a generic T that will work for all prime p, we can always take it to be powersmooth. For specific values of p, it might be possible to get a better choice by carefully looking at the divisors of $\#E(\mathbb{F}_q)$ for q that are small powers of p (in practice we will take $q = p^2$). This is what we will do in the concrete instantiations in Part II. The difficulty in finding a good T comes from the bound in Proposition 3.2.7 where we require $T > p$. We will come back later in Part II on the problem of finding good p and T.

Putting everything we have said so far together, we see that IdealToIsogenyFromKLPT will be made of consecutive executions of two main building blocks: IdealToIsogenyCoprime to translate and evaluate T-isogenies and IdealToIsogenySmallFromKLPT to translate ℓ^f isogenies using the endomorphism ring representation given by the output of IdealToIsogenyCoprime. Below, we discuss several tricks that will make our final algorithms more efficient.

Computing half of the isogeny from the image curve. One of the main limitations of the algorithm outlined above is the fact that we restrict ourselves to degrees that are defined on small $\mathbb{F}_{p^2}$ extensions, which is a strong constraint on the size of these degrees. There is a way to double the size of the degrees we can handle without changing the torsion requirement. Let us assume here that we want to perform the translation of an isogeny $\varphi : E_1 \to E_2$ of norm D^2. Instead of applying IdealToIsogeny on E_1 that would require to use the D^2-torsion, we can use the fact that $\varphi := \varphi_2 \circ \varphi_1$, each of degree D, to apply IdealToIsogeny once on E_1 and once on E_2 to compute φ_1 and $\hat{\varphi}_2$ respectively using only the D-torsion. We apply this idea in IdealToIsogenyCoprime to translate an ideal of norm dividing T^2 (instead of T previously) to the corresponding isogeny. This means we now only need $T \sim p^{\frac{3}{2}}$ instead of $T \sim p^3$ and this will greatly help for a choice of concrete parameters in our applications. We will also use it in IdealToIsogenySmallFromKLPT to make steps of size ℓ^{2f} instead of ℓ^f.

Meet-in-the-middle. If we push further the idea from the previous paragraph, we can do a little better. Let us now assume that we want to translate an isogeny of degree $D^2 e$. We can decompose it as $\varphi = \varphi_2 \circ \psi \circ \varphi_1$ where φ_i have degree D and ψ have degree e. We can compute $\varphi_1, \hat{\varphi}_2$ from the D-torsion as before. Let us write E_3, E_4 the codomains of these curves so that $\psi : E_3 \to E_4$. If ψ is small enough and smooth enough, it is conceivable that we can simply

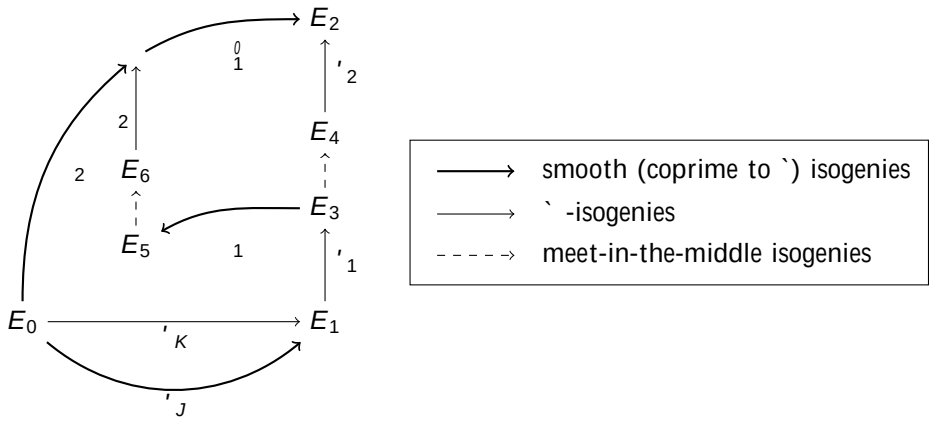


Figure 4.1: Graphical representation of the ideal to isogeny translation of IdealToIsogenySmallFromKLPT

compute in $O(\overline{p}, \overline{e})$ by performing a meet-in-the-middle of degree e between $E_3; E_4$ (i.e., perform brute force through all the isogenies of degree e_1 starting from E_3 and e_2 starting from E_4 with $e_1 e_2 = e$ until we find a collision). We will apply this idea by taking $e = \ell$ in IdealToIsogenySmallFromKLPT to make steps of size ℓ^{2f+} instead of ℓ^{2f}. In our applications, the size of will not be negligible before the size of f, which is why this trick is useful. We do not apply it in IdealToIsogenyCoprime because, as T is much bigger than ℓ, it will not change much.

Our two ideas put together are illustrated in Figure 4.1. The notations from Figure 4.1 are consistent with the ones in IdealToIsogenyCoprime and IdealToIsogenySmallFromKLPT;

Ideal to isogeny from KLPT. We are now ready to give a detailed description of our main algorithm IdealToIsogenyFromKLPT in Algorithm 23. As we explained, our method is inspired by the algorithms from [EHL+18] and this is why we give only sketches of proofs for the next propositions.

IdealToIsogenyFromKLPT translates an O-ideal of norm ℓ^e in the corresponding isogeny for any maximal order O. It requires K a left O_0-ideal and right O-ideal of degree in ℓ along with the corresponding isogeny $\varphi_K : E_0 \to E$ where $O = \mathrm{End}(E)$. The value ℓ is an implicit parameter, as are the exponents f and (for the meet-in-the-middle) and the integer T.

As explained, IdealToIsogenyFromKLPT uses the following subroutines:

- IdealToIsogenyCoprime$(J; K; \varphi_K)$: described in Algorithm 21, it takes $J; K$ two left O_0-ideals of norm $n(K)$ 2ℓ and $n(J)$ dividing T^2 along with the isogeny $\varphi_K : E_0 \to E$ and outputs φ_J.
- IdealToIsogenySmallFromKLPT$(I; J; K; \varphi_J; \varphi_K)$: described in Algorithm 22, it takes I a left O_0-ideal of norm dividing $T^{2\ell^{2f+}}$, J containing I of norm dividing T^2 and K J of norm ℓ along with $\varphi_J; \varphi_K$ and outputs φ of degree ℓ^{2f+} such that $\varphi_I = \varphi \circ \varphi_J$.

Algorithm 21 IdealToIsogenyCoprime$_T(J; K; \theta_K)$

Input: Two equivalent left ideals $J; K$ of O_0, with J of norm dividing T^2 and K of norm ℓ, and the corresponding isogeny $\theta_K : E_0 \to E$.
Output: θ_J.

1: $H_1 \leftarrow J + TO_0$.
2: Let $\alpha \in K$ such that $J = \alpha_K(\ )$.
3: $H_2 \leftarrow \langle \ ; (n(J)/n(H_1))\rangle$.
4: $\theta_{H_i} \leftarrow$ IdealToIsogeny$_{n(H_i)}(H_i) : E_0 \to E_i$.
5: Let $\varphi : E \to E/\theta_K(\ker \theta_{H_2}) = E_1$.
6: **return** $\widehat{\varphi} \circ \theta_{H_1}$.

Proposition 4.2.4. *The algorithm* IdealToIsogenyCoprime *is correct and terminates in* $O(\mathrm{poly}(\log(pn(K)T))\sqrt{p}\overline{B})$*, where* B *is the maximum of* ℓ *and the smoothness bound of* T*.*

Proof. (sketch) The coefficients of a basis of O_0 are in $O(\mathrm{poly}(p))$. Thus, the coefficients of a basis of $J; K$ are in $O(\mathrm{poly}(pTn(K)))$, and so every operation over $B_{p;1}$ can be carried out in $O(\mathrm{poly}(\log(pTn(K))))$. The T-isogeny translation and computations takes $O(\log(t)\sqrt{p}\overline{B})$-operations over $\mathbb{F}_{p^2}$ by Lemma 4.2.2 and the results from Section 4.1. IdealToIsogeny can be applied because we perform the translation for O_0-ideals and the curve E_0 is one of the special cases where we have a concrete isomorphism between $\mathrm{End}(E_0)$ and O_0. The evaluation through θ_K can be done in $O(\sqrt{p}\overline{B}\log(n(K))$ and since ℓ and T are coprime, is well-defined as the push-forward through θ_K of θ_{H_2}.

□

Algorithm 22 IdealToIsogenySmallFromKLPT$(I; J; K; \theta_J; \theta_K)$

Input: I a left O_0-ideal of norm dividing $T^{2\ell^{2f+}}$, an O_0-ideal in J containing I of norm dividing T^2, and an ideal $K \subset J$ of norm a power of ℓ, as well as θ_J and θ_K.
Input: $\theta = \theta_2 \circ \theta_1 : E_1 \to E_2$ of degree ℓ^{2f+} such that $\theta_I = \theta \circ \theta_J$, $L \subset I$ of norm dividing T^2 and θ_L.

0: Write $\theta_J; \theta_K : E_0 \to E_1$.
1: Let $I_1 = I + \ell^f O_0$.
2: Let $\theta'_1 =$ IdealToIsogeny$_{\ell^f}(I_1)$.
3: Let $\theta_1 = [\theta_J] \circ \theta'_1 : E_1 \to E_3$.
4: Let $L =$ KLPT$_{D(T^2)}(I)$.
5: Let $\alpha \in K$ such that $J = \alpha_K(\ )$.
6: Let $\beta \in I$ such that $L = \beta_I(\ )$.
7: Let $\gamma = \beta\alpha/n(J)$. We have $\gamma \in K$, $\gamma \in L$, and $n(\gamma) = T^{2\ell^{2f+}} n(K)$.
8: Let $H_1 = \langle \ ; n(K)\ell^f T\rangle$. We have $\theta_{H_1} = \varphi_1 \circ \theta_1 \circ \theta_K : E_0 \to E_5$, where φ_1 has degree T.
9: Let $H_2 = \langle \overline{\gamma}; \ell^f T\rangle$. We have $\theta_{H_2} = \varphi_2 \circ \varphi_2 : E_0 \to E_6$, where φ_2 has degree T and θ_2 has degree ℓ^f.
10: Find $\varphi : E_5 \to E_6$ of degree ℓ with meet-in-the-middle.
11: Let $\theta_2 = [\widehat{\varphi}_1] \circ \widehat{\varphi}_2 : E_3 \to E_2$ and $\varphi'_1 = [\widehat{\varphi}_2] \circ \widehat{\varphi}_1$.
12: **return** $\theta = \theta_2 \circ \theta_1$, L and $\varphi'_1 \circ \varphi_2$.

Proposition 4.2.5. *(UPHA) For any* > 0*, there exists* $= O(\log\log(p) + \log(\ ))$ *such that if* T *is a* B*-smooth integer with* $\log(T) > 3/2\log(p) +$ *and* $\ell < B < p^{\overline{p}} +$ *, then* IdealToIsogenySmallFromKLPT *will succeed with probability bigger than* $1 \quad 2^{\ }$ *when* I *is in a random ideal class. The expected running time is* $O(\text{poly}(\log(pTn(K)f))\ p^{\overline{B}}\ )$

Proof. (sketch) For the same reason as in Proposition 4.2.4 the ideals have coefficients of size in $O(\text{poly}(pTn(K)))$ and so all the operations over $B_{p;1}$ can be performed in $O(\text{poly}(\log(pTn(K))))$. The meet-in-the-middle takes $O(p^{\overline{B}}\ )$ operations over small field extensions of $\log(p)$. The T-isogeny computations take $O(p^{\overline{B}}\log(T))$ and ℓ^f-isogenies translation and computations take $O(p^{\overline{B}}\log(T)f)$ by Lemma 4.2.2. The proof of the expected running time is completed by Proposition 3.2.7.

We refer to Figure 4.1 to verify the correctness of the various isogeny computations that we do. If $T > p^{3/2} +$ and is B-smooth with $B < p^{1/2} +$ (where the value depends on $_1; _2$ in Proposition 3.2.7) we will be able to write T^2 as T_1T_2 where $T_1; T_2$ respects the constraint of Proposition 3.2.7 and the probability of success follows from Proposition 3.2.7. □

Algorithm 23 IdealToIsogenyFromKLPT$_{\ell}$ $(I; K; \ '_K)$

Input: A left O-ideal I of norm a power of ℓ, K a left O_0-ideal and right O-ideal of norm ℓ , the corresponding $\ '_K$.
Output: $\ '_I$.

1: Write $I = I_v \quad I_1 \quad I_0 = O$ where $n(I_i)/n(I_{i\ 1}) \quad \ell^{2f+}$.
2: $J \quad \text{KLPT}_{T^2}(K)$.
3: $\ '_J \quad \text{IdealToIsogenyCoprime}(J; K; \ '_K)$.
4: **for** $i = 1; : : : ; v$ **do**
5: $\quad '_i; J; \ '_J \quad \text{IdealToIsogenySmallFromKLPT}(J \quad I_i; J; K; \ '_J; \ '_K)$.
6: $\quad K \quad K \quad I_i$.
7: $\quad '_K \quad '_i \quad '_K$.
8: **end for**
9: **return** $\ '_n \quad '_1$.

Proposition 4.2.6. *(UPHA) For any* > 0*, there exists* $= O(\log\log(pn(I)) + \log(\ ))$ *such that if* T *is a* B*-smooth integer with* $\log(T) > 3/2\log(p) +$ *and* $\ell < B < p^{\overline{p}} +$ *, then* IdealToIsogenyFromKLPT *will succeed with probability bigger than* $1 \quad 2^{\ }$ *when* I *is in a random ideal class. The expected running time is* $O(\text{poly}(\log(pTn(K)n(I)f))\ p^{\overline{B}}\ )$

Proof. (sketch) The result follows from Proposition 4.2.4 and Proposition 4.2.5. □

2.2.1 Ideal to isogeny from Eichler orders. Next, we introduce IdealToIsogenyFromEichler, an improved variant of IdealToIsogenyFromKLPT. The main change is in the sub-routine IdealToIsogenySmallFromKLPT that we replace by IdealToIsogenySmallFromEichler. The idea behind this second version is the following: the algorithm IdealToKernel from Section 4.2.1 only needs to evaluate one endomorphism of E_1, but we use a method that gives a way to evaluate any

endomorphism of E_1. Intuitively, this means that our algorithm is not optimal because we compute something more powerful than what we need. In IdealToIsogenySmallFromEichler, we thus strive to be more e cient by computing exactly one endomorphism of E_1 and nothing else. Note that this is not the endomorphism used in IdealToKernel. For IdealToIsogenySmallFromEichler, we will need that P; (P) is a basis of $E_1[\ell^f]$ for some P of order ℓ^f given in input. We consider that this generator is an input of IdealToIsogenyFromEichler, together with the isogeny whose kernel is $\langle P \rangle$ and the corresponding ideal. The exact constraint for is given in Lemma 4.2.7. With this, it will become clear that we can compute with SpecialEichlerNorm and this is the main reason why IdealToIsogenyFromEichler ends up being better than IdealToIsogenyFrom-KLPT because the size of the outputs of SpecialEichlerNorm is smaller than for KLPT. According to Proposition 3.3.3, we can nd an isogeny of degree $p^{5/2}$ (against p^3 for KLPT). Using the trick to divide an isogeny of degree T^2 in two isogenies of degree T, we will end up with T-isogeny computations where T $p^{5/4}$ (against $p^{3/2}$ before). Concretely, this means that we will decompose the endomorphism of norm dividing T^2 in two isogenies $\varphi_1; \varphi_2$ of degree $n_1; n_2$ dividing T such that $= \varphi_2 \hat{}_1$. The resulting sub-algorithm IdealToIsogenySmallFromEichler also ends up being slightly less convoluted than IdealToIsogenySmallFromKLPT, allowing for a more concise exposition, which is why we do not need the analog of IdealToIsogenyCoprime.

Algorithm 24 IdealToIsogenySmallFromEichler$_{\ell^f}(\mathcal{O}; I; J; \varphi_J; P)$

Input: I a left $\mathcal{O}$-ideal of norm ℓ^f, an $(\mathcal{O}_0; \mathcal{O})$-ideal J of norm in ℓ and φ_J : $E_0 \to E$ the corresponding isogeny, the generator $P \in E[\ell^f]$ of $\ker \varphi_K$ s.t $\hat{}_J = \varphi_{K^0} \varphi_K$.

Output: φ_I of degree ℓ^f

1: Set $K = \overline{J} + \mathcal{O}\ell^f$.
2: Compute = SpecialEichlerNorm$_T(\mathcal{O}; K + \mathcal{O}\ell)$ of norm dividing T^2.
3: Select $\in I$ s.t. $I = \mathcal{O}\langle ; \ell^f \rangle$.
4: Compute $C; D$ s.t. $(C + D) \in K$ and $\gcd(C; D; \ell) = 1$ using linear algebra.
5: Take any $n_1 \mid T$ and $n_2 \mid T$ s.t. $n_1 n_2 = n( )$. Compute $H_1 = \mathcal{O}\langle ; n_1 \rangle$ and $H_2 = \mathcal{O}\langle \bar{} ; n_2 \rangle$.
6: Compute $L_i = [J]$ H_i, and $\varphi_i = [\varphi_J]$ IdealToIsogeny$_{n_i}(L_i)$ for $i \in \{1; 2\}$.
7: Compute $Q = \hat{}_2$ $\varphi_1(P)$.
8: Compute φ_I of kernel $\langle [C]P + [D]Q \rangle$.
9: return φ_I.

Before stating that Algorithm 24 is correct and terminates, we need a preliminary lemma. For any $\mathcal{O}$-ideal K of degree ℓ^f we write $K_r = K + \ell^r \mathcal{O}$ for $1 \quad r \quad f$.

Lemma 4.2.7. *Let E be a supersingular curve and $\mathcal{O} = \mathrm{End}(E)$ be a maximal order. Let K and I be two $\mathcal{O}$-ideals of norm ℓ^f, let $\in \mathcal{O}$ r $(\mathbb{Z} + K)$ have norm coprime to ℓ. Let $E[K] = \langle P \rangle$, then $E[I] = \langle [C]P + [D]$ $(P) \rangle$ $\gcd(C; D; \ell) = 1$ and $(C + D) \in K$ for any s.t $I = \mathcal{O}\langle ; \ell^f \rangle$.*

Proof. Let us take $Q = [C]P + [D]$ (P) and assume that $E[I] = \langle Q \rangle$. Since Q has order ℓ^f, it is clear that $\gcd(C; D; \ell) = 1$. Let us take $\in I$ such

that $I = \langle \cdot ; \ell^f \rangle$. This condition is equivalent to $\ker \cap E[\ell^f] = E[I]$. We want to show that $(C + D) \in K$, i.e., that $(C + D)(P) = 0$ which is straightforward since $E[I] = \langle [C]P + [D](P) \rangle$. Conversely, let us assume that $\gcd(C, D, \ell) = 1$ and $(C + D) \in K$ for any s.t. $I = \langle \cdot ; \ell^f \rangle$. Taking such an , we get that $(C + D)(P) = 0$ which must imply that $[C]P + [D](P) = Q$ for some $\in \mathbb{Z}$ and Q such that $E[I] = \langle Q \rangle$. If we show that $\gcd(; \ell^f) = 1$ then we will have shown our result as P and (P) have order ℓ^f. Let us assume this is not the case. We have $\gcd(; \ell^f) = \ell^{e_0}$ for $e_0 > 0$. Then the point $P_0 = [\ell^{f-e_0}]$ of order ℓ^{e_0} satisfies $[D](P_0) = [C]P_0$. Since $\gcd(C, D, \ell) = 1$, we must have $\gcd(D, \ell) = 1$ and so $(P_0) = [ ]P_0$ where $= C/D \bmod \ell^{e_0}$. This proves that we have $\in \mathbb{Z} + K_{e_0} \subset \mathbb{Z} + K_1$, which contradicts our initial assumption. Hence, $\gcd(; \ell^f) = 1$ and we have proven the result. □

Proposition 4.2.8. *(UPHA)* IdealToIsogenySmallFromEichler *is correct. For any* > 0*, there exists* $= O(\log\log(p) + \log( ))$ *such that if* T *is a* B*-smooth integer (with* $\ell < B$*) with* $\log(T) > 5/4 \log(p) + $ *, then* IdealToIsogenySmallFromEichler *will succeed with probability bigger than* $1 - 2$ *when* O *is random maximal order. The expected running time is* $O(\text{poly}(\log(pTn(J)f)) \cdot \overline{B})$.

Proof. Correctness follows from Lemma 4.2.7. When we assimilate the endomorphisms and $[C] + [D]$ in $\text{End}(E)$ with their image through the isomorphism between $\text{End}(E)$ and O, we get that the composition $(C + D)$ becomes the multiplication of the quaternion elements $(C + D)$. Thus, by Lemma 4.2.7, the values C, D computed at Step 4 are such that $\ker \varphi_I = \langle [C] + [D](P) \rangle$. By definition of H_1, H_2, we have that $= \hat{}_2 \varphi_1$ and this concludes the proof that the output isogeny is indeed the one corresponding to I through the Deuring Correspondence.

Apart from the execution of SpecialEichlerNorm, the only step that needs justification is Step 4. First, it is not clear that such a solution must always exist. In fact, the existence of such C, D follows from $\in \mathbb{Z} + (K + \ell O)$. This condition implies that $P, (P)$ form a basis of $E[\ell^f]$, for otherwise we would have $[\ell^{f-1}]P = [\ell^{f-1}](P)$ and so $\in \mathbb{Z} + (K + \ell O)$, since $E[K] = \langle P \rangle$. When it exists, a solution C, D can easily be found using linear algebra, similarly to IdealModConstraint.

The condition on the size of T is compatible with the constraints in Proposition 3.3.3, and so the final result follows from Proposition 3.3.3. □

Now we are ready for our full algorithm, which is basically made of sequential executions of IdealToIsogenySmallFromEichler$_f$. For simplicity, we assume in Algorithm 25 that the ideal input to IdealToIsogenyFromEichler$_\ell$ has norm ℓ^e, where $e = fg$ for some $g \in \mathbb{N}$. We can derive easily the general case from there.

Proposition 4.2.9. *(UPHA)* IdealToIsogenyFromEichler *is correct. For any* > 0*, there exists* $= O(\log\log(pn(I)) + \log( ))$ *such that if* T *is a* B*-smooth integer (with* $\ell < B$*) with* $\log(T) > 5/4 \log(p) + $ *, then* IdealToIsogenyFromEichler *will succeed with probability bigger than* $1 - 2$ *when* I *is in a random ideal class. The expected running time is* $O(\text{poly}(\log(pTn(I)n(J)f)) \cdot \overline{B})$.

Proof. It is easily verified that the $O_i, I_i, J_i, \varphi_J \varphi_I, P_i$ are correct inputs to IdealToIsogenySmallFromEichler. If I is well-distributed, we can assume that the

Algorithm 25 IdealToIsogenyFromEichler$_{\ell,f}(I, J, \varphi_J)$

Input: I a left $\mathcal{O}$-ideal of norm ℓ^e with $e = fv$, an $(\mathcal{O}_0, \mathcal{O})$-ideal J of norm ℓ and $\varphi_J : E_0 \to E$ the corresponding isogeny
Output: φ_I of degree ℓ^e.

1: Set $J_i = J$, $I_i = I + \ell^f\mathcal{O}$, $I_i' = I_i^{-1}I$, $\mathcal{O}_i = \mathcal{O}$.
2: Set φ_i of degree ℓ^f as the isogeny such that $\widehat{\varphi_J} = \varphi' \circ \varphi_i$
3: Set $\varphi_I = [1]_E$ and $E_i = E$.
4: **for** $i \in [1, v]$ **do**
5: Compute $P_i \in E_i[\ell^f]$ s.t. $\ker \varphi_i = \langle P_i \rangle$.
6: Compute $\varphi_{I_i} = \text{IdealToIsogenyFromEichler}_{\ell,f}(\mathcal{O}_i, I_i, J_i, \varphi_J \circ \varphi_I, P_i)$.
7: Set $\varphi_i = \widehat{\varphi_{I_i}}$, $\varphi_I = \varphi_{I_i} \circ \varphi_I$ and E_i is the codomain of φ_{I_i}.
8: Set $J_i = J_i \cdot I_i$, $\mathcal{O}_i = \mathcal{O}_L(I_i')$, $I_i = I_i' + \ell^f\mathcal{O}_i$ and $I_i' = I_i^{-1}I_i'$.
9: **end for**
10: **return** φ_I.

I_i are in well-distributed ideal classes. Thus, the result follows from Proposition 4.2.8 □

Remark 4.2.10. Throughout this section, we have given tight bounds on the size of T. For concrete implementation, it will be important to take T as small as we can. However, in the remainder of this chapter, we will use IdealToIsogenyFromEichler in various algorithms for which we do not care about concrete efficiency, but we want the algorithm to succeed with overwhelming probability in polynomial time. For instance, it is easy to see from Proposition 4.2.9 that we can take $\ell^f = O(\log(p))$ and T to be B-smooth with $B = O(\log(p))$ such that IdealToIsogenyFromEichler will succeed with overwhelming probability. This is what we will implicitly do for the rest of this chapter.

Below, we explain more precisely how to perform Step 7 of IdealToIsogenySmallFromEichler. We left the technical details out of the description in Algorithm 24 for clarity, but they are important for an efficient implementation. For now, we have avoided the issues of potential failures of SpecialEichlerNorm that were mentioned in Section 3.5 by assuming that the ideal I is in a random class. We will discuss a bit later how to perform the computation in the event of a bad distribution.

Endomorphism evaluation. For Step 7 of IdealToIsogenyFromEichler$_{\ell,f}$, it is required to evaluate the endomorphisms $\theta = \widehat{\varphi_2} \circ \varphi_1$ after the computation of the two isogenies φ_1, φ_2. One might assume that it suffices to push P through φ_1 and then do the same through $\widehat{\varphi_2}$. This apparently simple algorithm is not so easy to implement concretely. The first problem lies with signs. Most of the efficient isogeny algorithm are using x-only (or X, Z) arithmetic, which implies that we can only evaluate isogenies up to signs. This is problematic as the ultimate goal is to compute $[C]P + [D]\,\theta(P)$. Solving this issue requires evaluating several other points through φ_1, φ_2, and there does not seem to be another easy way to remove the ambiguity. The second issue is with the dual computation in itself. For an isogeny φ of degree T and kernel $\langle P \rangle$, computing $\widehat{\varphi}(R)$ for some point R would first require to compute $\varphi(Q)$, where Q is of order

T and orthogonal to P to get $\ker \hat{}$, before using this kernel to compute $\hat{}(R)$ and this is expensive (see Section 5.4.1 for a more detailed cost analysis).

Targeting the application to IdealToIsogenySmallFromEichler, we present in Algorithm 26, a method that requires only two T-isogeny computations and 5 evaluations together with a few discrete logarithms to evaluate an endomorphism of the form $C + D$ where $= \hat{}_2 \ '_1$ on a point of power-smooth order (this performs both Step 7 and Step 8 of Algorithm 24 at once). Our method is entirely based on x-only arithmetic and requires a basis $P; Q$ of the $`^f$-torsion. The main principle is to express $'_1(P)$ as a linear combination of $'_2(P); '_2(Q)$ and see that $\hat{}_2 \ '_1(P)$ is a multiple of the linear combination of $P; Q$ with the same coe cients. When dealing with x-only arithmetic, we need also to compute $'_2(P + Q)$ to perform the discrete log computations. Finally, to lift the ambiguity (the linear combination that we obtain is only up to sign) we use the trace of $= \hat{}_2 \ '_1$ (which can be computed by expressing in the basis $h1; i; j; ki$). In the basis $P; Q$, the action of can be seen as a matrix of $M_2(Z=`^f Z)$. This matrix is essentially the one we obtain with the coe cient of the two discrete logarithms, and so it su ces to check the value of the trace to lift any sign ambiguity.

We recall that we have the function $\text{xBiDLP}_{`^f}(x(R); x(P); x(Q); x(P + Q))$ that computes the scalars $a; b$ such that $x(R) = x([a]P + [b]Q)$ for points $P; Q$ of order $`^f$. It has complexity $O(f \sqrt{p})$.

Algorithm 26 $\text{EndomorphismEvaluation}_{`^f}('_1; '_2; C; D; t; P)$

Input: Two isogenies $'_1; '_2 : E \to E'$, scalars $C; D$, the trace $t = \text{tr}(\hat{}_2 \ '_1)$ and a point P of order $`^f$

Output: $[C]P + [D]\hat{}_2 \ '_1(P)$

1: Compute Q such that $P; Q$ is a basis of $E[`^f]$ and compute $P + Q$.
2: Compute $x('_1(P)); x('_1(Q)); x('_2(P)); x('_2(Q)); x('_2(P + Q))$.
3: Compute $x_1; x_2 = \text{xBiDLP}_{`^f}(x('_1(P)); x('_2(P)); x('_2(Q)); x('_2(P + Q))$ and $x_3; x_4 = \text{xBiDLP}_{`^f}(x('_1(Q)); x('_2(P)); x('_2(Q)); x('_2(P + Q)))$.
4: Change the signs of $(x_1; x_2); (x_3; x_4)$ until $(x_1 + x_4) \deg '_2 = t \mod `^f$.
5: Set $a = C + x_1 D$ and $b = x_2 D$.
6: Compute $R = [a]P + [b]Q$.
7: **return** R.

Remark 4.2.11. If we want to compute a compressed representation of the output of IdealToIsogenyFromEichler using Compression, we could be more e cient by performing the successive steps of Compression at each execution of IdealToIsogenySmallFromEichler. Typically, the deterministic basis generation of Compression could be done at the same time as Step 1 of EndomorphismEvaluation in the choice of Q and so the value $a; b$ would be used to derive the nal representation. This is what we do for the implementation results presented in Chapter 5.

Handling failures. In Section 3.5, we explained that there are some inputs $O; K$ for which the computation of $\text{SpecialEichlerNorm}_{O(T)}(O; K)$ will fail if $T \ p^{5=4}$. If one of the O_i is one of those bad orders, the execution of IdealToIsogenySmallFromEichler at the i-th iteration in IdealToIsogenyFromEichler will fail. Since we cannot a ord to increase the size of T, we need another way

to handle the failures. Short of finding a new (and better) method, there are basically two options: use an adapted version of IdealToIsogenySmallFromKLPT to perform the translation, or use a special extremal order other than O_0 with SpecialEichlerNorm.

At first glance, going back to IdealToIsogenySmallFromKLPT might seem like an odd thing to do as we explained that the requirement for the size of T were greater than in IdealToIsogenySmallFromEichler. However, the failure cases for SpecialEichlerNorm are actually good cases for the method using KLPT because there is an ideal of norm $M \quad p^{1=2}$ connecting O_0 and O. As we explained in Section 3.5, this is only a bad thing for SpecialEichlerNorm because we have an additional constraint with the ideal K but KLPT does not suffer from the same limitation and should succeed to find an element of norm T^2 if $T \quad pM$. Hence, when $M < p^{1=4}$, we can hope to make it work with $T \quad p^{5=4}$. However, there is an obvious range of degrees $p^{1=4} \quad M \quad p^{1=2}$ where this solution will not work. This is why, in practice, we will use the second method described below.

Using another extremal order. The bad property resulting in failures directly depends on the special extremal order O_0 (which is unique by Definition 3.1.1). By looking at the -special extremal orders for values of slightly above the minimal one, we can gather a collection extremal orders in which we can run SpecialEichlerNorm with a negligible efficiency loss (we saw that the complexity was linear in $h(O)$, where we recall that O is the small quadratic order of discriminant d contained in the extremal order). Thus, it suffices to enumerate through our small set of extremal orders until we find one that does not have the bad property. To prove that this idea works, we need to make sure that a maximal order O will not have the bad property with all the extremal orders. Unfortunately, we do not have a definitive proof of this fact and are reduced to make it a heuristic assumption. Boneh and Love [LB20] showed that maximal quaternion orders admitting embeddings of small quadratic orders are far from one another. While this conveys the right idea, their bound in [LB20, Proposition 4.5] is too loose to help us. In practice, switching to another maximal order seems to work well.

Isogeny to ideal. In this section, we presented several methods to perform efficiently the ideal to isogeny translation. We chose this task because it will be the crucial part in several of our cryptographic applications. However, the algorithms that we have presented can be adapted to perform the reverse translation of computing the ideal corresponding to a chain of ℓ^f-isogenies. The method is roughly the same (up to replacing IdealToIsogeny with IsogenyToIdeal). However, note that the tricks we introduced for IdealToIsogenySmallFromKLPT to translate an ideal of norm ℓ^{2f+} instead of an ideal of norm ℓ^f appear hard to translate to the setting of IsogenyToIdeal. We do not give a full description of \isogeny-to-ideal" variants of IdealToIsogenyFromKLPT and IdealToIsogenyFromEichler as we do not need it and it does not appear to require any new interesting ideas.

4.2.3 Verification of the ideal representation

In this section, we come back to one of our initial motivation, which is to prove that a triple $D; E_1; E_2$ is in L_{isog}. The main result is Proposition 4.2.13 that

proves that we have a polynomial time verification algorithm.

Note that when the prover is unbounded, it is clear that he can compute the compact representation of an ideal I whose corresponding isogeny is connecting E_1 and E_2. Indeed, since there is a finite number of maximal orders and ideals of a given norm inside $B_{p,1}$, the prover can simply enumerate through all of them until finding a fitting one.

We now present IdealVerification, a verification algorithm that takes a triple $x = (D; E_1; E_2)$ and an ideal I and decides if $x \in L_{\text{isog}}$. The idea is to use the following procedure on ideals connecting a special order O_0 with $O_L(I)$ and $O_R(I)$: use KLPT to get an equivalent ideal of smooth norm and compute the corresponding isogeny with IdealToIsogenyFromEichler (note that we could have used IdealToIsogenyFromKLPT as well). This algorithm requires some parameter $f; T$ that we select as explained in Remark 4.2.10 to ensure termination in polynomial time with overwhelming probability.

Algorithm 27 IdealVerification$(x; I)$

Input: $x \in \mathbb{N} \times S(p)^2$ and I an ideal of $B_{p,1}$.
Output: A bit indicating if $x \in L_{\text{isog}}$.

Parse x as $D; E_1; E_2$ and take ℓ a small prime.
Compute $n(I)$ and $O_L(I); O_R(I)$.
if $n(I) \neq D$ or $I \not\subset O_L(I)$ then
  Return 0.
end if
Take O_0, the special extremal order of $B_{p,1}$ and E_0 the curve in $S(p)$ with $\text{End}(E_0) = O_0$;
Compute $I_1 = \text{ConnectingIdeal}(O_0; O_L(I))$, $I_2 = I_1 \cdot I$.
for $i \in [1; 2]$ do
  Compute $J_i = \text{KLPT}(I_i)$ and $\theta_i : E_0 \to E_i' = \text{IdealToIsogenyFromEichler}(J_i; O_0; [1])$.
end for
if $j(E_1'); j(E_2') \in \{j(E_1); j(E_2)); (j(E_1)^p; j(E_2)^p)\}$ then
  Return 0.
end if
return 1.

Lemma 4.2.12. *Let ℓ be any integer in $\mathbb{N}$ coprime to p. If $\theta : E_1 \to E_2$ has degree ℓ, then* IdealVerification$((D; E_1; E_2); I_\theta) = 1$.

Conversely, for $(D; E_1; E_2) \in \mathbb{N} \times S(p)^2$, if there exists an ideal I such that IdealVerification$((D; E_1; E_2); I) = 1$ *then $(D; E_1; E_2) \in L_{\text{isog}}$.*

Proof. Let us take $\theta : E_1 \to E_2$ of degree ℓ. By definition of I_θ, we have $n(I_\theta) = D$ and $I_\theta \subset O_L(I)$ so the first check passes. Then, the codomain of the two θ_{I_i} have endomorphism ring isomorphic to $O_R(I_i)$ so they might be either both E_i or both E_i^p (since $I_2 = I_1 I$, it cannot be $E_1; E_2^p$ or $E_1^p; E_2$). In both cases, the final output is 1.

If there exists an ideal I such that IdealVerification$((D; E_1; E_2); I) = 1$, then $n(I) = D$ and I is integral (this is from the first verification). Since $I = \overline{I_1} \cdot I_2 / n(I_1) \cdot \overline{J_1} \cdot J_2$ is an integral ideal of degree ℓ, there exists an isogeny of degree ℓ between $E_1'; E_2'$. Since the final output is 1, the two curves $E_1'; E_2'$ are equal

to either $E_1; E_2$ or $E_1^p; E_2^p$. Since $' : E_1^p \to E_2^p$ of degree ℓ imply the existence of $'^p : E_1 \to E_2$ of degree ℓ, in both cases we have that $(D; E_1; E_2) \in L_{\text{isog}}$. □

Proposition 4.2.13. *(UPHA)* IdealVerification *succeeds with overwhelming probability and terminates in expected* $O(\text{poly}(\log(pD)))$.

Proof. We consider that $\ell = O(1)$. The basis elements of left and right orders of an ideal of norm ℓ can be written in $O(\log(pD))$ bits by Lemma 2.2.5. Then, the results follows from Propositions 2.2.4, 3.2.7 and 4.2.9 with a choice of T (as in Remark 4.2.10 with smoothness bound in $O(\text{poly}(\log(pD)))$ and $f = O(\text{poly}(\log(pD)))$. With this constraint, we can expand the size of T as we need to get a success probability that is overwhelming according to Proposition 4.2.9. □

4.2.4 Isogeny Evaluation from the ideal representation

In this section, we show how to evaluate the isogeny $'_I$ on any point of order coprime to $n(I)$. For simplicity, we assume that I is an O_0-ideal, where O_0 is the special extremal order of $B_{p;1}$. We make this assumption, so we can evaluate efficiently the endomorphism of E_0. A generic algorithm of complexity $O(\text{poly}(\log(pD))$ can be derived using the tools we have developed, but the special case suffices to illustrate our point. An algorithm very similar to IdealEvaluation can be found in [FKM21]. The main idea is to apply KLPT and IdealToIsogenyFromEichler to find an equivalent isogeny of prime power degree and making use of it to perform the computation. As for IdealVerification, we will pick parameters $T; f$ to ensure success with overwhelming probability.

Algorithm 28 IdealEvaluation$(I; P)$

Input: I an O_0-ideal of $B_{p;1}$ and $P \in E_0(\overline{\mathbb{F}_q})$ of order coprime to $D = n(I)$.
Output: $'_I(P)$.

1: Take a small prime number ℓ.
2: Compute $J = \text{KLPT}_\ell(I)$ and set $K = I\,\overline{J}$. We write $\theta \in \text{End}(E)$ for the endomorphism $'_K$.
3: Compute $\theta(P)$.
4: Compute $'_J = \text{IdealToIsogenyFromEichler}_\ell(J; O_0; [1])$ and compute $Q = '_J(\theta(P))$.
5: Compute $\theta = n(J)^{-1} \bmod n(I)$.
6: **return** $[\theta]Q$.

Proposition 4.2.14. *(UPHA)* IdealEvaluation *is correct, succeeds with overwhelming probability, and terminates in probabilistic* $O(\text{poly}(\log(pD)))$ *operations over* $\mathbb{F}_q$.

Proof. We have $'_K = \hat{}_J \circ '_I$, and so $'_J(\theta(P)) = '_I(P)$. The division by θ makes sense mod $n(I)$ since the order of P is coprime to $n(I)$. The correctness of IdealEvaluation follows from the correctness of the sub-algorithms KLPT, IdealToIsogenyFromEichler. Step 3 can be executed because of our assumption on E_0. If we assume that $\ell = O(1)$, termination is a consequence of Propositions 2.2.4, 3.2.7 and 4.2.9 and that the computation of $'_J(P)$ can

be done in $O(\text{poly}(\log(p)))$ operations over the field of definition of P since $\deg \phi' = O(\text{poly}(p))$ and have smoothness bound in $O(\text{poly}(\log(p)))$. With the choice of T as explained in Remark 4.2.10, the algorithm succeeds with overwhelming probability. □

4.3 The suborder representation

In this section, we introduce a new isogeny representation that we call the *suborder representation*. This new representation is the subject of our paper [Ler21]. As for the ideal representations, the suborder representation relies heavily on the tools provided by the Deuring correspondence. The motivation behind this new construction is to be found in the cryptographic applications. In Chapter 7, we will argue that the suborder representation is not equivalent to the ideal representation under the hardness of a new problem and we will give examples of protocol that we can build upon this assumption.

We focus on the case where the degree D is prime because it is simpler. We will explain later in Chapter 7 why it appears to be the most promising for our applications. We will explain only informally how to deal with the composite order in the end of Section 4.3.2.

From now on, unless stated otherwise, D is prime. The suborder representation has also a small limitation: we can only verify that either E_1, E_2 are D-isogenous or E_1, E_2^p are D-isogenous when $\text{End}(E_1) \not\simeq \text{End}(E_2)$. Thus, we consider the alternate language $L_{\text{p\_isog}}$ defined as follows:

$$\{(D, E_1, E_2) \in \mathbb{P} \;|\; S(p)^2 \mid E_1 \not\simeq E_2, E_2^p \text{ and } (D, E_1, E_2) \in L_{\text{isog}} \text{ or } (D, E_1, E_2^p) \in L_{\text{isog}}\}$$

A brief overview. The main mathematical result underlying our new representation is Proposition 2.3.16. When D is prime, this result states that the quaternion sub-order $\mathbb{Z} + D\,\text{End}(E_1)$ is embedded inside $\text{End}(E_2)$, if and only if either $\text{End}(E_2) = \text{End}(E_1)$ or $(D, E_1, E_2) \in L_{\text{isog}}$. Thus, our new representation will be constituted of a maximal order $O_1 = \text{End}(E_1)$ and a concrete embedding of $\mathbb{Z} + DO$ inside $\text{End}(E_2)$, and this is what we concretely call a *suborder representation*. We highlight that O_1 is simply given as an order inside $B_{p,1}$ (through a basis of 4 quaternion elements), whereas the embedding of $\mathbb{Z} + DO_1$ is made of isogenies of smooth degree from E_2 to E_2. The suborder representation can be verified by computing the traces of the endomorphisms revealed in this manner.

4.3.1 Deriving the new representation from the ideal representation

The goal of this section is to introduce an algorithm IdealToSuborder that takes a maximal order O_1 and a O_1-ideal I of norm ℓ and outputs a representation of the embedding $\mathbb{Z} + DO_1 \hookrightarrow \text{End}(E_2)$. By a representation, we actually mean the embeddings of a *generating family* for $\mathbb{Z} + DO_1$ (see Definition 3.4.3).

Definition 4.3.1. Let $\phi : E_1 \to E_2$ be an isogeny of degree D. *A suborder representation* Φ for ϕ is made of an order $O_1 = \text{End}(E_1)$ and of compressed representations $s_1, \ldots, s_n$ of n endomorphisms of E_2 corresponding to a generating family of $\mathbb{Z} + DO_1$.

Our algorithm IdealToSuborder (Algorithm 29) is built upon the GeneratingFamily algorithm from Chapter 3 that will be used to generate the endomorphism of the suborder representation. We recall that GeneratingFamily$_N$ on input $O_1; D$ computes a generating family for $\mathbb{Z} + DO_1$ (see Definition 3.4.3) whose elements have their norm in N.

IdealToSuborder can be divided in two main parts: GeneratingFamily to obtain quaternion elements $_1; \dots; _n$ and an IdealToIsogenyFromEichler step to convert the ideals $O_R(I)\ _i$ to isogenies $'_i : E_2 \to E_2$. For all the algorithms of this section, we are going to fix a small constant prime ℓ.

Algorithm 29 IdealToSuborder(I)

Input: I an integral ideal of maximal orders inside $B_{p;1}$ of norm ℓ.

Output: Endomorphisms $'_i : E_2 \to E_2$ such that $: \mathrm{End}(E_2) \to O_R(I)$ sends $'_1; \dots; '_n$ to a generating family $_1; \dots; _n$ for $\mathbb{Z} + DO_L(I)$.

1: Compute $D = n(I)$ and $O = O_L(I); O' = O_R(I)$ and select a small prime ℓ.
2: Compute J = ConnectingIdeal$(O_0; O')$
3: Compute K = KLPT$_{}$ (J).
4: Compute $'_K$ = IdealToIsogenyFromEichler$_{}$ $(K; O_0; [1])$.
5: Compute $_1; \dots; _n$ = GeneratingFamily$_{}$ $(O; D)$.
6: **for** $i \in [1; n]$ **do**
7: Compute $'_i : E_2 \to E_{2;i}$ = IdealToIsogenyFromEichler$_{}$ $(O'\ _i; K; '_K)$.
8: Compute s_i = Compression$_{n(\ _i)}(E_2; '_i)$.
9: **end for**
10: **return** $O; (s_i)_{1 \le i \le n}$.

Proposition 4.3.2. *(UPHA)* IdealToSuborder *is correct, succeeds with overwhelming probability and terminates in* $O(\mathrm{poly}(\log(pD)))$ *and the output has size* $O(\mathrm{poly}(\log(pD)))$.

Proof. Correctness follows from the correctness of IdealToIsogenyFromEichler and GeneratingFamily. Similarly to ideals and Lemma 2.2.5, the left and right orders $O; O'$ admits a representation of size $O(\mathrm{poly}(\log(pD)))$. Termination and the overwhelming success probability follows from Propositions 3.2.7, 3.4.4 and 4.2.9 (with $n = O(1)$ and the parameters $T; f$ as in Remark 4.2.10). □

4.3.2 Verification of the suborder representation

This section focuses on the verification of the representations computed with IdealToSuborder. From Proposition 2.3.16, we know that it suffices to convince the verifier that $\mathbb{Z} + D\,\mathrm{End}(E_1)$ is embedded inside $\mathrm{End}(E_2)$ and $\mathrm{End}(E_1) \not\subset \mathrm{End}(E_2)$. The second part is easy to verify, it suffices to compute the j-invariants and verify that neither $j(E_1) = j(E_2)$ nor $j(E_1) = j(E_2)^p$. We achieve the first part of the verification with the endomorphisms $'_1; \dots '_n$. With, Lemma 4.3.3, we show that it suffices to check some traces and norms of endomorphisms computed from the $('_i)_{1 \le i \le n}$. We remind the reader that the Order() notation was introduced in Definition 3.4.3.

Lemma 4.3.3. *Two orders* $O_1 = \mathrm{Order}(\alpha_1, \ldots, \alpha_n)$ *and* $O_2 = \mathrm{Order}(\beta_1, \ldots, \beta_n)$ *of rank 4 in a quaternion algebra are isomorphic if* $n(\alpha_i) = n(\beta_i)$ *for all* $i \in [1, n]$ *and* $\mathrm{tr}(\prod_{j \in I} \alpha_j) = \mathrm{tr}(\prod_{j \in I} \beta_j)$ *for all* $I \subset [1, n]$.

Proof. In our setting, two quaternion orders are isomorphic if their norm forms are the same. Thus, we are going to give a bijection $\varphi : O_1 \to O_2$ and verify that it preserves norm and traces. We label $\alpha_0, \alpha_1, \ldots, \alpha_m$ (resp. $\beta_0, \beta_1, \ldots, \beta_m$) with $m = 2^n - 1$ the set of multi-products obtained from $\alpha_1, \ldots, \alpha_n$ (resp. $\beta_1, \ldots, \beta_n$), the multi-product α_0 (resp. β_0) corresponding to the empty set is simply 1. By the definition of a generating family, any element $\alpha \in O_1$ (resp. O_2) can be written as a linear combination of $\alpha_0, \ldots, \alpha_m$ (resp. $\beta_0, \ldots, \beta_m$). We are going to prove that the map $\varphi : \sum_{i=0}^{m} x_i \alpha_i \mapsto \sum_{i=1}^{m} x_i \beta_i$ is an isomorphism of quaternion orders. It is easy to verify that this map is bijective. It remains to check that it preserves the trace and the norm when $n(\alpha_i) = n(\beta_i)$ and $\mathrm{tr}(\alpha_i) = \mathrm{tr}(\beta_i)$ for all $i \in [0, m]$.

The trace being linear, it is clear that $\mathrm{tr}(\varphi(\alpha)) = \mathrm{tr}(\alpha)$ for all $\alpha \in O_1$. For any $\alpha = \sum_{i=0}^{m} x_i \alpha_i$, we have $n(\alpha) = \sum_{0 \le i < j \le m} x_i x_j \mathrm{tr}(\alpha_i \hat{\alpha}_j) + \frac{1}{2} \sum_{i=0}^{m} x_i^2 \mathrm{tr}(\alpha_i \hat{\alpha}_i)$. Thus, we need to prove that we have equality of traces for all $\alpha_i \hat{\alpha}_j$ and $\beta_i \hat{\beta}_j$. Since $\mathrm{tr}(ab) = \mathrm{tr}(ba) = \mathrm{tr}(\hat{a}\hat{b})$ and $\mathrm{tr}(a)\,\mathrm{tr}(b) = \mathrm{tr}(ab) + \mathrm{tr}(a\hat{b})$ for all $a, b \in B_{p,\infty}$, it suffices to verify the equality $\mathrm{tr}(\prod_{j \in I} \alpha_j) = \mathrm{tr}(\prod_{j \in I} \beta_j)$ to get the desired result. This also proves that we have equality of norms between α and $\varphi(\alpha)$. □

As Lemma 4.3.3 indicates, we need to compute some traces for the verification. This will be done by an algorithm $\mathrm{CheckTrace}_M$ (whose description we postpone until Section 4.3.3). The goal of this algorithm is to verify the validity of the traces modulo the parameter M (see Proposition 4.3.7).

Below, we give a bound above which equality will hold over $\mathbb{Z}$ if it holds mod M.

Lemma 4.3.4. *Given any* $\alpha \in \mathrm{End}(E_1)$, *if* $\mathrm{tr}(\alpha) = t \bmod M$ *for* $M > 4\sqrt{n(\alpha)}$ *and* $|t| < M/2$, *then* $\mathrm{tr}(\alpha) = t$.

Proof. Over $B_{p,\infty}$, the norm form is $n : (x, y, z, w) \mapsto x^2 + qy^2 + pz^2 + qpw^2$ where $q > 0, p > 0$. Since $\mathrm{tr} : (x, y, z, w) \mapsto 2x$, we can easily verify that $\mathrm{tr}(\alpha)^2 < 4n(\alpha)$. This gives a bound of $2\sqrt{n(\alpha)}$ on the absolute value of $\mathrm{tr}(\alpha)$. The result follows. □

Proposition 4.3.5. *If*

$$M > \max_{1 \le j \le n} 2\sqrt{n(\beta_j)^n},$$

then for $x \in \mathbb{P}S(p)^2$, *there exists a suborder representation* ι *such that*

$$\mathrm{SuborderVerification}_M(x, \iota) = 1$$

if and only if $x \in L_{p,\mathrm{isog}}$.

Proof. Assume that there exists a representation ι passing the verification for a given $x = (D, E_1, E_2)$. The check in Step 2 proves that O is a maximal order of $B_{p,\infty}$. The second verification in Step 8 proves that $\mathrm{End}(E_1) = O$. Finally, the verification in Step 13 proves that the θ_i are endomorphisms of E_2. Then, if $\mathrm{CheckTrace}_M(\theta_1, \ldots, \theta_n, \alpha_1, \ldots, \alpha_n, E_2) = 1$, the correctness of

Algorithm 30 SuborderVeri cation$_M(x;\ )$

Input: $x \in \mathrm{P}\ \ S(p)^2$ and a suborder representation.
Output: A bit indicating if $x \in L_{\mathrm{p\ isog}}$.

1: Parse x as $D; E_1; E_2$ and $= O; (s_i)_{1\ i\ n}$.
2: **if** If disc(O) p **then**
3: Return 0.
4: **end if**
5: Compute $_1; : : : ; \ _n =$ GeneratingFamily$_{\ }(O; D)$.
6: Compute $J =$ ConnectingIdeal$(O_0; O)$ and $L =$ KLPT$_{\ }(J)$.
7: Compute $: E_0 \ ! \ E_1^0 =$ IdealToIsogenyFromEichler$_{\ }(L)$.
8: **if** $j(E_1)$ $j(E_1^0)$ or $j(E_1)$ $j(E_1^0)^p$ **then**
9: Return 0.
10: **end if**
11: **for** $i \in [1; n]$ **do**
12: Compute $'_i : E_2 \ ! \ F_i =$ Decompression$_{n(\ i)}(E_2; s_i)$
13: **if** $j(F_i)$ $j(E_2)$ **then**
14: Return 0.
15: **end if**
16: **end for**
17: **return** CheckTrace$_M('_1; : : : ; '_n; \ _1; : : : ; \ _n; E_2)$.

GeneratingFamily and CheckTrace, Lemmas 4.3.3 and 4.3.4 imply that $Z + DO$ is embedded inside End(E_2) and Proposition 2.3.16 proves that $x \in L_{\mathrm{p\ isog}}$.

Now let us take $(D; E_1; E_2) \in L_{\mathrm{p\ isog}}$. By de nition, there exists an ideal I of norm ℓ and $O_L(I) =$ End(E_1), $O_R(I) =$ End(E_2). We are going to show that if $=$ IdealToSuborder(I), then we have SuborderVeri cation$_M(x;\ ) = 1$. First, since $O_L(I)$ is a maximal order, the veri cation of Step 2 passes successfully. This is also the case for the veri cation of Step 8 since $O_L(I) =$ End(E_1). Then, by the correctness of IdealToSuborder showed in Proposition 4.3.2, we have that s_i can be parsed as isogenies $'_i : E_2 \ ! \ E_2$ that correspond to the $O_R(I)_{\ i}$ through the Deuring Correspondence (since GeneratingFamily is deterministic). Thus, it is clear that CheckTrace will output 1, and this concludes the proof. □

With Proposition 3.4.4 and Proposition 4.3.5, we see that we can take $M = \#E(\mathrm{F}_{p^k})$ for some $k \in \mathrm{N}$ big enough to get that the veri cation algorithm SuborderVeri cation$_M$ is correct.

Proposition 4.3.6. *(UPHA) Let $k; M$ be as de ned above.* SuborderVeri cation$_M$ *terminates in probabilistic* $O(\mathrm{poly}(\log(pD)))$ *with overwhelming probability.*

Proof. We have $k = O(\mathrm{poly}(\log(pD)))$ by Proposition 4.3.5, and this implies $M = O(\mathrm{poly}(pD))$ by Proposition 1.1.23. When $N = \ell$, Proposition 3.4.4 implies that the execution of GeneratingFamily$_{\ }$ will succeed with overwhelming probability. Then, the expected running time follows from Propositions 2.2.4, 3.4.4 and 4.3.7. □

Veri cation in the composite case. Now, we explain brie y how to extend the SuborderVeri cation to perform the veri cation when the degree D is composite. With Proposition 2.3.16, we know that we need to check that the

embedding of $\mathbb{Z} + D\,\mathrm{End}(E_1)$ inside $\mathrm{End}(E_2)$is primitive. When D is prime, the embedding is either primitive or E_1 and E_2 are isomorphic, which is why it suffices to check that the latter is not true. When D is composite, the verification that the embedding is primitive is more complicated. For that, we need to check that $(\mathbb{Z} + D\,\mathrm{End}(E)) \not\subset \mathbb{Z} + N\mathcal{O}$ for some order $\mathcal{O} \subset \mathrm{End}(E_2)$ and $N \mid D$. Since $\mathcal{O} = \mathbb{Z} + (D/N)\,\mathrm{End}(E)$, it suffices to find one endomorphism $\alpha = d + \beta\hat{\ }$ and prove that $d' + (\beta - d)/N$ is not an endomorphism of E_2 to prove that $(\mathbb{Z} + D\,\mathrm{End}(E)) \not\subset \mathbb{Z} + N\mathcal{O}$ for any N of $\mathcal{O}$. If the norm of $d' + (\beta - d)/N$ is powersmooth and coprime to N, $G_N = \ker(d' + (\beta - d)/N)$ and E_2/G_N can be computed efficiently. Thus, the additional verification mechanism works as follows: for every prime N dividing D, use GeneratingFamily to compute a generating family $\gamma_1, \ldots, \gamma_n$ of norm coprime to N of $\mathbb{Z} + (D/N)\,\mathrm{End}(E)$, express each γ_i as $d' + (\beta_i - d)/N$ where $\beta_i \in \mathbb{Z} + D\,\mathrm{End}(E)$ and compute $G_{N,i} = \ker d' + (\beta(\gamma_i) - d)/N$. If there exists one N such that $j(E_2/G_{i,N}) = j(E_2)$ for all $1 \le i \le n$, the verification fails.

4.3.3 Checking traces

In this section, we present an algorithm $\mathrm{CheckTrace}_M$ to perform the verification of the suborder representation.

Computing the trace of an endomorphism is a well-studied problem, as it is the primary tool of the point counting algorithms such as SEA [Sch95]. For our application the task is even simpler as we merely have to verify the correctness of the alleged trace value and not compute it. With the formula $\mathrm{tr}(\alpha) = \alpha + \hat{\alpha}$, it suffices to evaluate α and $\hat{\alpha}$ on a basis of the M-torsion, and then verify the relation. In particular, we do not need M to be smooth since we just want to check equality.

Algorithm 31 $\mathrm{CheckTrace}_M(E, \beta'_1, \ldots, \beta'_n, \beta_1, \ldots, \beta_n)$

Input: $\beta_1, \ldots, \beta_n$, n endomorphisms of E and n elements of $B_{p,\ell_1\cdots\ell_n}$.
Output: A bit b equal to 1 if and only if $\mathrm{tr}(\beta_i) = \mathrm{tr}(\beta'_i) \bmod M$ for all $i \in [1, n]$.

1: Compute P, Q a basis of $E[M]$ over the appropriate field extension. Set $b = 1$.
2: **for** All $I \subset [1, n]$ **do**
3: Set $\beta_I = \prod_{j \in I} \beta_j$ and $\beta'_I = \prod_{j \in I} \beta'_j$.
4: Verify $\beta'_I(R) + \hat{\beta}_I(R) = [\mathrm{tr}(\beta_I)]R$ for $R \in \{P, Q\}$. If not, set $b = 0$.
5: **end for**
6: **return** b.

Proposition 4.3.7. *When $M > \#E(\mathbb{F}_{p^k})$, $n = O(1)$ and $\deg \beta'_i = O(\mathrm{poly}(pD))$ and have smoothness bound in $O(\mathrm{poly}(\log(pD)))$ for all $1 \le i \le n$, $\mathrm{CheckTrace}_M$ terminates in $O(\mathrm{poly}(k \log(pD)))$*

Proof. By definition of M, the points P, Q are defined over $\mathbb{F}_{p^k}$ and so operations over the M-torsion have $O(\mathrm{poly}(k \log(p)))$ complexity. By the assumption on the degree of the β'_i, computing all the $\beta'_I(P, Q)$ can be done in $O(\mathrm{poly}(\log(p)))$ since $n = O(1)$, and this concludes the proof. □

4.3.4 Evaluating with the suborder representation

In this section, we show that we can evaluate the isogeny φ from the suborder representation (in Section 4.2.4, we described Algorithm 28 to do that same operation from an ideal representation). The algorithm SuborderEvaluation that we introduce below is going to be one of the major building blocks behind one of the construction in Part II. In fact, we achieve something slightly less powerful than IdealEvaluation, as SuborderEvaluation computes images of cyclic subgroups rather than points. SuborderEvaluation can be extended to perform the same operation as IdealEvaluation but we do not need it here. For the sake of our application, we also choose to give the input as an ideal J rather than a subgroup. The output will then be $\varphi(E[J])$.

The SuborderEvaluation algorithm uses the sub-algorithm IdealSuborderEichler-Norm that we introduced in Section 3.4.

The principle of SuborderEvaluation is different from the one of IdealEvaluation. Since the suborder representation do not give the full endomorphism ring of E_2, we cannot apply KLPT to find another path of nice degree (which is the key step in IdealEvaluation). Instead, we propose to use the fact that the embedding of $\mathbb{Z} + D\,\mathrm{End}(E_1)$ inside $\mathrm{End}(E_2)$ is obtained by push-forward through φ. More precisely, this means that $\ker(\theta) = \varphi(\ker\psi)$ for any $\psi \in \mathbb{Z} + D\,\mathrm{End}(E_1)$. Thus, to find $\varphi(E_1[J])$, we want to find an endomorphism $\psi \in \mathbb{Z} + D\,\mathrm{End}(E_1)$ such that $\ker\psi \cap E_1[n(J)] = E_1[J]$. By definition of $E_1[J]$, and such a ψ is exactly found by IdealSuborderEichlerNorm. After that, it suffices to compute $\ker(\theta) \cap E_2[n(J)]$ and we are done. The integer k is taken as in Proposition 4.3.6.

Proposition 4.3.8. *(UPHA)* SuborderEvaluation *is correct when the output is not ?, and terminates with overwhelming probability in* $O(\mathrm{poly}(\log(pD)) + \sqrt{n(J)})$ *operations over the* $n(J)$ *torsion.*

Proof. First, we will prove correctness. The verification at the beginning proves that if the output is not ?, ψ is a valid suborder representation.
When $L = \mathrm{ConnectingIdeal}(\mathcal{O}_0, \mathcal{O})$ and $I = \mathrm{RandomEquivalentPrimeIdeal}(L)$ with $I \neq L$, then if $\psi \in (\mathbb{Z} + DI) \setminus \theta^{-1}J$, then $\theta^{-1}\psi \in (\mathbb{Z} + DL) \cap J \subset (\mathbb{Z} + D\mathcal{O}) \cap J$. This explains that we can decompose $\theta^{-1}\psi$ on the generating family $1, \ldots, \gamma_n$. Since θ gives a correct embedding of $\mathbb{Z} + D\mathcal{O}$ inside $\mathrm{End}(E_1)$ and so $\varphi = \sum_{i \in \{1,\ldots,n\}} c_i \prod_{j \in I} \varphi_j$ is an endomorphism of E_2 whose degree is a multiple of $n(J)$. To conclude the proof of correctness, it suffices to show that $\ker\varphi \cap E_2[n(J)] = \varphi(E_1[J])$. If $\theta^{-1}\psi = [d] + [D]$ for some $\psi \in \mathrm{End}(E_1)$, we have that $\varphi = [d] + \varphi\psi\varphi^{-1}$. Now let us take $P_0 \in E_1[J]$. Since $\theta^{-1}\psi \in J$, we have $([d] + [D])P_0 = 0$ and $(\varphi(P_0)) = [d]\varphi(P_0) + \varphi(\psi\varphi^{-1}\varphi(P_0)) = \varphi(([d] + [D])P_0 = 0$. This proves that $\varphi(E[j]) \subset \ker\varphi \cap E_2[n(J)]$. And we obtain equality since the two subgroups have the same order. Thus, we have showed that our protocol is correct.

Then, the complexity follows from Propositions 2.2.4, 3.4.4, 3.4.6 and 4.3.6 and the fact that $n(I) = O(\mathrm{poly}(p))$. □

Algorithm 32 SuborderEvaluation($E_1; E_2 ; D; J$)

Input: two curves $E_1; E_2$, a prime $D;$ a suborder representation for $(D; E_1; E_2) \; 2 \; L_{\text{p isog}}$ and an ideal J of norm coprime to $\grave{}$.
Output: $?$ or $'(E_1[J])$.

1: Parse as $O; s_1; \; ; s_n$.
2: Compute $_1; : : : ; \; _n$ = GeneratingFamily$\grave{}(O; D)$.
3: Compute $'_i$ = Decompression$_{n(\ _i)}(E_2; s_i)$.
4: **if** $O_L(I) \neq O$ **then**
5: Return $?$.
6: **end if**
7: **if** SuborderVeri cation$_{\#E_1(\mathbb{F}_{p^k})}((D; E_1; E_2); \ ) = 0.$ **then**
8: Return $?$.
9: **end if**
10: Compute L = ConnectingIdeal$(O_0; O)$ and I = RandomEquivalentPrimeIdeal(L) with $I = L$.
11: Compute = IdealSuborderEichlerNorm$\grave{}(D; I; \ ^1J \ )$.
12: Express $^1 = \sum_{I \ f1;:::;ng} c_{i;I}(\prod_{j2I} \ _j)$.
13: Compute $P; Q$, a basis of $E_2[n(J)]$.
14: Compute $R; S = \sum_{I \ f1;:::;ng} c_{i;I}(\prod_{j2I} \ '_j)(P; Q)$.
15: **if** $S = 0$ **then**
16: **return** $\langle Q \rangle$.
17: **end if**
18: Compute $a = \text{DLP}_{n(J)}(R; S)$.
19: **return** $\langle P \ [a]Q \rangle$.

Part II

Cryptographic protocols

Chapter 5

Signatures: SQISign

In this chapter, we introduce SQISign (short for Supersingular Quaternion and Isogeny Signature), a new signature scheme explicitly based on the algorithms of the Deuring correspondence from Chapter 3 and Chapter 4. This chapter mixes the content of our articles [DFKL+20], which introduced the SQISign protocol, and [DFLW22], which presented various improvements and results on cryptanalysis.

The principle behind our construction was first introduced by Galbraith, Petit and Silva [GPS17] in the so-called GPS signature protocol. The common idea is the following: use the knowledge of a curve's endomorphism ring to find paths in the isogeny graphs. This approach uses the norm-equation algorithms from Chapter 3 and the ideal-to-isogeny algorithms from Chapter 4. More precisely, the GPS protocol uses the KLPT algorithm (see Algorithm 7 in Section 3.2.3) to perform the computation in $B_{p,1}$ before translating it into an isogeny. The authors of GPS introduced the algorithms IdealToIsogeny and IsogenyToIdeal to go back and forth between the different isogeny representations (we presented those algorithms in Section 4.2.1). For SQISign, we will use GenericKLPT (Algorithm 11 in Section 3.3) for the computation over the quaternions and the efficient algorithms IdealToIsogenyFromKLPT or IdealToIsogenyFromEichler (see Section 4.2.2) to translate the output from GenericKLPT into an isogeny. Our new protocol can thus be seen as an improved version of GPS: with these new algorithmic tools, we get a scheme that is much more compact and efficient.

In Section 5.1 we introduce some useful preliminaries on signatures, and briefly present the state of isogeny-based signatures prior to the publication of [DFKL+20]. The rest of this chapter is dedicated to SQISign, its detailed description, and its security analysis.

5.1 Preliminaries

We start with a few security definitions for identification protocols and the Fiat–Shamir transform in Section 5.1.1. In Section 5.1.2, we give more background on GPS and isogeny-based signature schemes.

5.1.1 Identi cation protocols and Fiat{Shamir signatures

We brie y recall here the standard security de nitions for sigma-protocols (see [Dam10, Kat10, Ven15] for precise references).

A sigma protocol is a 3-round public-coin interactive protocol between a prover and a veri er. For R, a relation on a set $Y \quad X$, we de ne the language $L = fy\ 2\ Y; 9x\ 2\ X; R(x;y) = 1g$. The veri er receives $y\ 2\ L$ and the prover holds a witness x for this y and wants to prove it to the veri er without revealing x. A transcript is a triple $(a;b;c)$ where a is the commitment, b the challenge and c is the response. Intuitively, the role of the commitment is to bind the prover to some information at the beginning of the interaction with the veri er. Publishing that commitment before seeing anything from the veri er will force him to behave correctly. Then, comes the challenge from the veri er. This is a random string that the prover should not be able to anticipate without negligible probability. The response is then computed by the prover from his secret key, the commitment and the challenge. In the end, the veri er outputs a bit, indicating whether it accepts the transcript. When the veri er outputs 1, we saiy that the transcript is *accepting*.

De nition 5.1.1. A sigma-protocol is *complete* if the veri er outputs 1 with probability 1 when there exists a witness. A sigma-protocol is *special sound* if there exists a polynomial time algorithm, called an extractor, that can recover the witness x from two accepting transcripts $(a;b;c)$ and $(a;b^0;c^0)$ sharing the same commitment. A sigma-protocol is (computationally) *honest veri er zero-knowledge* if there exists a polynomial time algorithm, called a simulator,that on input $y\ 2\ L$, but without access to the witness x, generates accepting transcripts that are (computationally) indistinguishable from transcripts of the real protocol.

It is a classical result (see for instance [Ven15]) that canonical identi cation schemes secure against impersonation under passive attacks can be constructed from complete, special sound and honest veri er zero-knowledge sigma-protocols. A signature scheme unforgeable under chosen message attacks can be derived from such an identi cation scheme using the Fiat{Shamir transform [FS86]. The main goal of this transform is to remove the interaction from the interactive protocol. This can be done by replacing the veri er's challenge by the value $H(M\ k\ a)$ where H is a hash function, a is the commitment and M is the message. The idea is to prevent the prover from computing his commitment with the knowledge of the challenge. This way of constructing signatures from sigma-protocols is standard, and we refer the reader to [AABN02] for the proof of the following result:

Theorem 5.1.2. *Let ID be a non-trivial canonical identi cation scheme that is secure against impersonation under passive attacks. Let S be the signature derived from ID using the Fiat{Shamir transform. Then S is unforgeable under chosen-message attacks in the random oracle model.*

Extending the result above to the quantum random oracle model is an active area of research. Unruh proposed a post-quantum adaptation of the Fiat{Shamir transform in [Unr15], but the cost of applying Unruh's transformation is rather high. More recently, several results have appeared proving the security of the unmodi ed Fiat{Shamir transform under mild assump-

tions [LZ19, DFMS19], but we leave it as an open question to prove similar results for SQISign.

5.1.2 Isogeny-based signature schemes.

We present below an overview of isogeny-based signatures prior to the publication of [DFKL+20]. All known isogeny-based signature schemes are derived from an interactive identification protocol by applying the Fiat-Shamir transform. The soundness security property is mainly based on the hardness of guessing the challenge and this is why the challenge space is the crucial parameter. In all of the schemes that we present below, the challenge space of the underlying sigma protocol has size 2, so we need to repeat it times to obtain a global challenge space of size 2 . The resulting signature is a concatenation of the output of these executions which is not very compact. We start with a quick overview of the solutions based on the kernel representation of isogenies.

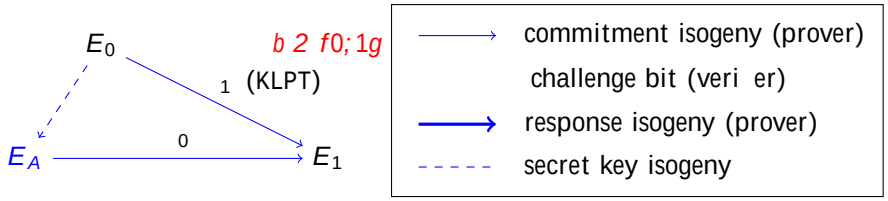
Signature from the kernel representation. The first practical isogeny signature scheme is based on the SIDH key exchange from De Feo and Jao [JD11] (presented in Section 6.1.1). It was derived by Yoo et al. in [YAJ+17] from the identification protocol presented in [DFJP14]. While it benefits from SIDH's efficiency, it is not very compact due to the soundness issue we mentioned above.

Then, SeaSign was introduced by De Feo and Galbraith [DFG19] in the group action setting of the CSIDH protocol. The original idea is due to Stolbunov [Sto10] and was only later reinterpreted in the context of CSIDH by De Feo and Galbraith. Using the additional structure of the group action, SeaSign offers some tradeoff between the size of the keys and the size of the signatures, but it is even less efficient than the signature derived from SIDH. Moreover, the quantum security of all schemes constructed from CSIDH remains unclear (see [BS20, Pei20]). Beullens, Kleinjung and Vercauteren introduced a modification of Seasign called Csi-FiSh [BKV19]. They show that Seasign can be considerably improved once the structure of the class group involved in CSIDH has been computed. In [BKV19], they broke the record of the largest class group computation and applied it to their scheme, thus obtaining an isogeny-based signature scheme with quite decent performances (in terms of size and efficiency). However, the class group computation has sub-exponential complexity, so the parameters of Csi-FiSh cannot be scaled to larger values. This is a major issue because of the security concerns we mentioned above. These three schemes, together with the GPS protocol that we introduce in the next paragraph, form a complete overview of isogeny-based signatures before the introduction of our SQISign construction.

The GPS signature. The protocol presented by Galbraith, Petit and Silva differs from the other solutions because it is the only one based on the Deuring correspondence and the ideal representation. Despite these differences, the concrete scheme has the same generic structure, since it is derived from an identification protocol with challenge space of size two.

Let E_0 be the special extremal curve in $S(p)$ and O_0 the special maximal order in $B_{p;1}$ (Definition 3.1.1). A GPS public key is a curve E_A 2 $S(p)$, and the

Figure 5.1: The GPS identification scheme.



secret key is the endomorphism ring of E_A (equivalently an isogeny between E_0 and E_A). The GPS identification protocol is made from a sigma protocol to prove the knowledge of $\mathrm{End}(E_A)$. The commitment is a curve E_1 computed from E as the codomain of some isogeny $\tau_0 : E_0 \to E_1$. The challenge is a bit $b \in \{0,1\}$. The response is τ_b where τ_1 is computed in the following manner: with τ_0 and $\mathrm{End}(E_A)$ find a maximal order $\mathcal{O}_1$ isomorphic to $\mathrm{End}(E_1)$, apply KLPT on $I(\mathcal{O}_0, \mathcal{O}_1)$ to find an equivalent ideal of powersmooth norm and apply IdealToIsogeny to compute $\tau_1 : E_0 \to E_1$. The situation is depicted in Figure 5.1

As we explained, the resulting GPS signature suffers from a small challenge space. Additionally, IdealToIsogeny is a rather more theoretical than practical algorithm, particularly when required to translate ideals with large norm (which is the case for the output of KLPT as we saw in Proposition 3.2.7). Because of those issues, GPS produces large signatures and was never implemented. Our protocol SQISign address both shortcomings. With the new algorithm Generic-KLPT introduced in Chapter 3, we solve the soundness issue by expanding the challenge space to have exponential size. The efficient algorithms IdealToIsogenyFromKLPT and IdealToIsogenyFromEichler, introduced in Section 4.2.2, target the practical efficiency.

5.2 A new identification protocol and signature scheme

In this section, we give an overview of our new protocol, SQISign. A more precise and concrete description, together with implementation results, will be given in Section 5.3 and Section 5.4. A completely explicit description of SQISign can be found in Section 5.4.3. Following the standard framework introduced in Section 5.1, our signature protocol is obtained from an interactive identification protocol by the Fiat–Shamir transform [FS86]. In Section 5.2.1, we present the underlying identification scheme and some results regarding soundness. Zero-knowledge requires more care, and we will treat it later in Section 5.5. In Section 5.2.2, we introduce the outline of our signature.

5.2.1 An identification protocol

Our new identification scheme is made of a *sigma protocol* to prove the knowledge of the endomorphism ring of a curve E (or equivalently an isogeny between E and a curve E_0 of known endomorphism ring, see Section 2.2). In the description below, we outline the generic principle of the protocol without giving too many details. In particular, we only sketch the response computation. Making

this operation safe and e cient will occupy a good part of this chapter.

Let be a security parameter. The setup is as follows.

setup : 7! param Pick a prime number p and let E_0 be the special extremal supersingular curve over $\mathbb{F}_{p^2}$. Select an odd, smooth, -bit number D_c, and let $D = 2^e$ where e is above the diameter of G_p^2.

keygen : param 7! (pk = E_A; sk =) Pick a random isogeny : E_0 ! E_A of degree N , leading to a random elliptic curve E_A. The public key is E_A, and the secret key is the isogeny .

To prove knowledge of the secret , the prover engages in the following -protocol with the veri er. We summarize this scheme in Figure 5.2.

Commitment The prover generates a random (secret) isogeny walk : E_0 ! E_1, and sends E_1 to the veri er.

Challenge The veri er sends the description of a cyclic isogeny ' : E_1 ! E_2 of degree D_c to the prover.

Response From the isogeny ' ^ : E_A ! E_2, the prover constructs a new isogeny : E_A ! E_2 of degree D such that ^ is cyclic, and sends to the veri er.

Veri cation The veri er accepts if is an isogeny of degree D from E_A to E_2 and ^ is cyclic. They reject otherwise.

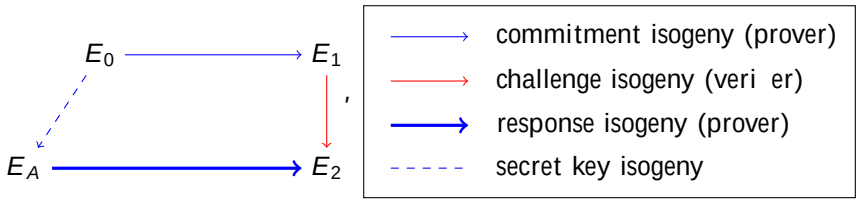


Figure 5.2: A picture of the identi cation protocol

Completeness follows from the correctness of the response computation, which allows the honest prover to construct : E_A ! E_2 such that ^ is cyclic. Special soundness is analyzed below. We will see in Lemma 5.2.1 that it is related to the following relation:

(E_A;) 2 R ( ) is a cyclic smooth degree endomorphism of E_A: (5.2.1)

Lemma 5.2.1. *The sigma protocol introduced above is special sound.*

Proof. It su ces to show that given two accepting conversations (E_1; ';) and (E_1; $'^0$; ^{0}) where ' 6= $'^0$, the composition $\hat{}^0$ $'^0$ ^ is a non-scalar endomorphism of E_A of smooth degree. By construction, $\hat{}^0$ $'^0$ ^ is an endomorphism of E_A of degree $(DD_c)^2$. This shows that the degree is smooth. It remains to prove that it is not a scalar. Suppose by contradiction that $\hat{}^0$ $'^0$ ^ = [DD_c]. The compositions ^ and $\hat{}^0$ ^{0} are two cyclic isogenies from E_A to E_1 of the same degree. Therefore, $\hat{}^0$ $'^0$ is the dual of ^ . We deduce that ^ = $\hat{}^0$ ^{0}, a contradiction. □

With the Fiat-Shamir transform, we can derive a secure signature scheme from an identification protocol that is complete, special-sound and honest verifier zero-knowledge when the special soundness is based on a relation R that is hard, i.e., for which it is hard to compute a witness from an element of the language L.

The relation R introduced in Eq. (5.2.1) is hard under the hardness of the Smooth Endomorphism Problem | a problem heuristically equivalent to the classic Endomorphism Ring Problem (see Remark 2.2.3).

Problem 5.2.2 (Smooth Endomorphism Problem (SEP)). *Given a curve $E \in S(p)$, find a (non-trivial) cyclic endomorphism of E of smooth degree.*

Response computation and the zero-knowledge property. The sketch given in Section 5.2.1 is incomplete, as it does not specify a method to compute the response isogeny . Imitating the approach from [GPS17], our proposed solution is to use the algorithmic tools obtained from the Deuring correspondence.

The response computation is divided into two main steps. First, we compute the response as an ideal using a variant of GenericKLPT (see Algorithm 11 in Section 3.3). This is the purpose of the SigningKLPT algorithm that we will introduce as Algorithm 33 in Section 5.3.1 (the difference between these two algorithms is motivated purely by security considerations). Then, the response isogeny can be computed as the translation of the output of SigningKLPT with the efficient ideal-to-isogeny algorithms introduced in Section 4.2.2. In particular, IdealToIsogenyFromKLPT was introduced in our paper [DFKL+20] to perform this operation efficiently. The more efficient IdealToIsogenyFromEichler was introduced in [DFLW22] as an improvement of IdealToIsogenyFromKLPT.

The zero-knowledge property of our identification scheme crucially depends on the approach employed to perform the computation of the response. We justify this briefly by exhibiting bad ways of doing it. The obvious method would be to reveal the isogeny $= ' \quad \hat{}$. This is, of course, a valid answer, but it directly reveals the secret . We might also be tempted to apply the method from [KLPT14] (described at the end of Section 3.2.3) instead of our adaptation of GenericKLPT. However, by design, the isogeny corresponding to an ideal obtained in this manner goes through the special extremal curve E_0, and so reveals the secret. This issue is in fact what motivated us to introduce GenericKLPT in [DFKL+20]. Arguing that our algorithm SigningKLPT does not lead to a similar leakage will be at the heart of the analysis we provide in Section 5.5 and Section 5.6. We will prove zero-knowledge in Section 5.5, under a new conjecturally hard computational problem introduced as Problem 5.5.6.

5.2.2 The signature scheme

The new signature scheme is simply a Fiat{Shamir transformation of the identification protocol introduced in Section 5.2.1. The main part of this transformation is to compute the challenge as some hash of the message and the commitment. In our case, this operation is not entirely trivial because the challenge is an isogeny of degree D_c. We propose composing the $\text{Decompression}_{D_c}$ algorithm introduced in Section 4.1.5 with a standard hash function. For any degree D_c, the set of cyclic isogenies of degree D_c corresponds with $\{0,1\}^{(D_c)}$

for some (D_c) under our compression method. Let $H : f0;1g \to f0;1g^{(D_c)}$ be a cryptographically secure hash function.

sign : (SK; m) 7! Pick a random (secret) isogeny $: E_0 \to E_1$. Let $s = H(j(E_1); m)$, and build the isogeny Decompression$_{D_c}(E_1; s) = ' : E_1 \to E_2$. From the knowledge of O_A, and of the isogeny $' : E_0 \to E_2$, construct an isogeny $: E_A \to E_2$ of degree D such that ^ is cyclic. The signature is the pair $(E_1;)$.

verify : (pk; m;) 7! true or false Parse as $(E_1;)$. From $s = H(j(E_1); m)$, recover the isogeny Decompression$_{D_c}(E_1; s) = ' : E_1 \to E_2$. Check that is an isogeny from E_A to E_2 and that ^ is cyclic.

Theorem 5.2.3. *The signature described above is secure against chosen-message attacks in the random oracle model assuming the hardness of Problems 5.2.2 and 5.5.6.*

Proof. This follows from Theorem 5.1.2 applied to the identi cation scheme described in Section 5.2.1. The associated sigma-protocol is complete as explained brie y in Section 5.2.1, special sound due to Lemma 5.2.1, and honest veri er zero-knowledge by combining Lemma 5.5.1 with Proposition 5.5.10. □

5.3 Concrete instantiation: security

In this section and the next, we ll the gaps left open in Section 5.2 and explain in detail how to instantiate our new signature scheme. Below, we are interested in security issues. Most of this section is spent on the SigningKLPT algorithm which computes the ideal corresponding to the response isogeny, but we will also talk about key generation.

5.3.1 Computation of the response isogeny: a secure algorithm to compute the ideal

In this section, we describe the SigningKLPT procedure (Algorithm 33) used in our signature scheme. This algorithm is a variant of GenericKLPT (presented in Section 3.3). We are going to apply SigningKLPT in the setting 2 to nd an ideal with norm a power of 2. There are two main di erences between SigningKLPT and GenericKLPT, and both stem from security considerations. First, we would like to have output of constant degree (so it does not depend on the secret). We will write SigningKLPT$_{2^e}$ to indicate that 2^e is the targeted norm of the ideal. Second, we need some randomization to ensure a good distribution. To that end, we introduce RandomEquivalentEichlerIdeal (Algorithm 34), used in Step 1 of SigningKLPT. We will motivate these changes later in this section.

First, we give a detailed description of SigningKLPT. As usual, O_0 is the special extremal order of $B_{p;1}$. SigningKLPT takes two cyclic ideals as input: I of norm N and right order O_1, corresponding to the secret isogeny , and I a cyclic O_1-ideal. The output $J \subset I$ of norm 2^e corresponds to the desired response isogeny in our protocol. Let us write $O = O_0 \setminus O_1 = Z + I$. The order O is an Eichler order of level N.

Let us explain brie y how we intend to deal with the xed norm constraint. As we will see in Section 5.5, it is important for security to deal with this issue

in the right manner. A good part of SigningKLPT (from Step 3 to Step 9) is very similar to IdealEichlerNorm. The goal of these steps is to find an element of norm 2^e in the ideal $O \setminus L$ for some ideal L of norm N such that $O \setminus I$ is equivalent to $O \setminus L$ as ideals of Eichler orders. The norm of the output of IdealEichlerNorm has two main factors: , the output of FullRepresentInteger in Step 3, has norm $N2^{e_0}$ and , the output of FullStrongApproximation in Step 8, has norm 2^{e_1}. We therefore have a natural decomposition $e = e_0 + e_1$. Since must be computed before , the value of e_0 uniquely determines the value of e_1. We will sample e_0 at random from a set $e_0(N)$ whose exact definition is given in Proposition 5.3.5 and mainly follows from Lemma 3.1.4 to ensure termination of FullRepresentInteger. The estimates from Lemma 3.1.6 let us determine what size we can take for e in order to ensure termination with a given probability.

Algorithm 33 SigningKLPT$_{2^e}(I\ ;I)$

Input: I a left O_0-ideal and right O_1-ideal of prime norm N inert in O, and I, a left O_1-ideal.
Output: J I of norm 2^e.

1: Compute K = RandomEquivalentEichlerIdeal($I; N$)
2: Compute K^0 = [I] K and set L = EquivalentPrimeIdeal(K^0), L = $_{K^0}$( ) for $2\ K^0$ with $N = n(L)$. Sample $e_0\ 2\ e_0(N)$.
3: Compute = FullRepresentInteger$_{N\,D(2^{e_0})}$.
4: Compute $(C_0 : D_0)$ = IdealModConstraint($L;$).
5: Compute $(C_1 : D_1)$ = EichlerModConstraint(I ; ;).
6: Compute C = CRT$_{N\ ;N}(C_0; C_1)$ and D = CRT$_{N\ ;N}(D_0; D_1)$.
7: e_0 $v_2(n($)) and e_1 e e_0.
8: Compute = FullStrongApproximation$_{f2^{e_1}g}(NN\ ; C; D)$. If it fails, go back to Step 3.
9: Set = .
10: **return** J = [I] $_L$( ).

The randomization procedure. Step 1 of SigningKLPT is to perform a randomization step which we will use in the security argument for our signature. This additional step has two interesting consequences for us. First, the output of Algorithm 33 only depends on the equivalence class of the input I. Second, it randomizes the execution.

RandomEquivalentEichlerIdeal takse a cyclic O_1-ideal I as input, and returns an equivalent random ideal, meaning (in this context) that if we write C for the class of I in Cl(O_1), then we want an output ideal equivalent to I and lying in a uniformly random class of Cl$_O$ (C) (see Definition 2.3.10). This condition might seem a little arbitrary at first; however, Proposition 5.3.3 will show that this is exactly the kind of randomness we need.

To reach our goal, we use the classical technique of finding some well-chosen $2\ I$ and outputting $_I$( ). The method to choose is inspired by the results of Section 2.3.1. The idea is to use the bijection from Proposition 2.3.12 in order to sample a class uniformly. Note that Proposition 2.3.12 does not hold for some special maximal orders O, but we may assume that this is not the case here (there are at most two such types of maximal orders among $O(p)$ possibilities).

Algorithm 34 RandomEquivalentEichlerIdeal($I; N$)

Input: I a left O_1-ideal.
Output: K I of norm coprime to N.

1: Sample a random element $!_S$ in O_1 until N is inert in $\mathbb{Z}[!_S]$.
2: Sample a random element in I such that $n( )/n(I)$ is coprime to N.
3: Select a random class $(C : D)$ 2 $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$.
4: Set $= (C + !_S D)$.
5: **return** $K = {}_I( )$

We now show that Algorithm 34 terminates and that its output distribution is correct.

Lemma 5.3.1. *Algorithm 34 terminates in expected* $O(\text{poly}(\log(pN\ n(I))))$ *and outputs an ideal equivalent to* I *and uniformly distributed among the* $N + 1$ *possible classes of* $\text{Cl}_O(C)$ *where* C *is the class of* I *in* $\text{Cl}(O_1)$.

Proof. Since $O = O_1 \setminus O_0$ has level N, we can represent a basis of O with coefficients of size in $O(\text{poly}(\log(pN\ )))$. We can find a quadratic suborder $\mathbb{Z}[!_S]$ O in which N is inert in $O(1)$ attempts. Then, it is clear that taking a random element in I will verify that $n( )/n(I)$ is coprime to N with overwhelming probability. The elements in I can be represented with coefficients of size $O(\text{poly}(\log(pN\ n(I))))$ and then we have $C; D$ smaller than N. Thus, the expected running time in $O(\text{poly}(\log(pn(I)N\ )))$ follows from the complexity of the operations over $B_{p;1}$.

The algorithm concretely instantiates the map from Proposition 2.3.12. This map is bijective, and if we choose $(C : D)$ uniformly at random from $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$ then the output is uniformly distributed. □

Consequently, the output of RandomEquivalentEichlerIdeal only depends on the class (inside $\text{Cl}(O)$) of the ideal in input. The call to RandomEquivalentEichlerIdeal in Step 1 of SigningKLPT thus implies the following lemma that will prove useful in Section 5.5.

Lemma 5.3.2. *For any* I, *the output distributions of* SigningKLPT($I; I$) *and* SigningKLPT($J; I$) *are the same for any* I J. *Put otherwise, for fixed* I, *the output distribution of Algorithm 33 only depends on the equivalence class of the input ideal* I.

Next, we describe how the distribution of L (as defined in Step 2 of Algorithm 33) is determined by the output distribution of RandomEquivalentEichlerIdeal. This is what motivates the current formulation of Algorithm 34.

Proposition 5.3.3. *The set* $G_I = \{L; L = \text{EquivalentPrimeIdeal}([I\ ]\ K)$ *for* K $I\}$ *has size at most* $N + 1$ *and for every* L 2 G_I *there exists an output* $K = \text{RandomEquivalentEichlerIdeal}(I)$ *such that* $L = \text{EquivalentPrimeIdeal}([I\ ]\ K)$. *When* $\#G_I = N + 1$, *the ideal* L *is uniformly distributed inside this set.*

Proof. As we mentioned before, there are exactly $N + 1$ classes for K I in $\text{Cl}_O(O)$. By Corollary 2.3.6[1], the class of K in $\text{Cl}_O(O)$ uniquely determines the class of $[I\ ]\ K$ in $\text{Cl}(O_0)$. As noted in Remark 3.2.8, the output

[1] Corollary 2.3.6 uses pushforwards rather than pullbacks, but we obtain the desired result by replacing I with $\overline{I}$.

of EquivalentPrimeIdeal is well-de ned and deterministic on $\mathrm{Cl}(\mathcal{O}_0)$. The result follows from this remark with Lemma 5.3.1. □

Remark 5.3.4. In full generality, we cannot prove more than the $N + 1$ upper bound of Proposition 5.3.3. However, in most cases, this number is exactly equal to $N + 1$. To estimate the di erence, we need to count the number of times when $[I] K_1 \quad [I] K_2$ for K_1 and K_2 lying in di erent classes of $\mathrm{Cl}_{\mathcal{O}}(\mathcal{O})$. Rewriting this in our commutative diagram (recall that the norms of K_1 and K_2 are coprime to N) we have $[I] K_1 \quad [I] K_2$ if and only if $[K_1]\overline{I} \quad [K_2]\overline{I}$. Thus, each of the $N + 1$ classes of $\mathrm{Cl}_{\mathcal{O}}(\mathcal{O})$ that we consider is mapped to one left $\mathcal{O}_R(I)$-ideal of norm N through $K \mapsto [K]\overline{I}$. Hence, we want to estimate the number of pairs of distinct equivalent ideals of norm N. In general, if $N + 1$ is small compared to p, then a maximal order has a very low probability of having two distinct equivalent ideals of same norm N, which means that with high probability there are exactly $N + 1$ classes. In any case, the number of possible equivalence classes is in (N).

Termination and complexity. We can now state the complexity of SigningKLPT.

Proposition 5.3.5. *(UPHA) For any* > 0*, there exist* $_1$, $_2$*, and* $\varepsilon = O(\log\log(p) + \log(\ ))$ *such that if* $\varepsilon > 1$*,* $e_0(N) = [\log(p) \quad \log(N) + \ _1; \log(p) \quad \log(N) + \ _1 + \varepsilon]$ *and* $e > 3\log(p) + 3\log(N) + \ _2 + \varepsilon$*, then* $\mathrm{SigningKLPT}_{2^e}$ *will succeed with probability higher than* $1 \quad 2$ *when* I *is in a random class of* $\mathrm{Cl}(\mathcal{O}_1)$*. The expected running time is in* $O(\mathrm{poly}(\log(pN\ n(I))))$*.*

Proof. In a manner similar to GenericKLPT, this result follows from Proposition 3.3.6 and Lemmas 3.1.4, 3.1.6 and 5.3.1, together with estimates from Lemma 3.2.3 for the size of N. □

Remark 5.3.6. The role of ε in Proposition 5.3.5 might appear mysterious, but it will prove useful later in Section 5.6 with Proposition 5.6.8. The condition $\varepsilon > 1$ is there to ensure that $e_0(N)$ contains at least one integral element. For our purposes, any $\varepsilon > 1 + 2\ _0$ (where $_0$ is the term from Lemma 3.2.3 and Remark 3.2.4) will work. This last constraint is also justi ed by Proposition 5.6.8.

Remark 5.3.7. If we take $= \log(p)$, we get that $\mathrm{SigningKLPT}_{2^e}$ succeeds with overwhelming probability. Thus, there exist two minimal values $_1; \ _2 = O(\log\log(p))$ such that Proposition 5.3.5 holds. Henceforth, we assume that, for a given p, these two values $_1; \ _2$ are xed and used to determine the exponent e. We also assume that a value $\varepsilon > 1 + 2\ _0$ is xed as explained in Remark 5.3.6, we get that the set $e_0(N)$ is fully determined by the value of p.

Remark 5.3.8. In our practical instantiation, executions of KLPT, SigningKLPT and all the algorithms from Chapter 3 are quite negligible compared to other operations (in particular the computations involved in the manipulation of the kernel representation of isogenies). This is why we have not spent much time trying to optimize these algorithms.

The security of the scheme resulting from SigningKLPT will be analyzed in Section 5.5 and Section 5.6. In particular, we will motivate the choices and modi cations underlying the formulation of SigningKLPT.

5.3.2 De ning the key space

We refer to the description from Section 5.2.2 for the notation in this section.

For statistical security, the degree N of the secret isogeny should be su ciently large to ensure a near-uniform distribution of the public key E_A over the set of supersingular curves. However, there is another constraint on N : the ideal I corresponding to is used in SigningKLPT, so we want N to be a prime inert in O (see the description of SigningKLPT). Proposition 5.3.5 proves that the size of N has an impact on the size of the output, and so on the global e ciency of our signature scheme. Thus, e ciency demands that we take N as small as possible.

In this section, we discuss a key sampling method which trades o statistical security for e ciency. We will use this alternative key space in our implementation. In this analysis, we focus exclusively on key-recovery attacks that use only the key only.

A natural method to generate the key would be as follows: x a bound B , then sample a prime degree N inert in O randomly in $[2; B]$ and - nally sample a random isogeny of degree N from E_0. With the estimates of Lemma 3.2.3, we see that an overwhelming proportion of the supersingular graph can be reached with this method if B is at least $p\overline{p}$. When E_A is nearly uniformly distributed, the best known classical attack to recover the key is due to Delfs and Galbraith [DG16] and has cost $\tilde{O}(p^{1=2})$ (it has the same complexity as the algorithm outlined in Section 2.2.1, but does not rely on any assumptions). It consists in performing a random walk from E_A until a curve E^0 over $\mathbb{F}_p$ is reached, then doing a search in the $\mathbb{F}_p$-graph for an isogeny connecting E^0 to E_0. The quantum version of this algorithm achieves a quadratic speed-up using Grover's algorithm, and thus has cost $O(p^{1=4})$ [BJS14]. Hence, for a classical security level , and quantum security level = =2, it is enough to choose $\log(p) = 2 = 4$. In particular, we estimate that $\log(p) = 256$, 384 and 512 reach the NIST's security levels 1, 3 and 5 respectively.

A simple way to improve the e ciency of our protocol is to decrease the bound B . This is reminiscent of the SIDH protocol [JD11], in which only a small fraction of the supersingular graph is used, and whose security is consequently not amenable to a generic isogeny problem. However, unlike in the SIDH case, slightly reducing the key space does not improve the cost of known attacks. Indeed, in our protocols N is a large prime, thus meet-in-the-middle strategies *a la* [ACVCD+19] would be ine ective.

When B is small enough, exhaustive search becomes the best strategy: compute all isogenies of degree smaller than B , and compare their codomain curve with E_A. All of the isogenies can be computed in polynomial time in $\log(p)$, even if N is not smooth, because we can translate the ideal into a smooth-degree one with KLPT. Since there are $\tilde{}(B)$ possible degrees N and (N) cyclic isogenies for each of these degrees, the classical complexity of this attack is in $\tilde{}(B^2)$, and Grover's algorithm yields again a quadratic speed-up at best. To defeat this attack, we only need $\log(B) > \frac{1}{4}\log(p)$ which is better than the $\log(N) > \frac{1}{2}\log(p)$ bound that we have in general.

This improvement produces a shorter and more e cient signature for the same level of security, as it reduces the output size of Algorithm 33 from $\frac{9}{2}\log_2(p)$ to $\frac{15}{4}\log_2(p)$ (see Proposition 5.3.5). We use it in the implementations presented in Section 5.4.4.

5.4 Concrete instantiation: e ciency

In this section, we discuss e ciency considerations for a concrete instantiation of SQISign. Together with the results from Section 5.3, we obtain a complete and detailed description of SQISign that we provide in Section 5.4.3. This leads to an implementation in C whose performances we report in Section 5.4.4. In what follows, we target the NIST level 1 of security and 128-bits of classical security. Following the discussion in Section 5.3.2, this means that we will take a prime p with $\log(p)$ 256.

The main issue that remains to be dealt with is the ideal to isogeny translation to derive the response from the output of SigningKLPT. Fortunately, we already have all the necessary tools. Indeed, the IdealToIsogenyFromKLPT and IdealToIsogenyFromEichler algorithms introduced in Section 4.2.2 were designed speci cally to perform that task e ciently. Despite the amount of e ort we spent on making these algorithms e cient, we will see in Section 5.4.4 that this is, by far, the bottleneck of the computation. In comparison, the other steps such as SigningKLPT are completely negligible. This is why we have spent little time discussing the concrete e ciency of this algorithm.

We already explained in Section 4.2.2 that IdealToIsogenyFromEichler should be considered an improvement of IdealToIsogenyFromKLPT, and we justify this below with a detailed cost analysis in Section 5.4.1. This will be con rmed in Section 5.4.4 by our implementation of the two methods. In Section 5.4.2, we explain how to nd good parameters to instantiate our translation algorithms.

5.4.1 Ideal to isogeny: cost estimate

We will observe in Section 5.4.4 that algebraic operations over $\mathbb{F}_{p^2}$ make up most of the cost of SQISign: up to 90% in our experiments. These mainly come from isogeny computations, and T-isogenies in particular. It is thus reasonable to ignore, in the analysis below, the computations over the quaternions and the linear algebra required by the algorithms from Chapter 3. Ideally we would count the number of $\mathbb{F}_{p^2}$-operations performed for each choice of parameters, however this is di cult given the complexity of the algorithms. Instead, we will use a much coarser metric based on four indicators.

We are only going to compare IdealToIsogenyFromKLPT, and IdealToIsogenyFromEichler, for a small prime ℓ (in practice, $\ell = 2$). A full description of these algorithms was given in Section 4.2.2 but we give a brief reminder below. Both algorithms decompose an ideal of norm ℓ^e into ideals of smaller norm. The former decomposes into ideals of norm ℓ^{2f+} for some constant , which are then translated to isogenies by IdealToIsogenySmallFromKLPT. The latter decomposes into ideals of norm ℓ^f, which are translated by IdealToIsogenySmallFromEichler.

The translations are achieved using T-isogenies where T is a smooth integer coprime to ℓ. Even if we use the same notation for this parameter, the two methods do not have the same requirements: we need $T > p^{3=2}$ for IdealToIsogenyFromKLPT, and $T > p^{5=4}$ for IdealToIsogenyFromEichler. The fact that this constraint is relaxed for IdealToIsogenyFromEichler explains why this version will end up being more e cient. Both sub-algorithms consist mostly of isogeny computations of degree T and ℓ^f. For each of them, we will count:

(T_c) How many isogenies of degree T are *computed*;

(T_e) At how many points the isogenies of degree T are *evaluated*;

(ℓ_c) How many isogenies of degree 2^f are computed/evaluated;

($\_{c}$) How many meet-in-the-middle searches for isogenies of degree ℓ are performed (this is exclusive to IdealToIsogenySmallFromKLPT).

The costs of T_c and T_e depend on the factorization of T. Instead of using the full factorization, we will base our estimate on a bound B such that all prime factors of T are $< B$. Using the method introduced in Section 4.1, the costs of computing and evaluating an isogeny of prime degree n grow with $\sqrt{n}$ (ignoring logarithmic factors), we will thus multiply T_c and T_e by $\sqrt{B}$. Since ℓ is small, the cost of computing and evaluating an isogeny of degree ℓ^f grows with $f\log(f)$ (ignoring the dependency in ℓ), we shall thus multiply ℓ_c by this factor. Finally, the meet-in-the-middle requires computing all $\sqrt{\ell^{\_}}$ isogenies, so we multiply $\_{c}$ by $\sqrt{\ell^{\_}}$.

Given an ideal of norm ℓ^e, IdealToIsogenyFromKLPT calls IdealToIsogenySmallFromKLPT $e/(2f+\_)$ times, while IdealToIsogenyFromEichler calls IdealToIsogenySmallFromEichler e/f times. For this reason, we divide all counts by $2f+\_$ and f, respectively.

Summarizing, for the method based on KLPT we will use the following 4-valued estimator:

$$(T_c\sqrt{B}\,;\ T_e\sqrt{B}\,;\ \ell_c f\log(f)\,;\ \sqrt{\ell^{\_}}\_{c})/(2f+\_)\,; \tag{5.4.1}$$

where the division is applied component-wise. For the method based on SpecialEichlerNorm, given that it does not use a meet-in-the-middle search, we will instead use

$$(T_c\sqrt{B/f}\,;\ T_e\sqrt{B/f}\,;\ \ell_c\log(f))\,: \tag{5.4.2}$$

We give below the estimation count for each of the operations. We will compute the exact value of the estimators for concrete examples of prime characteristic p in Section 5.4.2.

First Method. Some steps in IdealToIsogenySmallFromKLPT are rather vague, so we refer to the code at `https://github.com/SQISign/sqisign` to see how it is done in practice.

The operation count for IdealToIsogenySmallFromKLPT is as follows: Step 3 is 2 T_e (push $\ker\theta_1$ through θ_J) and 1 ℓ_c (compute θ_1), Step 8 is 1 T_c (compute $\_{1}$), Step 9 is 1 T_e, 1 T_c (compute $\_{2}$ and $\ker\_{2}$) and 1 ℓ_c (compute θ_2), Step 10 is 1 $\_{c}$, Step 11 is 2 T_e (compute $\ker\hat{}_1$), 2 ℓ_c (push $\ker\hat{}_1$ through $\_{2}$), 1 T_c and 1 T_e (compute $\_{1}^{0}$ and $\ker\hat{}_2$), 1 ℓ_c (compute θ_2) and 1 $\_{c}$ (compute $\_$). The total is 3 T_c, 6 T_e, 2 $\_{c}$ and 5 ℓ_c.

Second Method. Step 7 of IdealToIsogenySmallFromEichler requires solving a DLP instance in the ℓ^f-torsion, and we overestimate the complexity by saying that this is equivalent to 1 ℓ_c operation (asymptotically it is the same cost, but the DLP is faster in practice). We obtain the following count: Step 6 is 2 T_c, Step 7 is 5 T_e and 1 ℓ_c (see Algorithm 26), and Step 8 is 1 ℓ_c. Overall, we get 2 T_c, 5 T_e and 2 ℓ_c.

5.4.2 Parameter choices

The choice $\ell = 2$ yields the fastest verification possible so this is the value we take for our concrete instantiation of SQISign. For the fastest and simplest instantiation of our algorithms we want $E[T2^f]$ to be defined over $\mathbb{F}_{p^2}$ so we need $T2^f \mid p^2 - 1$. Since we have the constraint $T > p$ for both IdealToIsogenyFromEichler and IdealToIsogenyFromKLPT, we cannot hope to choose a good T and exponent f before finding the prime p. Thus, we will have to search through many primes until we find one with the good properties. In this section, we describe how to speed up this search and give examples with $\log(p) \approx 256$. These examples are the ones we used in our implementation to get the results presented in Section 5.4.4.

Remark 5.4.1. In each case, we target the smallest possible size for T and evaluate the quality of each prime by the smoothness of T. Efficiency clearly depends on the size of f. A bigger f means fewer executions of IdealToIsogenySmallFromKLPT and IdealToIsogenySmallFromEichler but a larger smoothness bound for T. Given the difficulty to estimate the exact complexity of the computation, we decided to target a fixed value of f. Based on some rough estimates, we chose $f \approx 32$ for IdealToIsogenyFromKLPT and $f \approx 64$ for IdealToIsogenyFromEichler; more work would be required to determine a finer choice.

Remark 5.4.2. For security, only the size of p is important (to ensure hardness of the ERP). In isogeny-based cryptography, primes are always chosen in very specific forms for better efficiency: for instance, SIDH takes $p = c2^{e_A}3^{e_B} - 1$ (see Section 6.1.1).

Search methods. Finding primes p such that $p^2 - 1$ has a smooth factor considerably larger than p is a difficult task because we need to control both $p - 1$ and $p + 1$ simultaneously. This problem was recently considered in the context of the SIDH-like key exchange B-SIDH [Cos19], where the focus is on finding p such that $p^2 - 1$ is fully smooth. Here, we have a slightly different problem, as we only need $p^2 - 1$ to contain a large enough smooth factor; in addition, we want it to be divisible by a large power of 2. We have explored two main approaches to find such p.

We call the first approach the *XGCD* method. It is rather simple, but it appears to be well-suited to our task as this is the one that gave the better primes. The idea is to construct primes p such that

$$p - 1 = 2^a \quad A;$$
$$p - 1 = \quad B;$$

where

- $a + \log_2(\quad) \approx f$, where f is our target for the power of 2-torsion,
- $\quad ;\quad$ are odd B_1-smooth for some bound B_1, and
- there is a B_2-smooth $\quad \mid AB$ such that $\log_2(\quad) > t$ for some threshold t.

We construct these by choosing a, $\quad$ and $\quad$ before using the Chinese remainder theorem to reconstruct p, and then testing the primality of p and the smoothness of AB.

We use two tricks to increase the probability of success. First, to increase the probability that p is prime, we always include some small factors in or , namely 3; 5; 7 and 11. Second, and most importantly, to increase the probability that AB contains a large smooth factor we observe that we have freedom in the choice of $\log_2(A)$ and $\log_2(B)$, as long as $\log_2(AB)$ has the right size, and that the probability of having a large smooth factor jAB is not maximized by $\log_2(A)$ $\log_2(B)$ unless $= AB$.

The second approach is to look for primes of the form $p = 2^f x^n$ 1 where n is a xed exponent and x will cover numbers of size $(\log(p)$ $f)/n$. This idea was introduced in [Cos19] to nd B-SIDH friendly primes, but it can be adapted to our setting quite easily. The problem here is that the search space ends up being quite reduced as soon as n grows.

Remark 5.4.3. An earlier version of [Cos19] explored the possibility of using St rmer's theorem [Ste03] for this search, but this theorem does not exactly match our needs; further, a recent update to [Cos19] reports that St rmer's theorem does not seem to produce good results for meaningful sizes. The methods of the recent work [CMN21] really target the case of B-SIDH and is not well suited to produce primes such that p^2 1 contains a large power of 2, or of any other small prime.

Concrete searches. The requirement on T in IdealToIsogenyFromKLPT comes from the KLPT algorithm. Given Proposition 3.2.7, we decided to look for T with $t = \log(T) > 390$ (in practice, this choice appears to be large enough for KLPT to succeed with overwhelming probability). We applied the XGCD method, with the following parameters: $a = 32$, $B_1 = 2^{10}$ and $B_2 = 2^{14}$. Using the probability estimates of [BS07], we found that the smoothness probability is maximized by taking $\log_2(B) = 87$, and thus $\log_2(\ ) \approx 168$, $\log_2(\ ) = 56$ and $\log_2(A) = 168$. We xed $= 5^{21}$ 7 11, and $= 3^b$ $\prod_{i\ i}$ where the number of B_1-smooth integers in is chosen to guarantee a large enough search space.

We implemented this strategy in C using the GMP library. A search e ort of about 6 CPU-years produced several useful primes. The most interesting one, called p_{6983}, has

$$p_{6983} + 1 = 2^{33}\ 5^{21}\ 7^2\ 11\ 31\ 83\ 107\ 137\ 751\ 827\ 3691\ 4019\ 6983\ 517434778561\ 26602537156291;$$

$$p_{6983}\ \ 1 = 2\ 3^{53}\ 43\ 103\ 109\ 199\ 227\ 419\ 491\ 569\ 631\ 677\ 857\ 859\ 883\ 1019\ 2713\ 4283:$$

We used this prime in our implementation. (At the time when we published [DFKL+20], we were not aware of the second method from [Cos19] and so we did not try it.)

For IdealToIsogenyFromEichler, we follow Proposition 3.3.4 and look for $t > 330$ (this value appears to be large enough in practice to ensure success with overwhelming probability). We used the method of [Cos19] to look for primes $p = 2^{61}x^4$ 1, sieving the whole interval x 2 $[2^{47}; 2^{49}[$ in approximately 360 CPU-days. We found 398 integers such that p^2 1 has a 2^{11}-smooth odd factor of more than 330 bits, of which 15 were prime (see Table 5.1); none of them has a large enough 2^{10}-smooth factor.

143189100303149	369428710635531	391443251922757	411099446409699
424067696488337	431716591494287	491224940548057	491531434028942
512391149388477	512583833108361	514414280000642	515727186701509
548396183941255	550470785518701	562456538440551	

Table 5.1: The list of integers x 2 $[2^{47}; 2^{49}[$ such that $2^{61}x^4$ 1 is prime and $x^4(2^{15}x$ 1)$(2^{15}x + 1)(2^{30}x^2 + 1)$ contains a 2^{11}-factor $> 2^{330}$.

Using the XGCD method, we found that we could obtain primes with f 64 and $B = 2^{12}$ at a reasonable cost. The best candidate we found, which we name p_{3923}, has 254 bits and

$$p + 1 = 2^{65} \; 5^2 \; 7 \; 11 \; 19 \; 29^2 \; 37^2 \; 47 \; 197 \; 263 \; 281 \; 461 \; 521$$
$$3923 \; 62731 \; 96362257 \; 3924006112952623;$$
$$p \quad 1 = 2 \; 3^{65} \; 13 \; 17 \; 43 \; 79 \; 157 \; 239 \; 271 \; 283 \; 307 \; 563 \; 599$$
$$607 \; 619 \; 743 \; 827 \; 941 \; 2357 \; 10069:$$

Despite the slightly larger smoothness bound, we found that p_{3923} performs better in practice than primes of the form $2^{61}x^4$ 1, probably owing to the large power of 3, which contributes favorably to T-isogeny computations.

Looking at the estimator values for p_{3923} in Table 5.2, we see that applying IdealToIsogenyFromEichler to this prime yields a signi cant gain during T-isogeny computations and meet-in-the-middle at the cost of a modest loss during 2^f-isogeny computations. Since the former tends to a ect performance much more than the latter, in practice, we expect IdealToIsogenyFromEichler to compare favorably to IdealToIsogenyFromKLPT. We also considered using p_{6983} with IdealToIsogenyFromEichler as a way to compare the two methods on the same prime. The estimators in Table 5.2 tends to prove that even for p_{6983}, which was selected for IdealToIsogenyFromKLPT, the better performances are obtained with IdealToIsogenyFromEichler. This will be con rmed by experiments.

In fact, we will see in Section 5.4.4 that in practice, the gain of IdealToIsogenyFromEichler is even larger than predicted by our estimator. Finding more accurate estimators to guide the prime search in SQISign is an interesting problem for future research.

algorithm	p	$\log(p)$	f	B	T_c	T_e	$`_c$	$_c$	estimator
IdealToIsogenyFromKLPT	p_{6983}	256	33	2^{13}	3	6	5	2	(3:4; 6:8; 10:4; 3:2)
IdealToIsogenyFromEichler	p_{6983}	256	33	2^{11}	2	5	2	{	(2:7; 6:9; 10:1)
IdealToIsogenyFromEichler	p_{3923}	254	65	2^{12}	2	5	2	{	(2:0; 4:9; 12:0)

Table 5.2: Operation estimates for several variants of ideal-to-isogeny translation. B is the smoothness bound of T.

5.4.3 SQISign: the concrete description

We now give a detailed description of all the steps composing the identi cation scheme for SQISign. We target = 128 bits of classical security.

Given the discussion in Section 5.4.1 and Section 5.4.2, we transform ideals to isogenies using IdealToIsogenyFromEichler. We use the prime p_{3923} de ned in Section 5.4.2. Tecall that $T \quad p^{5=4}$. We have $2^f = 2^{65}$ available torsion.

Experiments suggest that taking $D = 2^{1000}$ for the degree of the response isogeny su ces to ensure the success of SigningKLPT with overwhelming probability. This choice is not necessarily the tightest possible, but it is close enough.

We will use the challenge and commitment isogenies to compute the endomorphism ring of the curve E_2, so we need to be able to translate them into their corresponding ideals e ciently. Let us write D_c for the challenge degree and T^0 for the commitment degree. We will see that we can have $T^0 D_c j T 2^f$. For faster veri cation, we want the challenge isogeny computation to be as fast as possible, and this is why we use all the smallest factors in D_c.

Building (keygen). As explained in Section 5.3.2, we select the degree N of the secret isogeny to be a prime smaller than $2^{=2} = 2^{64}$ and inert in O, chosen uniformly at random among such numbers. Since N is a large prime, we never compute the isogeny concretely, as this would be too ine cient. Instead, we use the corresponding ideal I . This is enough to apply SigningKLPT but it does not give us the public key E_A. For this, we compute another isogeny $^0 : E_0 \; ! \; E_A$ of degree in 2 . We can obtain I_0 using KLPT and compute ^{0} with IdealToIsogenyFromEichler. We present an alternative and more e cient key generation procedure at the end of this section.

We do the keygen as follows:

1. Select a prime $N \quad B$ inert in O uniformly at random.
2. Select a left O_0-ideal I of norm N uniformly at random among the $N + 1$ possibilities.
3. Compute $J = \mathrm{KLPT}_2(I)$
4. Compute $^0 = \mathrm{IdealToIsogenyFromEichler}(J ; O_0; [1]_{E_0})$ and set pk = E_A, the codomain of ^{0}.

Building (commitment). There are several options for building the commitment (and, incidentally, the challenge); we present the most e cient option here. We note that, for security reasons, must be as hard to recover as the secret. This suggests taking a smooth isogeny of degree about p (here we do not gain anything by using the same idea as in Section 5.3.2). Given the factorization $2^f T = D_c \; T^0$, we choose as a random isogeny of degree T^0 from E_0. The exact value for D_c is given in the next paragraph. With this choice of T^0, computing the isogeny and converting it to the ideal I is e cient with IsogenyToIdeal.

Building ′ (challenge). The previous choice of commitment generation was motivated by the fact that we want an e cient way to translate the challenge into its corresponding ideal. For -bit soundness security we need a challenge space of size 2 , so the challenge isogeny needs to be of degree 2^{128} in our case. With p_{3923}, we get $D_c = 2^{65}3^{40}$. The remaining factors in T are used in T^0. Let $' : E_1 \; ! \; E_2$ be a random cyclic isogeny of degree D_c. Since the $T^0 D_c$-torsion is accessible over F_{p^2}, and we have $: E_0 \; ! \; E_1$ of degree coprime

to D_c, computing the corresponding ideal will be e cient for the prover using IsogenyToIdeal

Building (response). The response is computed as follows:

1. Compute $I_{\cdot} = [I\ ]$ IsogenyToIdeal$_{D_c}([\ ]\ ')$.
2. Set $I = \overline{I}\ I\ I_{\cdot}$ and compute $J =$ SigningKLPT$_{2^{1000}}(I; I\ )$.
3. Compute $\ =$ IdealToIsogenyFromEichler$(J; J\ ;\ ^0)$.
4. Compute $S =$ Compression$_{2^{1000}}(E_A;\ )$.

Veri cation. Upon receiving the string S the veri er needs to check that , the 2^{1000} isogeny represented by S, is a cyclic isogeny between E_A and E_2, and that the composition $'^{\wedge}$ is a cyclic isogeny. The veri er can recover the isogeny as Decompression$_{2^{1000}}(E_A; S)$. Then, the codomain of can be computed easily using the ideas from Section 4.1. To verify that $^{\wedge}$ is cyclic, it su ces to compute the action of $^{\wedge}$ on $E_A[2^f]$. There are points of order 2^f in this image if and only if $'$ is cyclic.

Remark 5.4.4. Here we have made the choice of responding with isogenies of degree in 2 to get the fastest veri cation possible, but this is not a necessity. In fact, there is a tradeo in signature e ciency versus veri cation e ciency. Signing time could be greatly improved by allowing some T-torsion inside the response isogeny. However, in this case, the veri er would be required to compute isogenies of degree dividing T, which is a lot less e cient than 2-isogenies with the current parameters. For instance, if $N\ p^{1=4}$, looking for a response isogeny of degree in T^2 would allow us to decrease the 2-adic valuation of the degree of by a factor of 5=3, at the cost of performing one T-isogeny computation. As each iteration of IdealToIsogenySmallFromEichler requires to compute several T-isogenies, we estimate that signing time could be decreased by approximately the same factor of 5=3. This would come at the cost of requiring the veri er to compute one T-isogeny. Further work might clarify the e ciency of our signature scheme if we were to push this idea to its full extent and look for with a degree dividing some power of T.

SQISign with p_{6983}. We can also devise a version of SQISign working with p_{6983} and IdealToIsogenyFromKLPT. Everything is roughly the same, up to the necessary changes of values for T and f. When IdealToIsogenyFromKLPT is used, we take the meet-in-the-middle exponent to be = 14 (this choice was based on empirical evidence, and is not necessarily the optimal choice).

An alternative key generation method. We now present an alternative key generation procedure. It is more e cient than the one presented above, but the distribution of the resulting secret keys is more di cult to analyze. We state it here for completeness. The idea is quite simple: instead of generating I rst and then computing J using KLPT, we generate an endomorphism of norm $N\ `$ and then derive I and J from . The endomorphism can be generated using FullRepresentInteger. Since this algorithm allows one to nd endomorphisms that are much smaller than in KLPT, we can obtain ^{0} of smaller

norm. Let us write e for the ℓ-valuation of the degree of θ. With KLPT we have $e \approx 2\log(p) + 2\log(N)$. Using this new method, the lower bound on e becomes the diameter of the graph (so that every secret isogeny of degree N^e can be generated that way). This allows one to take $e \approx \log(p)$. When e is smaller, key generation becomes more efficient as the translation from J to θ becomes faster. This idea leads to the key generation described below.

1. Select a prime $N \approx B$ inert in R uniformly at random.
2. Compute $\gamma \in O_0$ a random solution of $\gamma = \text{FullRepresentInteger}(n(I)N^e)$ and set $I = \langle N, \gamma \rangle$, $J = \langle \ell^e, \gamma \rangle$.
3. Compute $\theta = \text{IdealToIsogenyFromEichler}(J, O_0, [1]_{E_0})$ and set $pk = E_A$, the codomain of θ.

5.4.4 Performance

We discuss below the performance features of SQISign.

Signature size and comparison with existing schemes. For λ bits of classical security, we take $p \approx 2^{2\lambda}$. The public key is the j-invariant of the curve E_A, and it has size $2\log_2(p) = 4\lambda$. The secret can be seen as a pair N, I. The integer N is a $\log(p)/4$-bit prime, and we can represent I as a number in $[1, N^4 + 1]$, so another $\log(p)/4$-bit integer. In total, the secret key has size λ. The signature comprises E_1 and θ, where θ is produced by the CompressIsogeny algorithm from Section 4.1.5. As argued there, we can either use a full compression of exactly e bits, or allow for a few additional bits to accelerate the verification time. With the second method, the size is $e + 4(\text{def}(e/4) - 1)$. We recall that, with keys as in 5.3.2, $e = 15\lambda/4 \log(p) + O(\log(\lambda))$. Representing the commitment curve E_1 requires $2\log_2(p) = 4\lambda$ additional bits. We summarize these values in Table 5.3 when $\lambda = 128$. For our concrete instantiation we have $\log_2(p) = 254$, $f = 65$ and $e = 1000$.

Secret Key (bytes)	Public Key (bytes)	Signature (bytes)
16	64	204

Table 5.3: Size of SQISign keys and signature for the NIST-1 level of security and 128 bits of classical security.

These sizes make SQISign the most compact post-quantum digital signature targeting NIST-1 level of security, in terms of combined public key and signature size. With respect to round 3 candidates, it is more than 5 times more compact than Falcon [FHK+19] in terms of combined size, and only trails GeMSS [CFMR+19] in terms of signature size. SQISign signatures are more compact than RSA, and only about three times larger than ECDSA, for a comparable level of classical security.

Implementation. We implemented SQISign in C using the `libpari` library of PARI/GP 2.11.4 [The20]. We ran experiments on a 3.40GHz Intel Core i7-6700 (Skylake) CPU with Turbo Boost disabled. The code was compiled using clang-6.0 -O3 -Os -march=native -mtune=native -Wall -Wextra-std=gnu99

-pedantic . We used the fast algorithms presented in Section 4.1.4 for isogeny computations of degree greater than 100.

We made two di erent implementations. The rst, using the IdealToIsogeny-FromKLPT algorithm with p_{6983}, was introduced with the original submission of SQISign [DFKL+20]. It is available at https://github.com/SQISign/sqisign .

The second implementation was introduced in [DFLW22] and is available at https://github.com/SQISign/sqisign2 . It is based on the IdealToIsogeny-FromEichleralgorithms, supports both primes p_{3923} and p_{6983} and contains various additional optimizations. In particular, the rst implementation did not use the correct compression/decompression method, which explains why we observe a speed-up in the veri cation time as well. The speed-up in veri cation between p_{3923} and p_{6983} with the second implementation can be explained because we can put more 2^f-torsion in the challenge degree D_c.

The results are in Table 5.4. With our improvements, and moving from the old implementation with p_{6983} to the new one with p_{3923}, we observe a more than two-fold speed-up in all operations.

		FromKLPT/ p_{6983}			FromEichl/ p_{6983}			FromEichl/ p_{3923}		
		Keygen	Sign	Verify	Keygen	Sign	Verify	Keygen	Sign	Verify
Mcycles	1st quartile	1,918	7,722	135	2,569	6,673	82	717	3,645	64
	median	1,950	7,828	142	3,081	6,718	86	741	3,685	65
	3rd quartile	2,008	7,959	148	3,973	6,797	88	780	3,743	68
ms	1st quartile	562	2,266	39	754	1,958	24	210	1,070	19
	median	572	2,297	42	904	1,971	25	218	1,081	19
	3rd quartile	589	2,335	44	1,166	1,994	26	229	1,098	20

Table 5.4: Performance of SQISign in millions of cycles and in milliseconds. Statistics over 100 runs for key generation and signature, and over 250 runs for veri cation.

We stress that we did not attempt to produce a constant-time implementation. This appears to be an intensive task due to the complexity of the algorithms involved. In https://github.com/SQISign/sqisign-magma , we provide an additional implementation in Magma [BCP97] (it implements the same algorithms as the rst version of our C implementation). It performs poorly compared to our C code, but we hope it may serve as a useful reference.

Remark 5.4.5. One might wonder why key generation is so slow and has a huge variance with the algorithm IdealToIsogenyFromEichler and p_{6983}. This is because we encounter one of the bad cases described in Section 3.5: for the rst execution of IdealToIsogenySmallFromEichler, we get a maximal order that is connected with O_0 with an ideal of norm 2^{33}. We apply the countermeasure described at the end of Section 4.2.2, but the computation is much slower.

5.5 Zero-Knowledge

We now discuss the Zero-Knowledge property of our identi cation scheme.

5.5.1 An ad-hoc assumption

We prove that our identification scheme is computationally zero-knowledge assuming that the distribution of the response isogeny can be simulated. This result is not dependent on the concrete instantiation. In the following subsections, we will focus on the instantiation of SQISign with the SigningKLPT algorithm, and argue that in this case, the distribution D_{E_A} is indistinguishable from the uniform distribution of cyclic D-isogenies, assuming the hardness of Problem 5.5.6.

Let D_{E_A} be the distribution of isogenies in SQISign for a given public key E_A.

Lemma 5.5.1. If we assume that for any SQISign public key E_A, there exists a probabilistic polynomial algorithm S taking E_A as input whose output distribution is (computationally) indistinguishable from D_{E_A}, then the SQISign identification protocol is (computationally) Honest-Verifier Zero-Knowledge.

Proof. (Sketch.) We maintain the notation of Section 5.2. We construct a simulator as follows. The simulator generates the isogeny $S(E_A) = : E_A \to E_2$, a uniformly random isogeny $\hat{'} : E_2 \to E_1$ of degree D_c, and outputs $(E_1, ', E_2,)$. We now argue that transcripts constructed by the simulator are computationally indistinguishable from real transcripts. First, observe that in the real transcript all curves are nearly uniformly distributed, as long as D_c and the degree of are chosen large enough. This is due to the Ramanujan property of the supersingular isogeny graphs. With our assumption on S, the distribution of E_2 is (computationally) indistinguishable from the real one. The "challenge" is a random isogeny of degree D_c, so it is identically distributed in both the real and simulated transcripts; and thus so is the curve E_1.

It remains to prove that the "response" isogeny and its real counterpart cannot be efficiently distinguished. This stems directly from our assumption on S and the definition of D_{E_A}. □

5.5.2 On the distribution of signatures

The goal of this section is to understand the distribution of the isogenies obtained from $J = \text{SigningKLPT}_{2^e}(I; I)$. With Lemma 5.5.2, we will see that any such is the push-forward through the secret isogeny of some other isogeny . From the proof of Lemma 5.5.2, it appears that lies in a specific set of isogenies: the set P_N from Definition 5.5.4. This fact is obvious from Lemma 5.5.2 and the definition of P_N (which closely follows SigningKLPT). What is less trivial is that any element of P_N is a possible output of our algorithm, and that this set can be entirely computed from the knowledge of N. We will prove this in Proposition 5.5.5, and use it to state Problem 5.5.6 as our security assumption.

Figure 5.3 represents the isogenies involved in the proof of Lemma 5.5.2. The isogenies and curves that are public are highlighted in bold.

Lemma 5.5.2. Let $L \quad O$ and $\quad 2 L$ be as in Steps 2 and 9, respectively, of SigningKLPT. The isogeny corresponding to the output J of SigningKLPT_{2^e} is equal to $= [ ]$, where is an isogeny of degree 2^e verifying $= \hat{} \; '_L$.

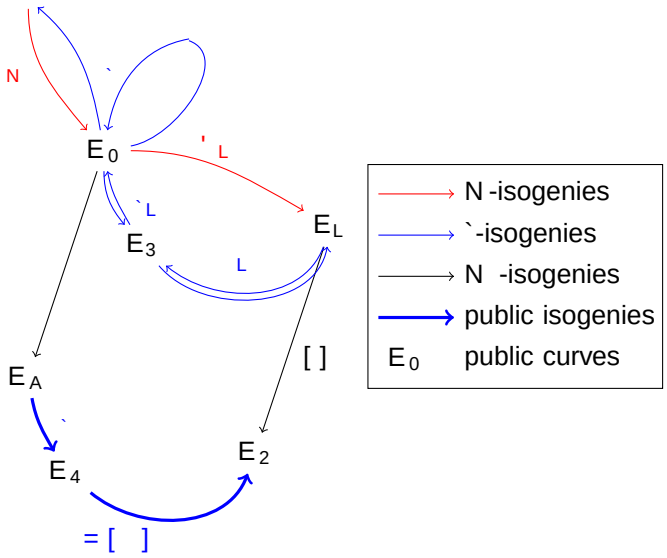


Figure 5.3: Analysis of the output of SigningKLPT under the Deuring correspondence

Proof. The endomorphism can be decomposed as $_N$ ` and then = = $_N$ ` . Since 2 L, it can be rewritten as `$_L$ $_L$ '$_L$ if we compose the other way, where `$_L$ $_L$ = ['$_L$] ` . Thus, we see that is de ned to be the dual of `$_L$ $_L$. Finally, is the image under of . We have the decomposition = [ ] = ` . Note that, equivalently, can also be seen as the image under $_N$ of the dual of ` . □

To show that lies in a public set P_N , we need to understand the exact link between and . It is clear that L is strongly related to as = ^ '$_L$. Hence, the codomain of is determined by the class of L in Cl(O_0). This underlies the de nition of P_N as the union of subsets $U_{L;N}$ indexed by all possible L (see De nition 5.5.4).

Remark 5.5.3. In Proposition 5.3.3, we saw that L lies among at most N + 1 possible values for a given input I . Each such L is uniquely determined by the class of K (with respect to $_O$) computed in Step 1 of Algorithm 33. In this sense, it would be more natural to divide outputs according to the classes of Cl_O (O). However, from this point of view, it is less clear that the set P_N is independent of . As argued in Remark 5.3.4, this number is exactly N + 1 with an overwhelming probability. To simplify the remaining statements, we will suppose that we are in this likely case.

Suppose we have chosen a class for among the N +1 candidates. We want to determine how the rest of the computation follows from this initial choice. We take a random e_0 2 e_0(N).

During Step 3, we compute . It is clear that N = n(L) and the value e_0 uniquely determines the distribution of outputs for FullRepresentInteger$_{D(`^{e_0})}$ (see Algorithm 2). Then, the projective pair (C_0 : D_0) only depends on L and

. We have proved in Corollary 3.3.2 that the projective pair $(C_1 : D_1)$ did not depend on the actual value of , so it is also uniquely determined by the choice of class for K (and thus of L) and . The rest of the computation is deterministic from there (up to failures that imply picking another).

We are now ready to characterize the set of possible outputs of SigningKLPT.

For a given L of norm N, we consider $U_{L;N}$ as the set of all isogenies computed as in Lemma 5.5.2 from elements $= 2 L$ where e_0 is a random element in $e_0(N)$, is a random output of FullRepresentInteger$_{N^{D(`e_0)}}$ and $= (C + !D)j$ where $p(C^2 + D^2)`e_1(N;N)$ is a quadratic residue mod N and where $C = CRT_{N;N}(C_0; C_1)$, $D = CRT_{N;N}(D_0; D_1)$ where $(C_0 : D_0) =$ IdealModConstraint$(L;)$ and $(C_1 : D_1)$ is a random element of $P^1(Z=N Z)$. For an equivalence class C in $Cl(O_0)$ we write $U_{C;N}$ for $U_{L;N}$ where $L =$ EquivalentPrimeIdeal(C).

De nition 5.5.4. $P_N = \bigcup_{C2Cl(O_0)} U_{C;N}$

With the next proposition we show that our de nition of P_N is the right one as it accounts for all the output of Algorithm 33.

Proposition 5.5.5. The set P_N from De nition 5.5.4 can be computed from the sole knowledge of N. The set $f J; J = [I] I ; 2 P_N g$ is exactly the set of outputs SigningKLPT$_{2^e}(I; I)$ as I runs over the non-trivial classes in $Cl(O)$.

Proof. The rst point follows directly from De nition 5.5.4. From the de nition, it is clear that each $U_{C;N}$ can be computed from $L =$ EquivalentPrimeIdeal(C) and N. With Lemma 5.5.2 and comparing the de nition of the $U_{L;N}$ with the steps of Algorithm 33 we see that its outputs are all contained in $f J; J = [I] I ; 2 P_N g$. To conclude the proof, we need to show that for any element J of this set, there exists an ideal I and an execution of Algorithm 33 such that SigningKLPT$_{2^e}(I; I) = J$. We write $=$ corresponding to $2 U_{L;N}$ for L I. For such a J, any I J can work as input as a consequence of Lemma 5.3.2. From Proposition 5.3.3, we know there exists K and a random input leading to $K =$ RandomEquivalentEichlerIdeal(I) with $[I] K$ O $L( )$ during the execution of Step 1. We can obtain as the output of FullRepresentInteger$_{N^{M(`e_0)}}$ by de nition of . Because of Corollary 3.3.2, we see that the elements C and D obtained in the execution of Algorithm 33 are in the same classes as the elements $C^0; D^0$ used in the computation of $=$ FullStrongApproximation$(N N ; C^0; D^0)$ in the de nition of P_N. Hence, we obtain the same element $2 L$, and we have just described an execution of Algorithm 33 that led to the output J precisely. □

5.5.3 Hardness Assumption for Zero-Knowledge

In this section, we present the hardness assumption on which the zero-knowledge property relies. We would like to show that the output of SigningKLPT cannot be linked to any isogeny from E_0 to E_A, and more speci cally . The formulation of Problem 5.5.6 is suggested by the results introduced in Lemma 5.5.2 and Proposition 5.5.5 where we showed that the signature isogeny is the image under of an isogeny lying in some public set of isogenies P_N (see De nition 5.5.4).

Recall that isog$(D; j(E))$ is the set of cyclic isogenies of degree D whose domain is a curve inside the isomorphism class of E. When P is a subset of

isog(D; j (E)) and : E ! E^0 is an isogeny with gcd(deg ; D) = 1, we write [ ] P for the subset f[ ] j 2 Pg of isog(D; j (E^0)). Finally, we let K denote a probability distribution on the set of cyclic isogenies whose domain is E_0, representing the distribution of SQISign private keys. With these notations, we de ne the following computational problem:

Problem 5.5.6. Let p be a prime, and D a smooth integer. Let : E_0 ! E_A be a random isogeny drawn from K, and let N be its degree. Let P_N isog(D; j _0) as in De nition 5.5.4, and let O be an oracle sampling random elements in [ ] P_N . Let : E_A ! ? be of degree D where either

1. is uniformly random in isog(D; j (E_A));
2. is uniformly random in [ ] P_N .

The problem is: given p; D; K; E_A; , to distinguish between the two cases with a polynomial number of queries to O.

We will assume that Problem 5.5.6 cannot be solved with non-negligible advantage by any polynomial time adversary. In Section 5.6 we brie y discuss several potential attack strategies; however, given current knowledge, no strategy seems better than direct key recovery, computing from the knowledge of E_A only.

Remark 5.5.7. To ensure the hardness of Problem 5.5.6, the size of the family P_N used in Proposition 5.5.5 must be exponential in the security parameter. With P_N from De nition 5.5.4, we have that jP_N j = ~(pN) (UPHA). Indeed, following the analysis of Section 5.5.2, there are (N) elements resulting from a given pair (L;) (the maximal number of possibilities is N + 1, and there is a constant probability that each element meets the quadratic residuosity condition). There are #Cl(O_0) = (p) possible L and x = O(log(p)) = ~(1) possible (following the choice for e_0(N) made in Remark 5.3.6 and Remark 5.3.7).

Remark 5.5.8. Here we formulated the security assumption of SQISign instantiated on top of SigningKLPT. Variants of Algorithm 33 would entail di erent families P_N in the de nition of Problem 5.5.6. We argue in Section 5.6 that any secure instantiation requires jP_N j to be exponential in the security parameter for any N but that this condition is not su cient.

Proposition 5.5.10 shows the security reduction to Problem 5.5.6. The proof relies on several heuristic assumptions. First, we need the heuristics behind Proposition 5.3.5 to prove that, for the chosen degree D = 2^e, SigningKLPT_{2^e} terminates in polynomial time.

Proposition 5.5.5 is not enough to prove Proposition 5.5.10: we need some information on the distribution of the outputs of Algorithm 33 over P_N . We will prove in Lemma 5.5.9 that when the input is uniformly distributed over Cl(O), the output distribution of SigningKLPT is statistically close to the uniform distribution on the set of possible outputs. This result is obtained with one new assumption:

Assumption 1. The distribution of classes obtained by taking the classes of the ideals I corresponding to 2 P_N is statistically close to the uniform distribution on Cl_O (O_0).

Lemma 5.5.9. (UPHA) The outputs of Algorithm 33, given uniformly distributed inputs, are distributed in a manner statistically indistinguishable from the uniform distribution on $\{J; J = [I] I ; 2 \nmid P_N\}$.

Proof. First, with Proposition 5.3.5, we showed UPHA that we can find an output of the correct degree. By Proposition 5.5.5, it lies in $\{J; J = [I] I ; 2 \nmid P_N\}$. From Lemma 5.3.1, we see that K lies in a uniformly random class of $\mathrm{Cl}_O(O)$ and so is K' in $\mathrm{Cl}_O(O_0)$. Once this class is fixed, the output is uniquely determined by the choice of . During Step 3, a random is selected. Repeating until the quadratic condition of Step 6 is met, we find a uniformly random solution among the elements in P_N contained in that equivalence class. By Assumption 1, this is statistically indistinguishable from a uniformly random element of $\{J; J = [I] I ; 2 \nmid P_N\}$. □

Under Lemma 5.5.1, the next proposition shows that Zero-Knowledge security reduces to Problem 5.5.6.

Proposition 5.5.10. (UPHA) When SQISign is instantiated with Signing-KLPT$_D$, distinguishing between D_{E_A} and the uniform distribution on $\mathrm{isog}(D; j(E_A))$ reduces to Problem 5.5.6.

Proof. We will show that we can construct a distinguisher for Problem 5.5.6 from a distinguisher between D_{E_A} and the uniform distribution on $\mathrm{isog}(D; j(E_A))$. When SigningKLPT is used to compute , the distribution D_{E_A} is statistically indistinguishable from the distribution of isogenies corresponding, through the Deuring correspondence, to the output of SigningKLPT$_D$ upon input $(I; I)$, where I lies in a uniformly random class of $\mathrm{Cl}(O)$ and I is computed from the secret key as an ideal corresponding to an isogeny between E_0 and E_A. Recall that the distribution of E_2 is nearly uniform, so the distribution of the class of I in the real execution is statistically close to the uniform distribution.

Clearly, the two distinguishers have compatible inputs. To prove the reduction, we have to show that the input distributions are statistically indistinguishable. Recall that for both problems there are two possible cases: either the isogeny is uniformly random of degree D or it has a special form. In the first case, the two problems clearly share the same input distribution. The second case is covered by Lemma 5.5.9. □

5.6 Cryptanalysis

In this section we study the hardness of Problem 5.5.6, the assumption behind SQISign's security introduced in Section 5.5. This problem is parameterized by a family of isogenies P_N . In Section 5.6.1, we start by looking at Problem 5.5.6 without any specific instantiation of P_N (we will only make assumptions on their size). Then, in Section 5.6.2, we study the case where a specific property verified by the family P_N might prove useful to break Problem 5.5.6, before trying to argue that our concrete family P_N from Definition 5.5.4 does not verify such a problematic property.

5.6.1 Cryptanalysis for generic families of P_N

We introduce below several variants of Problem 5.5.6 (Problems 5.6.1, 5.6.3 and 5.6.4), which we obtain by modifying the size of P_N and the range of

possible values N. We show that the first two problems can be solved efficiently, but argue that existing cryptanalysis techniques fall short for the last one.

A first important remark is that since P_N can be computed without the knowledge of , we do not have to worry about having problematic properties such as revealing a path from E_A to a special curve E (hence revealing critical information). This could happen, of course, but it cannot be more than an unlucky coincidence: the density of special curves is low in the set of all supersingular curves, so this event will only happen with negligible probability. Thus, to produce an effective distinguisher, an adversary has to exploit the information that is the image under of an element of the public set P_N.

In Problem 5.6.1, we assume that the value of N is fixed and publicly known, and we impose the constraint that the size of P_N is polynomial in the security parameter .

Problem 5.6.1. Let p be a prime and D be a smooth number, let B be a positive integer and let N be a prime smaller than B coprime to D. Let E_A be a supersingular curve for which there exists $: E_0 \to E_A$ of degree N. Let $P_N \quad \text{isog}(D; j_0)$ be a subset of size polynomial in and O an oracle sampling random elements in $[\ ] P_N$. Let $: E_A \to ?$ be of degree D, where either

1. is a uniformly random isogeny of degree D starting from E_A.
2. is a uniformly random element of $[\ ] P_N$.

The problem is: to distinguish between these two cases with a polynomial number of calls to O.

This problem can be easily solved due to the small size of P_N. Indeed, the number of isogenies in $[]\ P_N$ being polynomially bounded implies that we can enumerate them all with a polynomial number of calls. Once the list is made, distinguishing is easy.

Even though the problem is already broken, let us also study the prospects of key recovery; this will prove useful for the rest of this section. In fact, the setting of Problem 5.6.1 might allow efficient key recovery. The precise analysis is somewhat tedious, so we do not prove formally that this attack succeeds in polynomial time; rather, we sketch a brief outline and argue why it appears to be troublesome. If is the image of $2 P_N$ under , then its kernel is the image of the kernel of . In [Pet17], an attack on SIDH is devised using similar information (the action of the secret isogeny on some torsion group). Namely, if N is smaller than a certain bound (depending on D), then this could allow an adversary to recover . The actual parameters in our scheme are of the size that are troublesome for such an attack, where the degree of is much greater than N. With the estimates from Proposition 5.3.5, we see that $D \quad N^9$ in the generic case and this is enough for the attack of [Pet17] (that was recently improved in [KMP+20]).

Remark 5.6.2 The torsion point attack that we mention above is at the heart of the encryption scheme Seta introduced in Chapter 6.

Also note that the fact that N is public allows one to improve the brute-force key recovery attack. Indeed, in this case there are only $O(N)$ possible secret isogenies. Therefore, as mentioned in Section 5.3.2, the brute-force attack can be performed in (B).

In Problem 5.6.3 we look at a modi ed version of Problem 5.6.1 where we remove the assumption that N is public.

Problem 5.6.3. Let p be a prime, D a smooth number and B a positive integer. Let E_A be a supersingular curve for which there exists $: E_0 \to E_A$ of prime degree N with $N \quad B$. Let $P_N \quad \mathrm{Iso}_{D;j_0}$ be a family of subsets indexed by N of size polynomial in , and O an oracle sampling random elements in $[\ ]\ P_N$. Let $: E_A \to ?$ be of degree D, where either

1. is a uniformly random isogeny of degree D starting from E_A.
2. is a uniformly random element of $[\ ]\ P_N$.

The problem is: to distinguish between the two cases with a polynomial number of calls to O.

For the same reasons as for Problem 5.6.1, we can easily produce a distinguisher for Problem 5.6.3. Indeed, even if the exact N is unknown, there is still only one valid value and so $[\ ]\ P_N$ has polynomial size. Just as before, we can get it entirely by querying the oracle (we do not even have to know which N was used for this), so the problem is easy. A brute-force attack to recover the key will now cost (B^2), as we are back in the case described in Section 5.3.2.

The torsion attack of [Pet17] can no longer be applied, as it requires the knowledge of the exact value of N. However, we may still try to perform it on all possible N until one works. This yields an attack in (B). For Problem 5.6.4 we go back to the case where N is public, but this time P_N has exponential size with respect to the security parameter.

Problem 5.6.4. Let p be a prime and D be a smooth number, let B be a positive integer and let N be a prime smaller than B coprime to D. Let E_A be a supersingular curve for which there exists $: E_0 \to E_A$ of degree N (not provided as input). Let $P_N \quad \mathrm{Iso}_{D;j_0}$ be a family of subsets indexed by N and let O be an oracle sampling random elements in $[\ ]\ P_N$. Let $: E_A \to ?$ be of degree D, where either

1. is a uniformly random isogeny of degree D starting from E_A.
2. is a uniformly random element of $[\ ]\ P_N$.

The problem is: to distinguish between the two cases with a polynomial number of calls to O.

In this case, as with Problem 5.6.1, the brute-force key recovery attack is in (B). It is not clear that we can exploit the information provided by to help with key recovery. In particular, the potential key recovery attack against Problem 5.6.1 appears di cult to apply in this setting: the set P_N is too large to e ciently identify a possible preimage for $\ker$. We are reduced to trying all possible values for $\ker[\ ]$, which takes too long. In full generality, the calls to the oracles appear to be di cult to exploit. Indeed, enumerating all possibilities is out of the question, and it seems di cult to exploit anything else.

From the analysis above, it appears that having a secret N is important to prevent e cient attacks for key recovery, while having P_N of large size is the important feature for guaranteeing the hardness of the distinguishing problem. As argued in Remark 5.5.7, our family P_N from De nition 5.5.4 is large enough.

5.6.2 Exploiting the speci c properties of P_N

In this section, we focus concretely on the family P_N de ned in De nition 5.5.4 and used for the security of our signature scheme. In particular, we will study one way to build a distinguisher against Problem 5.5.6 assuming a hypothetical special property of the family P_N . We will show that there exists a family of isogenies, very closely related to our P_N , that su ers from this weakness. Then, we will argue why the family from De nition 5.5.4 does not su er from the same problem, and we will present experimental evidence to con rm our reasoning. In fact, the problematic family for which there is an easy distinguisher was the one used in the original SQISign paper [DFKL+20]. We later realized that there was a problem and a x was proposed in [DFLW22]. In SigningKLPT, we presented the secure version and not the original variant from [DFKL+20].

A generic distinguisher using the rst part of the walk. Let us assume that there exists an integer D_1 dividing D and polynomial in the security parameter such that for N there is a cyclic isogeny of degree D_1 from E_0 which is not in $I^{D_1} = \{ _1 \text{ of degree } D_1; 9 \ _2; \ _2 \ _1 \in P_N \}$. Then the set $\{ _1 \text{ of degree } D_1; 9 \ _2; \ _2 \ _1 \in [ ] P_N \}$ similarly misses some isogenies of degree D_1 from E_A. When D_1 is small (polynomial), we can enumerate all D_1 isogenies in polynomial time and use that to build a polynomial-time distinguisher.

Short of exploiting similar properties that allow one to construct a distinguishing criterion based on the study of a small, speci c part of , Problem 5.5.6 seems computationally hard. For instance, if the family P_N satis ed the criterion above but only for D_1 exponential in the security parameter, it is unclear that a distinguisher could be built from this knowledge.

A distinguisher against a modi ed version of the protocol. If we consider a modi ed version of SigningKLPT where FullRepresentInteger is replaced by RepresentInteger in Step 3, then the resulting family P_N su ers from the bad property we described above. In that setting, our distinguisher for Problem 5.5.6 is a consequence of the limitations pointed out in Section 3.1.3, because we take D to be a power of 2. Lemma 5.6.5 and the resulting Proposition 5.6.6 link the observations of Section 3.1.3 to a property of the set $[ ] P_N$.

Lemma 5.6.5. Let L be an O_0-ideal of norm N and let be an element in O_0 of norm N^e for some prime N. Take $2 O_0$ such that $= 2 L$. If $2 \{1; i; j; k\} \ i$, then $ _L( ) \ O_0[1 + i; 2i]$.

Proof. We have $2 \{1; i; j; k\} \ i \ O_0[1 + i; 2i]$. Now, $ _L( ) = O_0[\overline{ }; 2^e i]$, hence, $\overline{ } 2 O_0 \ O_0[1 + i; 2i] = O_0[1 + i; 2i]$; which proves the lemma. □

Proposition 5.6.6. Let $D = 2^e$ and $; N$ be as in Problem 5.5.6, and let the set P_N be de ned by a modi ed version of SigningKLPT where FullRepresentInteger has been replaced by RepresentInteger. There exists an isogeny $_0 2 \text{isog}(2; j(E_0))$ such that every $2 P_N$ can be decomposed as $= _1 \ _0$, where $_1$ is an isogeny of degree $2^{e \ 1}$.

Proof. Let J be the ideal corresponding to $2 [ ] P_N$. By the de nition of P_N , corresponds to the ideal $ _L( )$. It is easily veri ed that $L; ;$ satisfy

the requirements of Lemma 5.6.5 and that $\gamma \in \langle 1, i, j, k \rangle$ since it is a possible output of RepresentInteger$_{O_0}$. Thus, we can apply Lemma 5.6.5 and we get that $\theta_L(\gamma) \in O_0\langle 1+i, 2i\rangle$. This proves the result by taking θ_0 to be the isogeny corresponding to the ideal $O_0\langle 1+i, 2i\rangle$. □

Proposition 5.6.6 implies that, when defined as in Definition 5.5.4, the family $\mathcal{P}_N$ satisfies the special property introduced above. Indeed, we obtain that $I^1 = \{\gamma_1$ of degree 2^2 s.t. $\exists \gamma_2, \gamma_2\gamma_1 \in \mathcal{P}_N\}$ has size 1 (instead of 3), and so a trivial distinguisher can be built against Problem 5.5.6 simply by looking at the distribution of the first step of θ.

Fix against the attack. To block the distinguisher, the solution is to use FullRepresentInteger and not RepresentInteger, which is what we do in SigningKLPT. FullRepresentInteger was designed specifically to produce solutions that were not necessarily contained in $\langle 1, i, j, k \rangle$. If $\gamma = (x'+y'i+z'j+t'k)/2$ then it is easy to see that $\gamma \notin \langle 1, i, j, k \rangle$ as soon as $(x', y', z', t') \not\equiv (0, 0, 0, 0) \bmod 2$. Our analysis at the end of Section 3.1.3 showed that there were 4 possible configurations for $(x', y', z', t') \bmod 2$, and each can be obtained when the value of m' is greater than 1 (which we may assume). The reasoning above justifies that $\#I^1 > 1$ but not that it reaches the desired value of 3. Let us write I_1, I_2 for the two other O_0 ideals of norm 2. It can be verified that $I_1 = I_2 i$. Since $(x'+y'i+z'j+t'k)i = -y'+x'i+t'j-z'k$, it is easy to see that if some outputs of FullRepresentInteger are contained in I_1, then the same must be true for I_2 (and conversely). This proves that $\#I^1 = 3$, i.e., all three first steps are possible. Yet there could still be a bias in the distribution of that step, which would still give rise to an attack on Problem 5.5.6. We argue below that there is no such exploitable bias.

Further analysis on the first steps of the response isogeny. We continue the analysis by looking at what happens beyond the first 2-isogeny of the elements $\gamma \in \mathcal{P}_N$. For any $k \in \mathbb{N}$ smaller than e, we define $\theta_k : \gamma \mapsto \theta_k$ where θ_k is the unique isogeny of degree 2^k such that $\theta = \theta' \circ \theta_k$. We will study the sets $I^k = \theta_k(\mathcal{P}_N)$. We will start by trying to estimate $\#I^k$ for values of $k \leq 1/2 \log(p)$. Our analysis culminates in Proposition 5.6.8, which we prove under several plausible assumptions. Even though this does not prove that Problem 5.5.6 is hard, showing that $\#I^k$ is exponential in the security parameter rules out several possible attacks.

A truly meaningful result would be to show that the distribution D^k of the $\theta_k(\gamma)$ when γ is uniformly random in $\mathcal{P}_N$ is indistinguishable from the uniform distribution on the isogenies of degree 2^k. At the end of this section, we will try to argue that D^k is not biased when k is small. The results we obtain are not very formal, but we back them up with experiments.

The size of I^k. Our goal is to show that I^k contains a good portion of the isogenies of degree 2^k for values of $k \leq p^{1/2}$. Our final result is stated in Proposition 5.6.8 and basically follows from the fact that the isogenies of I^k only depend on the quaternion element γ of norm $N \ell^{e_0}$ when $k \leq e_0$ (this fact follows from the analysis underlying Lemma 5.5.2). We recall that in the definition of $\mathcal{P}_N$, γ is a possible output of FullRepresentInteger such that the end of the computation in SigningKLPT terminates.

One of the main ingredients of our proof is a result (stated as Proposition 5.6.7) on the number of of norm M that can be obtained as output of FullRepresentInteger. We use the notation $_M$ for the set of primitive $2 O_0$ of norm M.

Proposition 5.6.7. (UPHA) Let $M > p$. Under plausible heuristics, there exists a constant $c_1 > 0$ such that the number of $2 _M$ that are possible outputs of FullRepresentInteger$_{M}$ is larger than $\#_M c_1/\log(M)$.

Proof. Let $2 = x^0 + iy^0 + jz^0 + kt^0$ and $M^0 = 4M \quad p(f_O(z^0, t^0))$. Given the requirements of Cornacchia, is going to be an admissible output if and only if M^0 is a near-prime and the pair (z^0, t^0) can be sampled during the first two steps of Algorithm 8. For (z^0, t^0) it is easy to verify that this is the case. Indeed, the value of $|z^0|$ must be smaller than $2m$. Thus, there is a possibility that this value is picked. After that, we know that the correct value of $|t^0|$ must be smaller than m^0, and so there is also a possibility that the correct value is picked. Then, under the assumption that M^0 behaves as a normal integer of the same size, we get that there exists a constant c_1 such that a fraction $c_1/\log(M)$ of all the M^0 are near-primes. Thus, the same fraction of are possible outputs of FullRepresentInteger, and this concludes the proof. □

Proposition 5.6.8. (UPHA) Let $_0$ be as in Remark 3.2.4, and $_1$ as in Remark 5.3.7. Let us take $" > 1 + 2 _0$ and $e_0(N)$ as in Proposition 5.3.5. If $k \in [\frac{\log(p)}{2} + _0 + _1; \frac{\log(p)}{2} \quad _0 + _1 + "]$, then there exists a constant $c > 0$ such that

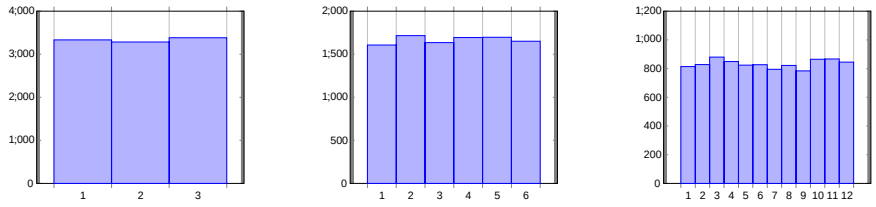
$$\#I^k \quad c \cdot 2 \cdot 3^{k-1}/(\log(p) + ").$$

Proof. First note that the constraint $" > 1 + 2 _0$ ensures that the interval $[\frac{\log(p)}{2} + _0 + _1; \frac{\log(p)}{2} \quad _0 + _1 + "]$ is not empty.

Let $'$ be an isogeny of degree 2^k. We write $I_{'}$ for the corresponding ideal and $L_{'} = \text{EquivalentPrimeIdeal}(I_{'})$, $N_{'} = n(L_{'})$. There exists a quaternion element $_{'}$ of norm $N_{'} 2^k$ such that $O_0 _{'} = I_{'} \overline{L_{'}}$. It can be easily verified that $' \in I^k$ if and only $_{'}$ is in the set of possible involved in the definition of P_N. For $_{'}$ to be in , we need to verify the following things: $k \in e_0(N_{'})$, $_{'}$ is a possible output of FullRepresentInteger$_{N_{'} 2^k}$ and the rest of the computation of SigningKLPT (Step 4 to Step 7) must succeed from $_{'}$.

From Lemma 3.2.3, we get that $\log(N_{'}) \in [\log(p)/2 \quad _0; \log(p)/2 + _0]$. If we apply that to the definition of $e_0(N_{'})$ from Proposition 5.3.5, we get that k must be in $e_0(N_{'})$ with overwhelming probability. Then, if we assume that $_{'}$ is distributed correctly over the $_{N_{'} 2^k}$, Proposition 5.6.7 tells us there exists a constant $c_2 > 0$ such that more than a fraction $c_2/(\log(p) + ")$ of the $_{'}$ will be possible outputs of FullRepresentInteger. Finally, we can make the assumption that a constant fraction of those $_{'}$ will lead to a successful computation of Step 4 to Step 7 in SigningKLPT. Thus, we obtain that there exists some constant $c > 0$ such that a fraction greater than $c/(\log(p) + ")$ of all the $_{'}$ are contained in , and we can conclude the proof. □

Proposition 5.6.8 rules out distinguishing attacks by counting the elements in I^k, which is necessary for security, as explained in Section 5.6.1. To fully rule out simple distinguishers, we need to understand the distribution D^k.



(a) k = 1 (b) k = 2 (c) k = 3

Figure 5.4: Distribution of the k rst steps of for 10 SQISign keys and random input ideal over 1000 attempts.

The distribution D^k. Biased distributions D^k, especially for small values of k, can be easily detected; this would break Problem 5.5.6. Once again, our analysis focuses on the quaternion element. Under the Deuring correspondence, any bias on the distribution D^k is a consequence of a bias on the distribution of $O_0\langle 1, \ell^k i\rangle$ among the ideals of norm 2^k. If $2 = x + yi + zj + tk$ for $x, y, z, t \in \mathbb{Z}$, it can be shown that $O_0\langle 1, \ell^k i\rangle$ depends on the values of $(x, y, z, t) \bmod (2\ell^k)$. It is easy to see that the values of z, t are sampled without any bias $\bmod (2^k)$ when m', m'' are big enough compared to ℓ^k (which we may assume since we look at small values of k). After that, we can only argue informally that the near-primality condition on M $pf(z, t)$ should not introduce any bias on the value of $z, t \bmod 2\ell^k$. It also seems plausible that the output of Cornacchia on random near-prime inputs of a given size should not skew the distribution of x, y but we cannot really prove it. Short of proving a positive result, we can at least point out that our formulation of FullRepresentInteger avoids several pitfalls that would have led to a noticeable bias. The two examples that we give below are focused on the case $\ell = 2$ and $k = 1$. Both can be veri ed easily with small experiments. When $\ell = 2$, one might be tempted to allow the computation to succeed even when the condition $x' = t' \bmod 2$ and $y' = z' \bmod 2$ is not met. In the end, if $M = N \cdot 2^{e_0}$, then $x' + y'i + z'j + t'k$ has norm $N \cdot 2^{2+e_0}$ and might be a suitable solution for the rest of the computation. However, this would result in a strong bias in the distribution of isogenies in I^{-1}, as it would favor elements in $\langle 1, i, j, k\rangle$ that are contained inside $O_0\langle 1 + i, 2i\rangle$. A skewed distribution is also the explanation behind Remark 3.1.9. When swapping x', y' to satisfy $x' = t' \bmod 2$ and $y' = z' \bmod 2$, one increases the chances that $(x', y', z', t') \bmod 2 \in 2\langle (1, 0, 0, 1), (0, 1, 1, 0)\rangle$ and thus modi es the distribution D^{-1}.

Experimental evidence. Figure 5.4 presents the result of an experiment studying the distributions D^k for small values of k. The results are consistent with our informal analysis.

5.7 Improvement perspectives

The complexity of our signature scheme entails several choices for instantiation, some of which might di er critically from the solutions we described. We have already described several ideas, but it is hard to nd the optimal choices in the

various tradeoffs that arise, so there are probably a lot of potential improvements.

Among the critical points for improvement are the norm equations algorithms presented in Chapter 3. The problem lies not in the efficiency of these algorithms in themselves, but in the size of the solutions. We have shown how much the difference in size between the outputs of KLPT and SpecialEichlerNorm can impact the performances of our schemes by allowing us to select better primes. Given that all those algorithms are using the same building blocks, it is our hope that an improvement for one of them would result in an improvement for all of them. This would allow us to improve upon every aspect of the signature computation. For instance, a smaller output to SigningKLPT would mean a shorter signature and also a faster ideal to isogeny translation. But a smaller output of KLPT would also mean faster executions of IdealToIsogenySmallFromKLPT. In the end, this would yield a huge improvement of the overall efficiency of the signature computation. Given the improvement we obtained by switching from IdealToIsogenyFromKLPT to IdealToIsogenyFromEichler, one might wonder if we cannot find other ways to obtain faster algorithms for the translation.

Finally, there is still a lot of work to do on parameter selection.

Chapter 6

Encryption: SETA

In this chapter, we present SETA (short for Supersingular Encryption from Torsion Attack) or Seta [1], a new isogeny-based encryption scheme. The content of this chapter is for all practical purpose identical to the paper \SETA: Supersingular Encryption from Torsion Attacks" authored jointly with L. De Feo, C. Delpech de Saint Guilhem, T. B. Fouotsa, P. Kutas, C. Petit, J. Silva and B. Wesolowski [DFFdSG+21]. At the end of this chapter, we introduce the \Uber" Isogeny problem, a new generic computational problem that is linked with various known isogeny problems.

The construction that we will present in Section 6.2 relies on some results originally published by C. Petit [Pet17] targeting the cryptanalysis of the key exchange SIDH [JD11] and later improved in [QKL+21].

6.1 Preliminaries

In this section, we present several useful preliminaries for the content of this chapter. In particular, we introduce SIDH and CSIDH, two key exchanges that are behind the two main families of schemes based on isogenies. Our new protocol Seta is related to SIDH because its key mechanism is inspired by an attack against SIDH that we present in Section 6.1.3. There is no direct link between Seta and CSIDH, but our \Uber" isogeny problem introduced in Section 6.5 is directly inspired by the group action framework of CSIDH.

6.1.1 The SIDH key exchange

The SIDH (short for Supersingular Isogeny Di e-Hellman) key exchange was introduced in 2011 by De Feo and Jao [JD11]. It was one of the rst scheme based on isogenies between supersingular curves. To this day, it remains one of the most promising protocols in isogeny-based cryptography. It is very compact, and very e cient compared to every other protocol based on isogenies. SIKE [ACC+20], the candidate representing isogenies in the NIST post-quantum competition, is derived from SIDH. The NIST has ranked it among the third-round alternate candidates in its post-quantum competition.

[1] which means \walk" in Hungarian

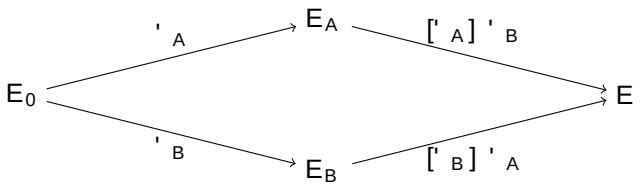


Figure 6.1: SIDH-isogeny diagram.

The idea of SIDH is the following: given a common starting curve E_0, the two participants, Alice and Bob, generate isogenies φ_A, φ_B of degree N_A, N_B with $\gcd(N_A, N_B) = 1$. Their public keys are the curves E_A, E_B, together with additional pieces of information, to make possible the computation of the two push-forward isogenies $\varphi[A]\varphi_B$ and $[\varphi_B]\varphi_A$ depicted in Figure 6.1. We have already explained in Section 1.1.2 that the codomains of these push-forward isogenies are isomorphic. Thus, Alice and Bob can derive a common key $j(E)$. In the case of SIDH the degrees are powers of two small primes ℓ_A, ℓ_B, which makes isogeny computations efficient from the kernels if the N_A, N_B torsion is defined over $\mathbb{F}_{p^2}$. To ensure that, we take a prime p with $p = f N_A N_B \pm 1$. More precisely, to compute the push-forwards we apply the formulas $\ker[A]\varphi_B = \varphi_A(\ker \varphi_B)$ and this is why Alice's SIDH public key is the curve E_A together with $\varphi_A(P_B), \varphi_A(Q_B)$ where $\langle P_B, Q_B \rangle = E_0[N_B]$ (and the reverse for Bob's).

Because of the additional torsion points that Alice and Bob need to include in their public keys, the problem behind the security of SIDH is not the SIP (Problem 6.5.7) but the CSSI presented as Problem 6.1.1 and its decisional variants. In fact, the CSSI is the SIDH key recovery problem but the decisional variant is the one underlying the security of the key exchange.

Problem 6.1.1. (CSSI) Let ℓ_A, ℓ_B be two distinct small prime numbers and $N_A = \ell_A^{e_A}, N_B = \ell_B^{e_B}$ for some exponents e_A, e_B. Let $\varphi_A : E_0 \to E_A$ be an isogeny whose kernel of degree N_A. Given E_A and the values $\varphi_A(P_B), \varphi_A(Q_B)$ for P_B, Q_B a basis of $E_0[N_B]$ find a generator R_A of $\ker \varphi_A$.

Remark 6.1.2. In Problem 6.1.1, the starting curve E_0 is fixed. In particular, in some cases its endomorphism ring may be known. In Section 6.1.3, we will consider the SSI-T, a generalization of the CSSI where the domain and codomain can be any supersingular curves.

Remark 6.1.3. The most promising attacks against SIDH are really targeting the CSSI and not the SIP in the sense that the additional torsion information are often at the heart of the attack. This is the case with the torsion point attacks that we present in Section 6.1.3.

6.1.2 The CSIDH key exchange

CSIDH (Commutative SIDH) is an alternative key exchange protocol introduced in [CLM+18]. It is based on the group action induced by the class group of $\mathbb{Z}[\sqrt{-p}]$ on the curves of $E_{\mathbb{Z}[\sqrt{-p}](p)}$ (see Section 2.4 for the definition of this group action). Since the class group is abelian we obtain naturally the commutative diagram depicted in Figure 6.2. Despite the similarities between Figure 6.1 and

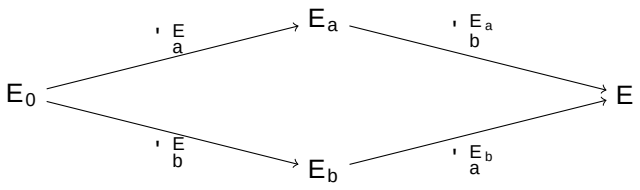


Figure 6.2: CSIDH-isogeny diagram.

Figure 6.2, the isogenies involved are not the same and are not computed in the same manner at all.

Let us write $\mathcal{O} = \mathbb{Z}[\pi]$. It is easy to show that $E_{\mathcal{O}}(p) = S(p) \setminus \mathbb{F}_p$. Indeed, when the curve is defined over $\mathbb{F}_p$, the p-power Frobenius π is an endomorphism of the curve, and so the embedding of $\mathcal{O}$ is obtained by identifying π with $\sqrt{-p}$ and the scalars in $\mathbb{Z}$ with the scalar multiplication morphism. For any ideal $\mathfrak{a}$ and curve $E \in E_{\mathcal{O}}(p)$, the action $\mathfrak{a} \star E$ is realized through an isogeny $\varphi_{\mathfrak{a}}^{E}$ of degree $n(\mathfrak{a})$. To obtain a key exchange, Alice and Bob will respectively pick secret ideals $\mathfrak{a}, \mathfrak{b}$ and compute the curves E_a, E_b that constitutes their public key. Then, the common key is $E_{ab} = E_{ba} = (E_a)_b = (E_b)_a$.

We state the CSIDH key recovery problem below [CLM+18, Problem 10]. As for SIDH, there is a decisional variant upon which holds the security of the key exchange.

Problem 6.1.4. Given two supersingular elliptic curves E, E_0 defined over $\mathbb{F}_p$ with the same $\mathbb{F}_p$-rational endomorphism ring $\mathcal{O}$, find an ideal $\mathfrak{a}$ of $\mathcal{O}$ such that $E_a = E_0$. This ideal must be represented in such a way that the action of $\mathfrak{a}$ on any curve can be evaluated efficiently, for instance $\mathfrak{a}$ could be given as a product of ideals of small norm.

6.1.3 Torsion attacks and trapdoor curves

In this section, we consider the following generalization of the CSSI.

Problem 6.1.5. (SSI-T) Let E_1, E_2 two supersingular curves in $S(p)$. Let D and N be coprime integers. Let $\varphi : E_1 \to E_2$ be a secret isogeny of degree D. Assume that we know the action of π on $E_1[N]$. Compute φ.

The SSI-T problem makes sense for any D, N which are coprime. But we are going to restrict to the case where we have compact representation of the input and output of our problem.

Definition 6.1.6. Let N be an integer and let p be a prime number. Let E be a curve in $S(p)$. We call $E[N]$ efficiently representable if representing points in $E[N]$ requires polynomial space in $\log p$.

Henceforth, we assume that D, N are smooth and that $E[DN]$ is efficiently representable (in particular, as in Chapter 5, we will focus on the case where all the relevant torsion is defined over $\mathbb{F}_{p^2}$). This implies that we have polynomial time algorithms to manipulate all the objects involved in our problem (see Section 4.1). The SSI-T is one of the problem underlying the security of the Séta.

The work of C. Petit in [Pet17] and the later improvements from [QKL+21] targeted ways to solve the SSI-T using the knowledge of the endomorphism ring of E_1. The idea relies on the fact, proven in Section 2.3.2, that the existence of an isogeny between E_1, E_2 imply an embedding of a suborder of $\mathrm{End}(E_1)$ inside $\mathrm{End}(E_2)$. The idea from Petit was that the torsion information given by the image of $E_1[N]$ through φ could be used to compute one endomorphism of this embedding.

We recall (slightly modified version of)[QKL+21, Theorem 3] how finding one of those endomorphism of E_2 relates to finding the secret isogeny φ:

Theorem 6.1.7. Let $\varphi : E_1 \to E_2$ be a secret isogeny of degree D. Assume that $E[N]$ and $E[D]$ are efficiently representable for any supersingular curve E and that the action of φ on $E_1[N]$ is given. Suppose, furthermore, that we know $\theta \in \mathrm{End}(E_1)$ and $d, e \in \mathbb{Z}$ such that the trace of θ is 0 and $\deg(\varphi\hat{\varphi} + [d]) = N^2 e$. Let M be the largest divisor of D such that $E_2[M] \subset \ker(\varphi\hat{\varphi}) \cap E_2[D]$. Let k be the number of distinct prime divisors of M. Then we can compute φ in time $\tilde{\mathcal{O}}(2^k \sqrt{e})$.

Proof. We sketch the proof of the theorem. Let $\theta = \varphi\hat{\varphi} + [d]$. Then if $\ker(\theta)$ is cyclic, then $\theta = \theta'$ where $\deg(\theta) = \deg(\theta') = N$ and $\deg(\theta) = e$ and the kernels of θ and θ' are cyclic. [QKL+21, Theorem 3] shows that $\ker(\theta)$ is always cyclic if N is odd and if N is even, then $\theta = \theta' \circ [K]$ where $\deg(\theta) = \deg(\theta') = N/K$, $\deg(\theta) = e$ and $K = 1$ or $K = 2$.

Then one can compute θ and K using the torsion point information and θ' using the observation that $\ker(\hat{\theta}\theta') = \theta(E_2[B])$. The isogeny θ can be computed by a meet-in-the-middle algorithm. Once θ is computed, one can compute φ by looking at $G = \ker(\varphi\hat{\varphi}) \cap E_2[D]$. If $M = 1$ then G is cyclic and can be recomputed easily. If not, then one can use [Section 4.3][Pet17] to recover φ. The cost of this step is $\tilde{\mathcal{O}}(2^k)$ where k is the number of prime factors of M. □

Remark 6.1.8. Theorem 6.1.7 in particular implies that one can recover φ in $\tilde{\mathcal{O}}(\sqrt{e})$ whenever the number of distinct prime divisors of D (and hence M) is smaller than $\log\log p$. In Section 6.2.3, we introduce a condition on the quadratic order $\mathbb{Z}[\theta]$ to ensure that M is always equal to 1.

The key ingredient to Theorem 6.1.7 is the knowledge of θ. When $M = 1$ (which will be the case for the concrete inversion procedure in Algorithm 35), all we really need is the action of θ on $E_1[N]$. Indeed, from the sketch of proof of Theorem 6.1.7, we see that θ is only used to compute the kernel of the two isogenies θ and θ' of degree N. These kernels are computed by evaluating the N-torsion $\theta = \varphi\hat{\varphi} + [d]$, which can be done with the action of θ and φ on $E_1[N]$.

Note, that the action of θ on $E_1[N]$ is hard to recover from the knowledge of E_1 only (mainly because computing endomorphisms of E_1 is hard). This motivates a notion of (D, N)-trapdoor T to encompass any kind of information that enables the computation described in the proof of Theorem 6.1.7.

Definition 6.1.9. Let p be a prime number and let D and N be coprime smooth integers. Then a tuple (E, T) is called a (D, N)-trapdoor curve if one can use T to solve any instance of the SSI-T problem (with parameters D, N, p) with starting curve E in polynomial time. We sometimes call T the trapdoor.

In [QKL+21] the authors introduces a polynomial-time algorithm for constructing (D, N)-trapdoor curves whenever $N > D^2$ and the number of prime divisors of $D < \log\log p$. The main idea is to reproduce the set-up of Theorem 6.1.7. Thus, if one can construct a supersingular elliptic curve E together with an endomorphism $\in \mathrm{End}(E)$ verifying the requirements of Theorem 6.1.7, and compute the action of this endomorphism on $E[N]$, then one can solve SSI-T in polynomial time (by finding an e which is sufficiently small).

The conditions put on $$ in Theorem 6.1.7 are essentially conditions on the minimal polynomial of $$, meaning that every trace zero element in the quaternion algebra whose norm is $(N^2e - d^2)/D^2$ can be used as a suitable $$. This implies we can obtain potential (D, N)-trapdoor curves from curves in $E_{\mathcal{O}}(p)$ for a quadratic order $\mathcal{O}$ of the form

$$Z \sqrt{\frac{N^2e - d^2}{D^2}}.$$

Let us now describe how such a $$ can be generated. We have introduced in Chapter 3 algorithms to solve norm equations in $Z + D\mathcal{O}$ where $\mathcal{O}$ is a maximal order of $B_{p,1}$. But this is not enough here, because we need an element of specific norm and trace. In our case, we are happy to generate the maximal order $\mathcal{O}$ (and the curve E) after the quadratic order $\mathcal{O}$. Thus, our idea, is to find an element in $B_{p,1}$ with the correct norm and trace, take the quadratic order $\mathcal{O} = Z[\,]$ and then find a curve $E \in E_{\mathcal{O}}(p)$ by finding a maximal order $\mathcal{O}$ containing $$. Since $\mathrm{Tr}(\,) = 0$, the element $\in B_{p,1}$ can be written as $ci + bj + aij$. The degree of $= [d] + \hat{\ }$ is $D^2(p^2a + p^2b + c^2) + d^2$ when $$ has degree D. Observe that a, b, c can be rational numbers, but since $$ is an integral element its norm $p^2a^2 + p^2b^2 + c^2$ must be an integer. So one has to find d, e such that $N^2e - d^2$ is divisible by D^2 and is positive before expressing $n = (N^2e - d^2)/D^2$ as $p^2a^2 + p^2b^2 + c^2$.

This can be achieved when $N > D^2$. Let $nD^2 = N^2e - d^2$. Then one has to find a rational solution to the equation $p^2a^2 + p^2b^2 + c^2 = nD^2$, which exists whenever n is a quadratic residue modulo p (if that is not the case, then one chooses a different d and e). A solution can be found using Denis Simon's algorithm [Sim05]. From there, we can find a maximal order $\mathcal{O}$ containing $$ and then compute a supersingular elliptic curve whose endomorphism ring is isomorphic to $\mathcal{O}$ (see Algorithm 37 in Section 6.3.2). After that, the action of $$ on the N-torsion can be found using an explicit representation of $\mathcal{O}$ and the algorithms on the Deuring correspondence. Everything can be done in polynomial time (see `SetaQuadraticOrderGen` and `SetaCurveGen` in Section 6.3 for more details), leading to the following theorem:

Theorem 6.1.10. Let p be a prime number and let D and N be smooth coprime integers such that $N > D^2$ and the number of distinct prime divisors of D is smaller than $\log\log p$. Then there exists a polynomial-time algorithm which outputs a (D, N)-trapdoor curve E with the following information:

- The j-invariant of E.
- Integers d, e with $e = O(\log(p))$.
- A basis P, Q of $E[N]$ and the points $(P), (Q)$ for a trace 0 endomorphism such that $\deg([D] + [d]) = N^2e$.

6.2 Séta trapdoor one way function and public key encryption scheme

In this section we describe a general trapdoor one-way function where the main idea is to turn the attacks from [QKL+21] into a trapdoor mechanism.

We first generalize the CGL hash function, and we describe a trapdoor subfamily of this generalization. We then provide more details on key generation, evaluation, and inversion. We finally describe the Séta public key encryption scheme and its CCA version.

6.2.1 Generalized Charles-Goren-Lauter hash function

We generalize the CGL hash function family introduced in [CLG09]. To select a hash function from this family, one selects a j-invariant $j \in \mathbb{J}_p$ which canonically fixes a curve $E/\mathbb{F}_{p^2}$ with $j(E) = j$. There are $\ell + 1$ isogenies of degree ℓ connecting E to other vertices. These $\ell + 1$ vertices can be ordered canonically, and one of them can be ignored. Then, given a message $m = b_1 b_2 \dots b_n$, with $b_i \in [\ell]$, hashing starts by choosing a degree-ℓ isogeny from E according to symbol b_1 to arrive at a first curve E_1. Not allowing backtracking, there are then only ℓ isogenies out of E_1 and one is chosen according to b_2 to arrive at a second curve E_2. Continuing in the same way, m determines a unique walk of length n. The output of the CGL hash function h_j is then the j-invariant of the final curve in the path, i.e. $h_j(m) := j(E_n)$, where the walk starts at the vertex j and is defined as above. We see that starting at a different vertex j' results in a different hash function $h_{j'}$.

We modify this hash function family in three ways. First, we consider a generalization where we do not ignore one of the $\ell+1$ isogenies from the starting curve E. That is, we take inputs $m = b_1 b_2 \dots b_n$ where $b_1 \in [\ell + 1]$ and $b_i \in [\ell]$ for $2 \le i \le n$; this introduces a one-to-one correspondence between inputs and cyclic isogenies of degree ℓ^n originating from E.

Secondly, we consider a generalization where the walk takes place over multiple graphs G_{ℓ_i}. Given an integer $D = \prod_{i=1}^{n} \ell_i^{e_i}$ where the ℓ_i are prime factors, we introduce the notation $\Pi(D) := \prod_{i=1}^{n} (\ell_i+1)\,\ell_i^{e_i-1}$. We then take the message m to be an element of

$$[\Pi(D)] = \{(m_1, \dots, m_n) \mid m_i = b_{i1} b_{i2} \dots b_{ie_i};\ b_{i1} \in [\ell_i + 1];\ b_{ij} \in [\ell_i] \text{ for } 2 \le j \le e_i;\ \text{for } 1 \le i \le n\}$$

where each m_i is hashed along the graph G_{ℓ_i}. To ensure continuity, the j-invariants are chained along the hash functions, that is, we write $j_i = h_{j_{i-1}}(m_i)$, where j_{i-1} is the hash of m_{i-1}. Thus, only $j = j_0$ parameterizes the overall hash function. As before, this generalization returns the final j-invariant $j_n = h_{j_{n-1}}(m_n)$ as the hash of m.

Thirdly, we also modify the CGL hash function to return the images of two canonically defined torsion points P_j and Q_j of order N under the D-isogeny $\varphi_m : E_j \to E_{j_n}$.

We call the resulting hash function family generalized CGL or G-CGL, and we denote it by $H^{p;D;N}$, namely

$$H^{p;D;N} = \left\{ h_j^{D;N} : m \mapsto (j(E_n); \varphi_m(P_j); \varphi_m(Q_j)) \mid j \in \mathbb{J}_p \right\}.$$

Remark 6.2.1. The method that we just described is quite similar to the Compression/Decompression algorithms that we introduced in Section 4.1.5 in the way it takes an isogeny and convert it into a number.

6.2.2 A trapdoor function family from the G-CGL family

Given p, D and N, let $J_{T;p} \subset J_p$ be the set of j-invariants of (D, N)-trapdoor curves defined over F_{p^2} (see Definition 6.1.9). By definition of a trapdoor curve, for any $j_T \in J_{T;p}$, the hash function $h_{j_T}^{D;N}$ can be inverted using the trapdoor information. We hence obtain the following family of trapdoor functions:

$$F_T^{p;D;N} = \left\{ f_{j_T}^{D;N} : m \mapsto (j(E_n), \varphi_m(P_{j_T}), \varphi_m(Q_{j_T})) \mid j_T \in J_{T;p} \right\},$$

where $f_{j_T}^{D;N} := h_{j_T}^{D;N}$.

Injectivity. We observe that, for a proper choice of parameters, the functions are injective.

Lemma 6.2.2. Let $N^2 > 4D$. Then for any $j_T \in J_{T;p}$, $f_{j_T}^{D;N}$ is injective.

Proof. Let $N^2 > 4D$ and $j_T \in J_{T;p}$, suppose that a function $f_{j_T}^{D}$ is not injective, i.e. that there are two distinct isogenies φ and φ' of degree D from E_{j_T} to E_c, corresponding to two distinct messages, with the same action on $E_{j_T}[N]$, implied by the colliding images of P_{j_T} and Q_{j_T}. Then, following [MP19, Section 4], their difference is also an isogeny between the same curves whose kernel contains the entire N-torsion. This, together with [Sil86, Lemma V.1.2], implies that $4D \deg(\varphi - \varphi') \mid N^2$. Taking $N^2 > 4D$ ensures that in fact $\varphi = \varphi'$ and therefore that $f_{j_T}^{D;N}$ is injective. □ □

One-wayness. One-wayness of our function family relies on Problem 6.2.3 below. This problem is a variant of the CSSI problem introduced in [JD11], with the difference that the starting j-invariant is chosen at random from $J_{T;p}$ (instead of being fixed) and only the min-entropy of the distribution is specified.

Problem 6.2.3 (Trapdoor computational supersingular isogeny (TCSSI) problem). Given p and integers D and N, let j_T be a uniformly random element of $J_{T;p}$ and $\varphi_m : E_{j_T} \to E_m$ be a random isogeny of degree D sampled from a distribution X with min-entropy $H_1(X) = O(\lambda)$. Let $\{P_{j_T}, Q_{j_T}\}$ be a basis of the torsion group $E_{j_T}[N]$. Given $E_{j_T}, P_{j_T}, Q_{j_T}, E_m, \varphi_m(P_{j_T})$ and $\varphi_m(Q_{j_T})$, compute φ_m.

Lemma 6.2.4. Let j_T be a uniformly random element of $J_{T;p}$. Then the function $f_{j_T}^{D;N} \in F_T^{p;D;N}$ is (quantum) one-way under the (quantum) hardness of Problem 6.2.3.

Proof. It is easy to check that the distribution of isogenies resulting from hashing a uniform $m \xleftarrow{\$} [\lambda(D)]$ has the required entropy; hence, the reduction is immediate. □ □

Algorithm 35 InverseTrapdoor(j_T, T, c)

Input: $j_T \in J_{T,p}$, a trapdoor T and c.
Output: $m \in [\ (D)]$ such that $f_{j_T}^{D,N}(m) = c$.

1: Parse c as $(j_m, P_m, Q_m) \in F_{p^2} \quad (\overline{F_{p^2}})^2 \quad (\overline{F_{p^2}})^2$.
2: Parse T as $e, d, P_{J_T}, Q_{j_T}, \ (P_{j_T}), \ (Q_{j_T})$.
3: Compute the canonical curve E_m having j-invariant j_m.
4: Let $= \ '_m \quad '\hat{}_m + [d] \in \mathrm{End}(E_m)$.
5: Compute as described in the proof of Theorem 6.1.7.
6: Compute $\ker('_m \quad '\hat{}_m) \cap E_m[D] = \ker(\quad [d]) \cap E_m[D] = \ker('\hat{}_m)$.
7: Compute $\ker('_m)$ using $\ker('\hat{}_m)$.
8: return $m \in [\ (D)]$ that corresponds to $\ker('_m)$.

6.2.3 Inversion

In this section, we concretely show how to use methods from [QKL+21] to invert a given function $f_{j_T}^{D,N} \in F_T^{p,D,N}$ with trapdoor information T. We assume that D is odd and that $\gcd(D, N) = 1$. We take E_{j_T} a supersingular curve inside E_O, where O is the quadratic order $\mathbb{Z}[\ (N^2e \quad d^2)/D^2]$ for some integers d, e. We write for the endomorphism of $\mathrm{End}(E_{j_T})$ of trace 0 and norm $(N^2e \quad d^2)/D^2$ such that $\mathbb{Z}[\ ] = O$. Let us also take a basis P_{j_T}, Q_{j_T} of $E_{j_T}[N]$. If we define T as $e, d, P_{J_T}, Q_{j_T}, \ (P_{j_T}), \ (Q_{j_T})$, then E_{j_T}, T is a (D, N)-trapdoor curve as produced in Theorem 6.1.10.

To make the inversion mechanism efficient on all inputs, we require the additional condition that the discriminant of O is a quadratic non-residue modulo every prime divisor of D. The concrete statement can be found in Lemma 6.2.5. We explain how to generate $E_{j,T}$, O and T in Sections 6.3.1 and 6.3.2.

We are given (j_m, P_m, Q_m) as the output of $f_{j_T}^{D,N}$ for some input m, which we want to recover. Let the isogeny corresponding to m be denoted by $'_m$. We assume that $P_m = '_m(P_{j_T})$ and $Q_m = '_m(Q_{j_T})$. Let $:= '_m \quad '\hat{}_m + [d]$ and let $G := \ker(\quad [d]) \cap E_m[D]$.

Lemma 6.2.5. *If $= \mathrm{disc}(\ O)$ is a non-quadratic residue, the group G is cyclic and equal to $\ker('\hat{})$.*

Proof. It is clear that $\ker('\hat{}_m) \quad G$ since it is contained in $\ker('_m \quad '\hat{}_m)$ and in $E_m[D]$ as well. We now show that G is cyclic. Let M be the largest divisor of D such that $E_m[M] \quad G$. Then $'_m$ can be decomposed as $'_{D/M} \quad '_M$. Then by [Pet17, Lemma 5] the kernel of $'_M$ is fixed by . The proof [Pet17, Lemma 6] shows that a subgroup of $E_{j_T}[M]$ can only be fixed by an endomorphism if $\mathrm{Tr}( )^2 \quad 4\deg( ) = \mathrm{disc}(\mathbb{Z}[\ ]) =$ is a square modulo M. Thus, the quadratic residuosity condition on ensures that $M = 1$, which implies that G is cyclic. The order of G is a divisor of D since G is cyclic and every element of G has order dividing D. However, G contains $\ker('\hat{}_m)$, which is a group of order D. This implies that $G = \ker('\hat{}_m)$. □ □

The group $G = \ker(\hat{})$ can be computed by solving a double discrete logarithm problem, which is efficient as D is smooth. We summarize the steps needed for inverting the one-way function in Algorithm 35.

In [QKL+21] it is shown that Algorithm 35 runs in polynomial time whenever $E_m[D]$ is efficiently representable and $\Delta = \mathrm{disc}(\mathbb{Z}[\pi])$ is as in Lemma 6.2.5.

6.2.4 Seta Public Key Encryption

We now build Seta, a Public Key Encryption scheme using the trapdoor one-way function family of Section 6.2.2, and we show that it is OW-CPA secure. Concretely, we define the Seta PKE scheme as the tuple (keygen, enc, dec) of PPT algorithms described below.

Parameters. Let λ denote the security parameter. Let p be a prime such that $p^2 - 1 = DNf$ where D, N are smooth integers and f is a small co-factor such that $2^{2\lambda} < D$, $D^2 < N$. We let $\mathrm{param} = (\lambda; p; D; N)$.

Key generation. The $\mathrm{keygen}(\mathrm{param})$ algorithm proceeds as follows:

1. Compute a uniformly random $(D; N)$-trapdoor supersingular elliptic curve $(E_{j_T}; T)$ defined over $\mathbb{F}_{p^2}$ using SetaQuadraticOrderGen and SetaCurveGen (see Section 6.3).
2. Set $\mathrm{pk} := (j_T)$ and $\mathrm{sk} := T$.
3. Return $(\mathrm{pk}; \mathrm{sk})$.

Encryption. The $\mathrm{enc}(\mathrm{param}; \mathrm{pk}; m)$ algorithm proceeds as follows. For a given $m \in \{0,1\}^{n_m}$, where $n_m = \lfloor \log_2 \Delta(D) \rfloor$, first cast m as an integer in the set $[\Delta(D)]$ and then:

1. Parse $\mathrm{pk} = j_T \in \mathcal{J}_{T;p}$.
2. Compute $(j_m; P_m; Q_m) \leftarrow f_{j_T}^{D;N}(m)$.
3. Return $c = (j_m; P_m; Q_m)$.

Decryption. The $\mathrm{dec}(\mathrm{param}; \mathrm{pk}; \mathrm{sk}; c)$ algorithm proceeds as follows:

1. Given param, sk and c, parse c as $(j_c; P_c; Q_c) \in \mathbb{F}_{p^2} \times (\overline{\mathbb{F}_{p^2}})^2 \times (\overline{\mathbb{F}_{p^2}})^2$; if that fails, return $\perp$.
2. Apply InverseTrapdoor to recover $m' \in [\Delta(D)]$; if this fails, set $m' = \perp$.
3. If $\perp$ was recovered, return $\perp$.
4. Otherwise, from $m' \in [\Delta(D)]$, recover $m \in \{0,1\}^{n_m}$ and return it.

Theorem 6.2.6. Let p be a prime, let D and N be integers such that $D^2 < N$. Suppose that the output distribution of Algorithm 37 is statistically close to uniform. Let E_{j_T} be an output of Algorithm 37. If Problem 6.2.3 with $p; D; N; E_{j_T}$ and X such that $H_1(X) = \lambda$ is hard for quantum PPT adversaries, then the PKE scheme above is quantum OW-CPA secure.

Proof. Let $M = \{0,1\}^{n_m}$ denote the message space of the encryption scheme, with $n_m = O(\lambda)$. We see that a randomly sampled $m \xleftarrow{\$} M$ directly embedded as an integer $m \in [\Delta(D)]$ yields a distribution Y with min-entropy $H_1(Y) \geq \lambda$ on isogenies of degree D starting from E_{j_T}. The challenge of opening a given ciphertext c then reduces to recovering the secret isogeny of Problem 6.2.3 with $X = Y$. □ □

A ciphertext is composed of a j-invariant $j_c \in \mathbb{F}_{p^2}$, which can be represented with $2\log p$ bits, and two torsion points $P_c, Q_c \in E_{j_c}[N]$, each of which can be represented with $2\log N$ bits by identifying each N-torsion point with a pair of elements in $\mathbb{Z}_N$. Therefore, the bit size of a ciphertext is

$$2\log p + 4\log N.$$

Further compression is possible, representing both torsion points with $3\log N$ bits, using the techniques in [CJL+17, Section 6.1].

6.2.5 IND-CCA encryption scheme

We obtain an IND-CCA secure PKE scheme by applying the generic post-quantum OAEP transformation [TU16, Section 5] (detailed at the end of this section) to Seta, for which we prove that our function $f_{j_T}^{D,N}$ is quantum partial-domain one-way.

Definition 6.2.7. Let k_1, k_0 and n_c be integers. A family F of functions $f : \{0,1\}^{\lambda+k_1} \times \{0,1\}^{k_2} \to \{0,1\}^{n_c}$ is partial domain one-way if for any polynomial time adversary A, the following advantage is negligible in λ:

$$\mathrm{Adv}_\lambda(A) = \Pr\left[s' = s, s' \neq 0 \;:\; A(1^\lambda, y), y = f(s,t), (s,t) \xleftarrow{\$} A^{B, f \leftarrow F}\right]$$

Lemma 6.2.8. Let j_T be a uniformly random element of $J_{T,p}$. The function $f_{j_T}^{D,N}$ defined in Section 6.2.2 is a quantum partial-domain one-way function, under the hardness of Problem 6.2.3.

Proof. We note that in our case, partial domain inversion is the same as domain inversion where only the first part of the path is required. More precisely, factor D as $D_1 \cdot D_2$ such that $\gcd(D_1, D_2) = 1$, $2^{\lambda+k_1} \leq \ell(D_1)$ and $2^{k_0} \leq \ell(D_2)$ (where $\lambda + k_0 + k_1$ is the bit-length of input strings) and then embed each of s and t into $\ell(D_1)$ and $\ell(D_2)$ respectively. Then we can set $f_{j_T}^{D,N}(s,t) := f_{j_1}^{D_2,N}(t)$ where $(j_1, P_1, Q_1) = f_{j_T}^{D_1,N}(s)$ and $f_{j_1}^{D_2,N}$ uses $\{P_1, Q_1\}$ as basis of $E_{j_1}[N]$. Since $2^{\lambda+k_1} \leq \ell(D_1)$, then recovering s from $y = f_{j_T}^{D,N}(s,t)$ is hard under the same assumption as Theorem 6.2.6 with D replaced by D_1. □ □

Theorem 6.2.9 ([TU16], Theorem 2). If $f_{j_T}^{D,N}$ is a quantum partial-domain one-way function, then the OAEP-transformed scheme is IND-CCA secure in the QROM.

The IND-CCA version adds an output of a hash function H', which has the same size as the input of the one-way function f. Thus, the total bit size of the ciphertext is

$$2\log p + 4\log N + k.$$

Post-quantum OAEP transformation. We present here the post-quantum OAEP generic transformation we use for SETA. Let

$$f : \{0,1\}^{\lambda+k_1} \times \{0,1\}^{k_0} \to \{0,1\}^{n_c}$$

be an invertible injective function. The function f is the public key of the scheme, its inverse f^{-1} is the secret key. The scheme makes use of three hash functions

$$G : \{0,1\}^{k_0} \to \{0,1\}^{k-k_0};$$
$$H : \{0,1\}^{k-k_0} \to \{0,1\}^{k_0};$$
$$H' : \{0,1\}^{k} \to \{0,1\}^{k};$$

modelled as random oracles, where $k = n + k_0 + k_1$. Given those, we define the encryption scheme as follows:

- Enc: given a message $m \in \{0,1\}^n$, choose $r \xleftarrow{\$} \{0,1\}^{k_0}$ and set

$$s = m\|0^{k_1} \oplus G(r); \qquad t = r \oplus H(s);$$
$$c = f(s,t); \qquad d = H'(s\|t);$$

  and output the ciphertext (c,d).

- Dec: given a ciphertext (c,d), use the secret key to compute $(s,t) = f^{-1}(c)$. If $d \neq H'(s\|t)$ output $\perp$. Otherwise, compute $r = t \oplus H(s)$ and $\overline{m} = s \oplus G(r)$. If the last k_1 bits of $\overline{m}$ are 0, output the first n bits of $\overline{m}$, otherwise output $\perp$.

6.3 Key generation method

In this section, we describe how to generate keys for Seta. We first describe SetaQuadraticOrderGen, as Algorithm 36, to generate integers d, e from which we can derive a quadratic order $\mathcal{O} = \mathbb{Z}[\frac{N^2 e - d^2}{D^2}]$ that satisfies the quadratic residuosity conditions imposed in Section 6.2.3 to make a suitable candidate for Seta.

Then, we present how to generate a uniformly random supersingular elliptic curve inside $E_{\mathcal{O}}(p)$, together with the remaining part of the trapdoor information T. This is done with SetaCurveGen, described as Algorithm 37.

6.3.1 Computing the trapdoor information

We recall that the required condition is that $n = \frac{N^2 e - d^2}{D^2}$ must be a quadratic non-residue modulo every prime dividing D and also modulo p. For simplicity, we fix $e = 1$ and look for d of a special form. We show how to find d in SetaQuadraticOrderGen (described as Algorithm 36).

Lemma 6.3.1. *If (d,e) is the output of SetaQuadraticOrderGen, then $n = \frac{N^2 e - d^2}{D^2}$ is a quadratic non-residue modulo all ℓ_i.*

Proof. Let r_i, s_{ℓ_i}, T and u be as in SetaQuadraticOrderGen. Let r be an integer such that $r \equiv r_i \pmod{\ell_i}$. Then we show that for every i, the integer $\frac{N^2 e + (D^2 r + u)^2}{D^2}$ is not a quadratic residue modulo ℓ_i, which implies that $\frac{N^2 e - d^2}{D^2}$ is not a quadratic residue modulo every ℓ_i since $T\ell + r \equiv r_i \pmod{\ell_i}$ for every integer ℓ. We have that

$$\frac{N^2 e + (D^2 r + u)^2}{D^2} = \frac{N^2 e + u^2}{D^2} + D^2 r^2 + 2ur$$

Algorithm 36 SetaQuadraticOrderGen(D, N)

Input: D, N as above. Let S be the product of primes dividing D.

Output: (d, e) such that $\frac{N^2e - d^2}{D^2} < 0$ is a quadratic non-residue modulo every prime dividing D and is a quadratic non-residue modulo p.

1: Set $e = 1$.
2: Find u such that $u^2 \equiv N^2e \pmod{D^2}$.
3: **for** every prime ℓ_i dividing D **do**
4: Let s_{ℓ_i} be a quadratic non-residue modulo ℓ_i.
5: $r_i \leftarrow (s_{\ell_i} - \frac{N^2e+u^2}{D^2})(2u)^{-1} \pmod{\ell_i}$.
6: **end for**
7: Compute a residue r modulo S with the property that $r \equiv r_i \pmod{\ell_i}$.
8: $c \leftarrow 0$.
9: $d \leftarrow D^2(Sc + r) + u$.
10: $A \leftarrow \frac{N^2e - d^2}{D^2}$.
11: **if** $A < 0$ **then**
12: **return** $?$
13: **end if**
14: **if** A is not a square modulo p **then**
15: $c \leftarrow c + 1$.
16: **go to** Step 9.
17: **end if**
18: **return** (d, e)

. By our choice of r, we have that

$$\frac{N^2e+u^2}{D^2} + D^2r^2 + 2ur \equiv \frac{N^2e+u^2}{D^2} + 2ur_i \equiv s_{\ell_i} \pmod{\ell_i};$$

which is a quadratic nonresidue by the choice of s_{ℓ_i}. □ □

Lemma 6.3.2. (UPHA) Let S be the product of all primes dividing D. If $N > D^2S$, then SetaQuadraticOrderGen returns a correct pair (d, e) with probability higher than $1 - 2^{\frac{N}{SD^2}+1}$.

Proof. Since u is found by solving an equation modulo D^2, we obtain $u < D^2$. Similarly, we have $r < S$. Under plausible heuristic assumptions, we can estimate to 1/2 the probability that the quadratic residuosity condition on A is satisfied. Thus, we obtain a bound on the failure probability by counting how many values ℓ can be tried before A becomes negative. With the conservative bound that $D^2r + u \leq D^2S$, we obtain that we can try $\frac{N - D^2S}{DS^2}$ different values for small d, which gives the result.

Correctness of the result follows from Lemma 6.3.1. □

6.3.2 Trapdoor curve generation

Now we focus on generating a random supersingular elliptic curve whose endomorphism ring contains an embedding of $\mathcal{O} = \mathbb{Z}[\sqrt{n}]$ where $n = (N^2e - d^2)/D^2$ with d, e the outputs of SetaQuadraticOrderGen. In [QKL+21, Section 5.1] it is discussed how one can generate a specific curve inside $\mathcal{E}_0(p)$. Essentially,

this is achieved by computing a maximal order $\mathcal{O}$ containing the suborder $\mathcal{O}$ (with [Voi13, Algorithm 7.9]) and then computing a supersingular elliptic curve whose endomorphism ring is isomorphic to $\mathcal{O}$ (with [EHL+18, Algorithm 12]). This procedure can be made concretely efficient with our efficient ideal-to-isogeny algorithms (IdealToIsogenyFromKLPT or IdealToIsogenyFromEichler) under some conditions on the prime p that partly underlie the choice of prime described in Section 6.4.2. As usual, $\mathcal{O}_0$ is the special extremal order of $B_{p,1}$ and E_0 is the corresponding supersingular curve. However, this procedure is essentially deterministic, so an adversary knowing the quadratic order $\mathcal{O}$ can just recompute the same trapdoor curve. The point of this subsection is to show how to randomize the procedure.

We obtain randomization by first generating a curve with the deterministic procedure, and then applying the action of a random class group element to derive another random curve with the same embedding. This operation would be costly if it required to compute a lot of isogenies. However, we can do it over the quaternions at a negligible cost before applying the translation algorithm from maximal orders to elliptic curves.

For concrete randomization, we use the fact (see [JMV09]) that there exists a bound B (polynomial in p) for which the graph whose vertices are curves in $\mathcal{E}_{\mathcal{O}}(p)$ and edges are isogenies of prime degree smaller than B is an expander graph. The fast mixing property of expander graphs implies that the distribution of curves obtained after a random walk of fixed length quickly converges to the uniform distribution as the length of the walk grows. More precisely, for any ε, we can find a length ℓ (logarithmic in the size of the graph and ε) for which the statistical distance between the random walk distribution and the uniform distribution is less than ε. So once we fix the length ℓ (corresponding to a sufficiently small ε), for any starting curve E_0 in $\mathcal{E}_{\mathcal{O}}$, the curve $\left(\prod_{i=1}^{n} \mathfrak{l}_i^{e_i}\right) \star E_0$ where $\mathfrak{l}_1, \ldots, \mathfrak{l}_n$ are prime ideals above the n prime $\ell_1, \ldots, \ell_n$ smaller than B that are split in $\mathcal{O}$ and $(e_1, \ldots, e_n)$ is uniformly random among the vectors in $\mathbb{Z}^n$ such that $\sum_{i=1}^{n} |e_i| = \ell$, is statistically close to a uniformly random element in $\mathcal{E}_{\mathcal{O}}$. This result underlies SetaCurveGen.

Proposition 6.3.3. *SetaCurveGen is correct and terminates in polynomial time.*

Proof. All the sub-algorithms run in polynomial time and, by choice of B and ℓ, the number of iterations in the loop is also polynomial.

It is easy to verify that the ideal I corresponds through the Deuring correspondence to the isogeny φ_{I_i}. Thus, our method simulates a random walk over the graph that we described at the beginning of this section. For the reasons explained there, the curve E_{j_τ} obtained in the end is statistically close to a random element in $\mathcal{E}_{\mathcal{O}}(p)$. □ □

6.3.3 Constraints on the prime

In Seta, we compute and evaluate isogenies of degree D and N. Hence, we always require that D and N are smooth and that the DN-torsion groups are efficiently representable, i.e., that they are defined on extensions of $\mathbb{F}_{p^2}$ of small degree. For example, if we require that $E[DN] \subset E(\mathbb{F}_{p^4})$, then DN must divide $p^2 - 1$. The smoothness bound B_1 of D impacts the efficiency of encryption, and the smoothness bound B_2 of N impacts the efficiency of decryption. For a

Algorithm 37 SetaCurveGen(p, N, O, B, ε)

Input: A prime p, an integer N, a quadratic order $\mathcal{O}$, a bound B, a length ε.
Output: A uniformly random curve $E_{j_T} \in \mathcal{E}_{\mathcal{O}}(p)$, a basis P_{j_T}, Q_{j_T} of $E_{j_T}[N]$, and (P_{j_T}), (Q_{j_T}) with $\theta \in \mathrm{End}(E_{j_T})$ such that $\mathbb{Z}[\theta] = \mathcal{O}$.

1: Find a max. order $\mathcal{O}' \supset \mathcal{B}_{p,1}$ with $\mathcal{O}$ embedded in $\mathcal{O}'$ with the alg. from [QKL+21].
2: Compute $\ell_1, \ldots, \ell_n$ the n primes split in $\mathcal{O}$ smaller than B.
3: Select a random vector $(\varepsilon_1, \ldots, \varepsilon_n)$ in $\mathbb{Z}^n$ with L_1 norm equal to ε.
4: Set $\mathcal{O}_{j_T} = \mathcal{O}$.
5: **for** $1 \le i \le n$ **do**
6: Compute $\pi_i \in \mathcal{O}$ such that $I_i = \mathcal{O}\langle \pi_i, \ell_i \rangle$ is a prime ideal above ℓ_i.
7: **for** $1 \le j \le |\varepsilon_i|$ **do**
8: Compute the ideal $I = \mathcal{O}_{j_T}\langle \pi_i, \ell_i \rangle$.
9: Set $\mathcal{O}_{j_T}$ as the right order of I.
10: **end for**
11: **end for**
12: Compute $J = \mathrm{ConnectingIdeal}(\mathcal{O}_0, \mathcal{O}_{j_T})$ and select ℓ a small constant prime coprime to N and $\mathrm{disc}(\mathcal{O})$.
13: Compute $K = \mathrm{KLPT}_{\ell}(J)$.
14: Compute $\varphi = \mathrm{IdealToIsogenyFromKLPT}(K, \mathcal{O}_0, [1]_{E_0})$.
15: Set E_{j_T} as the codomain of φ.
16: Compute a canonical basis P_{j_T}, Q_{j_T} of $E_{j_T}[N]$.
17: Select the correct element $\theta \in \mathcal{O}_{j_T}$ such that $\mathcal{O} = \mathbb{Z}[\theta]$.
18: Use $\mathcal{O}_{j_T}$ and φ to compute (P_{j_T}), (Q_{j_T}).
19: **return** $E_{j_T}, P_{j_T}, Q_{j_T}$ of $E_{j_T}[N]$; (P_{j_T}); (Q_{j_T}).

given security level λ, we require $2^{2\lambda} < D$ in order to protect the scheme against the meet-in-middle attack.

Since we have the range $D^2 < D^2 S < D^3$ depending on the value of S (product of primes dividing D), and that Lemma 6.3.2 implies that $N > D^2 S$, then we can estimate that the value DN will be between 2^6 and 2^8. If we want DN dividing $p^2 - 1$, we can estimate that the minimum size for the prime p will be between 3 and 4 bits. The actual size will depend on the size of $(p^2 - 1)/DN$.

Besides encryption and decryption, key generation also restricts the types of primes to be used in Seta because we use IdealToIsogenyFromKLPT (when [DFFdSG+21] was published, the algorithm IdealToIsogenyFromEichler has not been discovered and so we used IdealToIsogenyFromKLPT instead) and so we have the same constraints as in SQISign in Chapter 5. We will look for a prime of form $p^2 - 1 = \ell^f T f_2$, where ℓ is the small prime that we are going to use in SetaCurveGen, $T > p^{3/2}$ is a smooth integer co-prime to ℓ and f_2 is a cofactor.

6.4 Implementation

We made a C implementation of Seta. Our code is available at https://github.com/seta-isogeny-encryption/seta. It reuses large parts of the code base of

SQISign[2] for the Key generation.

6.4.1 Main building blocks

Key generation consists of two parts. An execution of SetaQuadraticOrderGen and an execution of SetaCurveGen. The difficult part of this procedure in practice is a subroutine for finding a supersingular elliptic curve whose endomorphism ring is isomorphic to a particular maximal order $\mathcal{O}$. For this step, we reused a substantial amount of the code used for SQISign to implement the IdealToIsogenyFromKLPT algorithm.

Encryption consists in the evaluation of an isogeny of degree D at points of order N. In order to make this efficient, we choose parameters where D has small prime factors and both D and N divide $p^2 - 1$ to avoid using extension fields.

Decryption also uses evaluations of isogenies, but for isogenies of degree N. Furthermore, decryption requires some linear algebra modulo D (when computing the intersection $\ker(\varphi[d]) \cap E_m[D]$) and modulo N (when computing the isogenies φ and φ'). In these steps one uses subroutines for solving discrete logarithms but due to N and D being smooth, this step is negligible compared to other computations.

6.4.2 Prime search

To efficiently implement Seta, it is necessary to select a prime satisfying the many constraints mentioned in Section 6.3.3. To maximize efficiency of encryption and decryption, while maintaining reasonably efficient key generation, we opted to search for a prime satisfying the following constraints: (1) $p^2 - 1 = DN$, with both D and N smooth; (2) $D \approx 2^2$ and $N \approx 2^4$; and (3) D has as few prime factors as possible.

This search is quite similar to the one we made for SQISign, but the constraint are slightly different here (in terms of size of p and the amount of smooth torsion required). This is why, contrary to SQISign, the method that gave the best result is the one by Costello from [Cos19] where we look for primes of the form $p = 2x^n - 1$. Thanks to this technique, D can be taken as a factor of $p + 1$, and has thus much fewer prime factors than a generic smooth prime of the same size. The drawback of the technique is that, as n increases, the search space decreases, to the point where we cannot find smooth integers anymore.

Concretely, for $\lambda = 128$, we need to have $\log p \approx 400$. So we fixed $n = 12$ and we sieved within the space $2^{32} < x < 2^{33}$, i.e., $2^{385} < p < 2^{397}$. This yielded four primes with the largest factor bounded by 2^{25}, and three with bound 2^{26}, corresponding to $x = 4679747572$, 4845958752, 4966654633, 5114946480, 6334792777, 8176556533, 8426067021. Unfortunately, we fully explored the search space, meaning that no better primes exist for $n = 12$.

The relatively large smoothness bounds negatively affect performance of all algorithms in Seta. Unfortunately, it appears to be difficult to find better primes given current knowledge. Even dropping the constraint on the number of prime factors of D, the best algorithms known today can hardly beat a 2^{20} smoothness bound for a prime of 384 bits [CMN21, Table 3].

[2] https://github.com/SQISign/sqisign

Remark 6.4.1. [DFFdSG+21] was published before [DFLW22] and this is why we use IdealToIsogenyFromKLPT rather than IdealToIsogenyFromEichler. Given that the size of the prime is much bigger in Seta than for SQISign, the relaxed requirement of torsion in IdealToIsogenyFromEichler should allow us to select torsion with much better torsion, and so we expect a huge improvement if we use IdealToIsogenyFromEichler instead. We leave this to future work.

6.4.3 Experimental results

We ran experiments on a 4.00GHz Quad-Core Intel Core i7, using a single core. We used the prime $p = 2^{12} \cdot 842606702^{12} - 1$, and the smooth factors

$$D = 43^{12} \cdot 84719^{11};$$

$$N = 3^{21} \cdot 5 \cdot 7 \cdot 13 \cdot 17 \cdot 19 \cdot 23 \cdot 73 \cdot 257^{12} \cdot 313 \cdot 1009 \cdot 2857 \cdot 3733 \cdot 5519 \cdot 6961 \cdot 53113 \cdot 499957 \cdot 763369 \cdot 2101657 \cdot 2616791 \cdot 7045009 \cdot 11959093 \cdot 17499277 \cdot 20157451 \cdot 33475999 \cdot 39617833 \cdot 45932333$$

We run the key generation only once, and took 1043 hours. The encryption procedure took 463 seconds, and the decryption took 1066 minutes, averaged over six runs. The decryption time is almost entirely devoted to the evaluation of isogenies with degrees among the largest factors of N.

6.5 \Uber" isogeny assumption

In this section, we introduce a generic framework, which we label Uber Isogeny assumption in analogy to [Boy08], aiming at generalizing isogeny computation problems encountered in the main families of isogeny-based schemes such as SIDH [JD11], CSIDH [CLM+18], OSIDH [CK19] and Seta (presented in this work).

The Uber isogeny problem does not directly underlie the security of these various schemes (in the sense that no formal reduction is yet known). However, for each of these protocols there exists a set of parameters for which if one can solve the Uber isogeny problem, then one can break the scheme. At a higher-level, our new problem can be seen as a generic key recovery problem.

By introducing this new assumption, our goal is twofold. First, we highlight the proximity between the various isogeny schemes, and we provide a common target for cryptanalysis. Second, the generic attack that we describe in Section 6.5.3 gives a lower-bound on the security of any future scheme whose security may be related to our uber assumption similarly to SIDH, CSIDH, OSIDH and Seta.

6.5.1 The new generic problem

The principal mathematical structure behind the uber isogeny problem is the group action on the curves of $E_O(p)$ that emerge through the class group $Cl(O)$. We introduced the main definitions and properties about this topic in Section 2.4.

Problem 6.5.1 (O-Uber Isogeny Problem (O UIP)). Let $p > 3$ be a prime and let O be a quadratic order of discriminant . Given $E_0; E_s \in E_O(p)$ and

an explicit embedding of $\mathcal{O}$ into $\mathrm{End}(E_0)$ (i.e., the knowledge of $\omega_0 \in \mathrm{End}(E_0)$ such that $\mathbb{Z}[\omega_0] = \mathcal{O}$), find a powersmooth ideal $\mathfrak{a}$ of norm coprime to ℓ such that $[\mathfrak{a}] \in \mathrm{Cl}(\mathcal{O})$ is such that $E_s = \mathfrak{a} \star E_0$.

Remark 6.5.2 In Problem 6.5.1, the powersmoothness condition on the norm is to ensure that the resulting isogeny can always be computed in polynomial time. In some special cases where the form of the prime p enables to compute some smooth isogenies in polynomial time, this condition might be relaxed a little.

6.5.2 Relation with various isogeny-based constructions

We start with the link with CSIDH [CLM+18] (see Section 6.1.2) which is quite obvious.

Proposition 6.5.3. When $p = 3 \bmod 4$ and $\ell = 4p$, Problem 6.5.1 is equivalent to the CSIDH key recovery Problem 6.1.4.

Proof. In the case of CSIDH, the curves admitting an embedding of $\mathbb{Z}[\sqrt{p}] = \mathbb{Z}[\pi]$ in their endomorphism rings are the curves defined over $\mathbb{F}_p$ (i.e left stable by the Frobenius morphism). Then, it is quite clear that Problem 6.5.1 is equivalent to Problem 6.1.4. □

The OSIDH protocol [CK19] is a generalization of CSIDH where $\mathbb{Z}[\pi]$ is replaced by a larger class of quadratic orders. The link between OSIDH and Problem 6.5.1 is also straightforward. Let us fix some notations[3] for this protocol and briefly recall the principle. The OSIDH key exchange protocol starts from a descending chain of ℓ-isogenies of size n that we write $\varphi_0 : F_0 \to E_0$ where F_0 admits an $\mathcal{O}_0$-orientation (i.e., an embedding of $\mathcal{O}_0$ inside $\mathrm{End}(E_0)$). From there, φ_0 induces an $\mathcal{O}$-orientation on E_0. The secret keys of Alice and Bob are $\mathcal{O}$-ideals $\mathfrak{a}, \mathfrak{b}$ whose action on E_0 will lead to curves $E_A = \mathfrak{a} \star E_0$ and $E_B = \mathfrak{b} \star E_0$. These curves have also an $\mathcal{O}$-orientation, which implies the existence of ℓ^n-isogenies $\varphi_A : F_0 \to E_A$ and $\varphi_B : F_0 \to E_B$, as in Proposition 2.4.25. Alice public key is E_A together with some torsion points (which allows Bob to compute $\mathfrak{b} \star E_A$).

Proposition 6.5.4. When $\mathcal{O}_0$ is a quadratic order of class number 1 and $\mathcal{O} = \mathbb{Z} + \ell^n \mathcal{O}_0$, then if there exists a PPT algorithm that can break Problem 6.5.1, there is a PPT algorithm that can recover the keys of the OSIDH protocol presented in [CK19].

Proof. From the definition of the group action of $\mathrm{Cl}(\mathcal{O})$ on the curves having an $\mathcal{O}$-orientation, finding a smooth ideal $\mathfrak{c}$ such that $E_A = \mathfrak{c} \star E_0$ is enough to recover the secret key. □

Note that we do not have equivalence in Proposition 6.5.4 because the OSIDH public keys include more information than just curves. This will be the same for SIDH and Proposition 6.5.5.

[3] These notations do not exactly agree with the ones introduced in [CK19] because we want to highlight the link with our $\mathcal{O}$-IOP.

For SIDH, we write[4] F_0 for the common starting curve. In SIDH, recovering the secret key from the public key is equivalent to the computational supersingular isogeny problem (CSSI) that we introduced in Section 6.1.1 as Problem 6.1.1

The proposition below requires a bit more work, as the link between SIDH and group actions is less obvious.

Proposition 6.5.5. Assume that F_0 admits an $\mathcal{O}_0$-orientation with $\mathcal{O}_0$ a maximal quadratic order of class number 1. If there exists a PPT algorithm solving Problem 6.5.1 for $\mathcal{O} = \mathbb{Z} + N_A' \mathcal{O}_0$ where N_A' divides N_A, then there exists a PPT algorithm that breaks the CSSI problem with overwhelming probability.

Proof. First, note that N_A is chosen so that the kernel points of A-isogenies have a polynomial-size representation. Then, since N_A is also smooth, the discrete logarithms can be solved in polynomial time in the A-torsion and isogenies of degree A can be computed in polynomial time.

For the rest of this proof, let us write ι the endomorphism of F_0 such that $\mathbb{Z}[\iota]$ realizes the embedding of $\mathcal{O}_0$ inside $\mathrm{End}(F_0)$.

If the curve E_A is N_A-isogenous to F_0, then E_A admits an embedding of $\mathbb{Z} + N_A\mathcal{O}_0$. This embedding is not necessarily primitive, but we know there exists N_A' dividing N_A such that $\mathcal{O} = \mathbb{Z} + N_A'\mathcal{O}_0$ admits a primitive embedding in $\mathrm{End}(E_A)$ (see Proposition 2.4.25). Conversely, since the class number of $\mathcal{O}_0$ is 1, then any $\mathbb{Z} + N_A'\mathcal{O}_0$-orientation on E_A implies the existence of an A'-isogeny between E_A and F_0. Let us write $\varphi_{A'} : F_0 \to E_A$ this isogeny of degree N_A'. Then φ_A, the secret isogeny in Problem 6.1.1 is the composition of $\varphi_{A'}$ with an endomorphism α_A of $\mathcal{O}_0$ of degree N_A/N_A'. Since N_A/N_A' is a power of ℓ_A, there are two possibilities for α_A. Thus, the difficulty lies in recovering $\varphi_{A'}$.

We can generate a curve E_0 in $E_{\mathbb{Z}+N_A'\mathcal{O}_0}$ by generating $\varphi_0 : F_0 \to E_0$ a descending isogeny of degree N_A'. Any ideal $\mathfrak{a}$ such that $E_A = \mathfrak{a} \star E_0$ can be interpreted as an isogeny $\varphi_{\mathfrak{a}} : E_0 \to E_A$ of degree $n(\mathfrak{a})$. We conclude the proof with the fact that $\ker \hat{\varphi}_{A'} = \varphi_{\mathfrak{a}}(\ker \hat{\varphi}_0)$, which we prove below. Once $\ker \hat{\varphi}_{A'}$ has been computed, it is easy to recover $\ker \varphi_{A'} = \hat{\varphi}_{A'}(E_A[N_A'])$ and find a solution to the CSSI as we explained above.

To prove $\ker \hat{\varphi}_{A'} = \varphi_{\mathfrak{a}}(\ker \hat{\varphi}_0)$, we need to understand how the fact that $\mathfrak{a}$ is an $\mathcal{O}$-ideal translates to the action of $\varphi_{\mathfrak{a}}$ on $\hat{\varphi}_0$. As explained in Proposition 2.4.25 and the following paragraph, the embedding of $\mathcal{O}$ in E_0 (resp. E_A) is obtained as $\mathbb{Z}[\varphi_0 \circ \hat{\varphi}_0] = \mathbb{Z}[\alpha_0]$ (resp. $\mathbb{Z}[\varphi_{A'} \circ \hat{\varphi}_{A'}] = \mathbb{Z}[\alpha_{A'}]$). By definition of $\mathfrak{a}$ being an $\mathcal{O}$-ideal, we have that $\varphi_{\mathfrak{a}}(\ker \alpha_0) = \ker \alpha_A$. Thus, we need to prove that $\ker \alpha_0 \cap E_0[N_A'] = \ker \hat{\varphi}_0$ and $\ker \alpha_{A'} \cap E_A[N_A'] = \ker \hat{\varphi}_A$ (note that this property is exactly what underlies the inversion mechanism in Section 6.2.3). We will do it for α_0, the property for $\alpha_{A'}$ holds for the exact same reasons. It is clear from the definition of $\alpha_0 = \varphi_0 \circ \hat{\varphi}_0$ that we have $\ker \hat{\varphi}_0 \subset \ker \alpha_0$. Let us take $P \in E_A[N_A'] \cap \ker \hat{\varphi}_0$, then $Q = \hat{\varphi}_0(P) \in \ker \varphi_0 \cap \langle 0 \rangle$. If we assume that $P \in \ker \alpha_0$, it implies that $\iota(Q) \in \ker \varphi_0$. Since $\ker \varphi_0$ is cyclic, we have that $\iota(Q) = \lambda Q$ for some $\lambda \in \mathbb{Z}$. This contradicts the fact that φ_0 is descending. Indeed, if we write φ_Q, the isogeny whose kernel is generated by Q, we have $\varphi_0 = \lambda \circ \varphi_Q$ for some isogeny φ_Q and the condition $\iota(Q) = \lambda Q$ implies that φ_Q is not descending and so φ_0 would not be descending, which is a contradiction.

[4] In Proposition 6.5.5, we use this notation and note the one introduced in Section 6.1.1 in order to highlight the link with Problem 6.5.1

Thus, we have proven that $\ker \varphi_0 \cap E_0[N_A^0] = \ker \hat{\varphi}_0$ and this concludes the proof as explained above. □

We refer to Section 6.2 for the full details and notations about Seta. We write $\mathcal{O} = \mathbb{Z}[\sqrt{-n}] = \mathbb{Z}[\pi]$ and assume that n is public. This assumption is plausible, as SetaQuadraticOrderGen is essentially deterministic.

Proposition 6.5.6. If there exists a PPT algorithm solving Problem 6.5.1 for $\mathcal{O}$, then there exists a PPT algorithm that takes a Seta public key E_s and recovers a trapdoor T such that $E_{j_T}; T$ is a (D, N)-trapdoor curve.

Proof. Let E_{j_T} be a Seta public key. By applying Algorithm 37 in $\mathcal{O}$ and adding the integers e, d, a (D, N)-trapdoor curve $E_0; T_0$ can be found in polynomial time with $E_0 \in \mathcal{E}_{\mathcal{O}}$. Thus, we can apply the PPT solver for Problem 6.5.1 on E_0 and E_{j_T} to compute an isogeny $\varphi_a : E_0 \to E_{j_T}$ corresponding to an $\mathcal{O}$-ideal $\mathfrak{a}$. If we write $\alpha_0 \in \mathrm{End}(E_0)$ and $\beta \in \mathrm{End}(E_{j_T})$ the endomorphisms such that $\mathcal{O} = \mathbb{Z}[\alpha_0] = \mathbb{Z}[\beta]$. Then, by definition of $\mathcal{O}$-ideals, we have that $\beta \circ \varphi_a = \varphi_a \circ \alpha_0$. So if $T_0 = e, d, P_0, Q_0, \alpha_0(P_0), \alpha_0(Q_0)$, then $T = e, d, \varphi_a(P_0), \varphi_a(Q_0), \varphi_a(\alpha_0(P_0)), \varphi_a(\alpha_0(Q_0))$ is such that $E_{j_T}; T$ is a (D, N)-trapdoor curve. □

We finish this section by proving that some instances of Problem 6.5.1 are related to the more generic isogeny problem of finding a smooth isogeny between any two supersingular curves (Problem 6.5.7 below). For that, it suffices to show that there exists some quadratic order that is embedded inside the endomorphism ring of any supersingular curve.

Problem 6.5.7. Let $p > 3$, be a prime number. Given E_1, E_2 two distinct supersingular curves over $\mathbb{F}_{p^2}$. Find $\varphi : E_1 \to E_2$, an isogeny of powersmooth degree.

Proposition 6.5.8. There is an absolute constant $c > 0$ such that the following holds. Let $\mathcal{O}$ be a quadratic order of conductor ℓ^e inside $\mathcal{O}_0$, a maximal quadratic order, such that ℓ is inert in $\mathcal{O}_0$, and $e \geq c \log_\ell(p)$. If there exists a PPT algorithm that can break Problem 6.5.1, then there is a PPT algorithm that breaks Problem 6.5.7.

Proof. From the fact that the ℓ-isogeny graph is Ramanujan, and the rapid mixing of non-backtracking random walks in expander graphs [ABLS07], we deduce that for $e = \Theta(\log_\ell(p))$, there exists a non-backtracking path of degree ℓ^e between any two supersingular curves in the graph.

In particular, if E_0 is any $\mathcal{O}_0$-orientable curve, there exists a cyclic isogeny of degree ℓ^e from E_0 to any other E, and since ℓ is inert in $\mathcal{O}_0$, this isogeny must be a sequence of descending isogenies. This implies that any E is $\mathcal{O}$-orientable. Thus, if we write E_1 and E_2, the two curves in the generic isogeny problem, then we can construct a middle curve E_0 with an explicit embedding of $\mathcal{O}$, then use the PPT algorithm to find paths between E_0, E_1 and E_0, E_2, and finally concatenate the two paths to obtain a path between E_1 and E_2 of powersmooth degree. □

Remark 6.5.9. Recently, in [Wes21], Wesolowski proved some reductions between our uber isogeny assumption and other hard problems. Among other things, he studies the link with the endomorphism ring problem.

6.5.3 Analysis of the uber isogeny assumption

In this section, we investigate the complexity of solving Problem 6.5.1. We are going to see that there are various special cases leading to various complexities.

We start by giving a generic estimate which can be seen as the worst case complexity.

A first upper bound: exhaustive search. The simplest method to solve Problem 6.5.1 is to apply an exhaustive search, for instance by selecting a set of small primes ℓ_i all split in $\mathcal{O}$ and trying all combinations of $\prod_i \mathfrak{l}_i^{e_i} \star E_0$ until one is isomorphic to E_s, where each $\mathfrak{l}_i$ is a prime ideal above ℓ_i. The expected running time of this algorithm is in $\tilde{O}(\#\mathcal{E}_{\mathcal{O}}(p))$.

Since we have $\#\mathcal{E}_{\mathcal{O}}(p) \geq h(\mathcal{O})$, the classical estimate $h(\mathcal{O}) = \tilde{\Theta}(\sqrt{p})$ gives a first upper-bound on the complexity to solve Problem 6.5.1. In particular, it shows that solving Problem 6.5.1 is easy when the discriminant is small. However, when grows, it is harder to estimate how this bound reflects on the actual complexity of the problem. The whole point of the results we presented in Section 2.4 was to get more precise estimates on the complexity of this problem. An effective lower bound on $\#\mathcal{E}_{\mathcal{O}}(p)$ can be obtained by combining Proposition 2.4.19 and Proposition 2.4.26.

There are some special cases for which we can be a bit more precise than that. For instance, when the discriminant are short, we saw Corollary 2.4.8 that proves we must have $\#\mathcal{E}_{\mathcal{O}}(p) = h(\mathcal{O})$ when the discriminant is smaller than p.

When the conductor of $\mathcal{O}$ is big enough, Proposition 2.4.26 shows that $\mathcal{E}_{\mathcal{O}}(p)$ must contain all supersingular curves. This is what we used in Proposition 6.5.8.

The case of CSIDH. The key recovery problem of CSIDH has received a lot of attention from the community [CLM+18, BS20, Pei20, CSCDJRH20] since it was the first scheme that naturally fits into this framework. In fact, there are improvements over the exhaustive search strategy in both the classical and quantum settings. The main ingredient behind these speed-ups is the ability for anyone to obtain a concrete embedding (through the Frobenius morphism) of $\mathcal{O} = \mathbb{Z}[\sqrt{-p}]$ inside $\mathrm{End}(E)$ for any $E \in \mathcal{E}_{\mathcal{O}}$. In particular, computing $a \star E$ becomes easy for any $E \in \mathcal{E}_{\mathcal{O}}$ when a has smooth norm. In the classical setting, this implies a quadratic speed-up over the generic exhaustive search by using a meet-in-the-middle technique (see [CLM+18]). In the quantum setting, the speed-up is even more radical, as it creates a malleability oracle (see [KMPW21]) that reduces CSIDH's security to an instance of the hidden shift problem which can be solved in quantum sub-exponential time as described in [Pei20, BS20] for instance.

Note that neither of these attacks can be used in the generic case, as it seems hard to obtain this malleability oracle for other group actions. For instance, in OSIDH [CK19] the public keys are made of a curve E and some torsion points to make possible the computation of $a \star E$ for some secret ideal a. These additional torsion points are not needed in CSIDH because they can be easily computed.

Smooth conductor inside a maximal quadratic order. A better algorithm also exists when the conductor f of $\mathcal{O}$ is smooth. By Proposition 2.4.25, there exists an isogeny of degree f between any curve $E \in \mathcal{E}_{\mathcal{O}}(p)$ and any curve in $\mathcal{E}_{\mathcal{O}_0}(p)$, where $\mathcal{O}_0$ is the quadratic maximal order containing $\mathcal{O}$. Let E_0, E_s

given by in an instance of Problem 6.5.1, and let us write $\varphi_0 : F_0 \to E_0$ and $\varphi_s : F_s \to E_s$ the two isogenies of degree f.

The alternative resolution method enumerates through all possible $F_s = \mathfrak{a}_0 \star F_0$ in $\mathcal{E}_{\mathcal{O}_0}(p)$, then tries to find φ_s of degree f. Since f is smooth, we can apply a meet-in-the-middle technique to reduce this part to $\mathcal{O}(\sqrt{f})$. Once $\varphi_s : F_s \to E_s$ and an $\mathcal{O}_0$-ideal $\mathfrak{a}_0$ such that $F_s = \mathfrak{a}_0 \star F_0$ has been found, we can compute an $\mathcal{O}$-ideal such that $E_s = \mathfrak{a} \star E_0$, as described in [CK19, Section 5.1].

If we write $\Delta = f^2 \Delta_0$, where Δ_0 is the fundamental discriminant of $\mathcal{O}_0$. The complexity of this algorithm is $\tilde{\mathcal{O}}(f\sqrt{|\Delta_0|})$, which is better than $\tilde{\mathcal{O}}(\sqrt{\Delta}) = \tilde{\mathcal{O}}(f\sqrt{|\Delta_0|})$.

Other cases. When we are not in one of the above cases, there is no known improvement over the exhaustive search (classically or quantumly). Thus, the presumed security entirely relies on the size of $\mathcal{E}_{\mathcal{O}}$. Our results in Section 2.4 give a lower bound on this number, but we explained that it do not scale well asymptotically. Nonetheless, it seems sufficient for some cryptographic application, as we exhibit in Section 6.5.4.

6.5.4 A numerical application to the parameters of SETA

In this section, we are going to use Proposition 2.4.19 to provide a lower bound on the complexity of attack against the Couveignes-Kohel-Robert isogeny problem, where $\mathcal{O}$ is the quadratic order used in our SETA encryption scheme. This will give a lower bound on the hardness of SETA key recovery, as explained above. More precisely, we are able to prove, under a few reasonable assumptions, that there exists a suitable choice of quadratic order $\mathcal{O}$ such that the brute-force recovery attack is hard enough for 128 bits of security with the parameters given in Section 6.4.

In Section 6.4, we did not describe a concrete value of $\Delta = (N^2 - d^2)/D^2$ because solutions can be found quite easily. Below, we computed one such solution where n is easy to factor so that we could compute the conductor of $\mathcal{O}$. For instance, we found the value:

$$n = 113 \cdot 337 \cdot 43913 \cdot 69522129914593554713466657355275000660185257908972492522431413808553767205401453148081325894556965991428307754649539266033342875068026023370667830770225304 57$$

Since, n is square free and equal to 1 mod 4, the order $\mathcal{O} = \mathbb{Z}[\sqrt{n}]$ is the ring of integer of $K = \mathbb{Q}(\sqrt{n})$. To derive the concrete lower-bound in Corollary 6.5.10 from Proposition 2.4.19, the classical lower bound on $h(\mathcal{O})$ stated in Eq. (1.2.3) and the upper-bound in Eq. (2.4.6) on the size of Δ.

Corollary 6.5.10. Let the values p, n be as above and $\mathcal{O} = \mathbb{Z}[\sqrt{n}]$. Assuming GRH, the size of $\mathcal{E}_{\mathcal{O}}(p)$ is bigger than 2^{269}.

Proof. Proposition 2.4.19 tells us that $\#\mathcal{E}_{\mathcal{O}}(p)$ is bigger than $(1/2)\min(A, B)$ where:

$$A = h(\mathcal{O}) \text{ and } B = \frac{h(\mathcal{O})^2}{3(4p + 4n + 1)} \frac{p}{\max_{0 < N < 4n^2/p} \mathcal{R}(N)}.$$

Assuming that our n is big enough for it to hold, we are going to use $A = h(O) > \frac{\sqrt{4n}}{24e} \frac{\sqrt{4n}}{\log\log(4n)} > 2^{270}$. To get a lower bound on B, it remains to get an upper bound on $\max_{0 \le N \le 4n^2/p} \ell(N)$.

We can compute this bound manually using Eq. (2.4.8). Indeed, it can be easily verified that $\ell(4n^2/p) \le 98 \le 74$ and so we can compute all

$$1 + \frac{\log(4n^2/p)}{k \log(k)}^k \quad \text{for all } 1 \le k \le 98.$$

As expected, the maximum is reached for $k = 98$ and we have that

$$\max_{0 \le N \le 4n^2/p} \ell(N) < 2^{105}.$$

Thus, using the bound on $h(O)$, we get that $B > 2^{279}$. So, under GRH and the assumption that n is big enough so that our simplification of the Littlewood bound hold, we get that

$$\#E_{Z[\sqrt{n}]}(p) > 2^{269}.$$

□

Chapter 7

Cryptographic applications of the suborder representation

In this chapter, we explore the possibilities o ered by the suborder representation introduced in Section 4.3. We start by motivating our approach in the paragraph below. Then, in Section 7.1, we argue that the suborder representation is not equivalent to the ideal representation under the hardness of a new computational problem. In Section 7.2, we introduce a new non-interactive key exchange based on this new hard problem. Finally, in Section 7.3 we mention other potential applications.

Limitations of the ideal representation for cryptographic applications.
Section 4.2 was dedicated to illustrate the collection of nice algorithms in our possession to handle the ideal representation and perform some computations from it. However, the existence of those e cient algorithms is not necessarily a good thing in the context of cryptography. Indeed, the bottom line is that an ideal I reveals pretty much everything there is to know about the corresponding isogeny $\varphi_I : E_1 \to E_2$ and the two curves E_1, E_2. With I we get all the isogenies connecting these two curves, and we can do pretty much anything with the norm equation algorithms from Chapter 3 to get nice degrees. Thus, there is not much hope to use an ideal representation as anything else than secret keys. Even as secret knowledge, ideal representations have their limitations. For instance, practical zero-knowledge proofs of ideal representation knowledge appear hard to obtain. Since we have an e cient veri cation algorithm, there are some generic constructions to obtain zero-knowledge proof systems for $\mathcal{L}_{\text{isog}}$ under standard cryptographic assumptions such as the existence of one way-functions [GMW91]. While this result is nice in theory, it is not really helpful to build a practical zero-knowledge proof-system for $\mathcal{L}_{\text{isog}}$.

One of the main motivations behind the introduction of the suborder representation is to address the shortcomings of the ideal representation in that regard. In particular, under the hardness of the new SOI problem (see Problem 7.1.1), we will see that it seems plausible to use suborder representations as public elements. This idea is the basis of pSIDH, the new NIKE scheme that

we introduce in Section 7.2. More generally, the gap between ideal and suborder representations open interesting cryptographical prospects, as we discuss in Section 7.3.

7.1 Deducing the ideal representation from the suborder representation

We saw with Proposition 4.3.2 that our new suborder representation can be computed from the ideal representation in polynomial time. The goal of this section is to study the reverse problem of extracting an ideal representation from a suborder representation. We are going to argue that this problem is hard. This supposed hardness and the resulting gap between the ideal and suborder representations motivates our new construction. Some applications discussed in Section 7.2 and Section 7.3 will specifically rely on the hardness of Problem 7.1.1.

Problem 7.1.1. (SubOrder to Ideal, SOI) Let $x = (D, E_1, E_2) \in L_{p\text{-isog}}$, and ω be a suborder representation such that $\text{SuborderVerification}(x, \omega) = 1$. Compute I, an ideal such that $\text{IdealVerification}(x, I) = 1$ or $\text{IdealVerification}((D, E_1, E_2^p), I) = 1$.

We will show in Proposition 7.1.3 the equivalence of Problem 7.1.1 with the problem of computing the endomorphism ring of the codomain from the suborder representation (Problem 7.1.2).

Problem 7.1.2. (SubOrder to Endormophism Ring (SOER)) Let $x \in L_{p\text{-isog}}$, and ω be a suborder representation such that $\text{SuborderVerification}(x, \omega) = 1$. Let $x = (D, E_1, E_2)$, compute $O_2 \subset B_{p,1}$ with $O_2 = \text{End}(E_2)$.

Proposition 7.1.3. (UPHA) The SOI and SOER problems are equivalent.

Proof. Since $O_R(I) = \text{End}(E_2)$ when $\text{VerifIdealProof}((D, E_1, E_2), I) = 1$, it is clear that solving SOIP implies solving SOERP in polynomial-time. The reverse direction is more complicated.

Assume that ω, O_2 is given with $\text{VerifSuborderProof}((D, E_1, E_2), \omega) = 1$ and $O_2 = \text{End}(E_2)$. We describe an algorithm finding an ideal representation I for $x \in L_{\text{isog}}$ (up to swapping E_1 and E_1^p, we can assume that it is true). Parse $\omega = \theta_1, \theta'_1, \ldots, \theta'_n$. The isogenies $\theta'_1, \ldots, \theta'_n$ have their degree in ℓ for some small ℓ, and so they can be translated into ideals using any efficient isogeny-to-ideal algorithm (for instance IdealToIsogenyFromKLPT). In that way, we obtain $O_{2,1}, \ldots, O_{2,n}$ principal ideals. Compute $\gamma_1, \ldots, \gamma_n = \text{GeneratingFamily}(O_2, D)$. Select $\alpha \in O_1$ such that D is inert in $\mathbb{Z}[\alpha]$ and $\gcd(n(\alpha), D) = 1$. Express D as a linear combination of $\prod_{j \in I} \gamma_j$ for $I \subset \{1, \ldots, n\}$ and compute α as the same linear combination of the $\prod_{j \in I} \gamma_j$. Compute $J = O_2\langle \alpha, D \rangle$. Find β such that $O_1 = O_R(J)^{\beta}$ and output $I = \overline{J}^{\beta}$.

The important property is that if I_0 is the O_1-ideal that we look for, then $\overline{I_0} = O_R(I_0)\langle \alpha, D \rangle$ when $\alpha \in O_1$ is such that D is inert in $\mathbb{Z}[\alpha]$ and $\gcd(n(\alpha), D) = 1$. This is a consequence of Lemma 6.2.5. The rest of the algorithm described above is just to compute the value of α through the isomorphism between

$O_R(I)$ and O_2 to get the O_2-ideal J. Finally, we send J back through the inverse isomorphism to compute $I = I_0$.

With the knowledge of O_2, the isogeny-to-ideal translation algorithms can be applied in polynomial time. The same is true for GeneratingFamily due to Proposition 3.4.4 and the fact that we perform all the other operations over the quaternions in polynomial time. □

Interestingly, we can show that the SOIP is also equivalent to another problem, the Torsion to Ideal Problem (TIP) that can be seen as a generalization of the CSSI problem introduced by De Feo and Jao for SIDH [JDF11] (see Problem 6.1.1) when the endomorphism ring of the domain curve is known. This assumption is not part of the CSSIP but generating a curve with unknown endomorphism ring is notoriously hard, and so the endomorphism ring of the domain curve can be assumed to be known unless there is a trusted setup. This assumption is central in most cryptanalysis papers against SIDH.

Problem 7.1.4. (T-Torsion to Ideal (T-TI)) Let $x = (D; E_1; E_2) \in L_{isog}$ where D is coprime to T and let $\varphi : E_1 \to E_2$ be an element of $isog_D$. Let $P; Q$ be a basis of $E_1[T]$. Given End(E_1) and $\varphi(P); \varphi(Q) \in E_2[T]$, Compute I, an ideal such that VerifIdealProof$(x; I) = 1$.

We prove Proposition 7.1.5 to highlight a link between our new problem and existing hard problems in order to help build con dence in our new assumption.

Proposition 7.1.5. For every $D; p$, there exists a value of T, such that the SOIP is equivalent to the T-TIP.

Proof. In this proof, we take a powersmooth value T big enough so that GeneratingFamily$_{D(T)}$ will terminate with overwhelming probability (we recall that $D(T)$ is the set of divisors of T). This assumption will be useful to prove that if we can solve the SOIP, we can solve the TIP for T. For the other direction, we need not assume anything on the size T, only that it is powersmooth. The fact that T is powersmooth implies that points of $E_1[T]$ and $E_2[T]$ have a polynomial representation, that discrete logarithms can be computed in $E_1[T]$ and that isogenies of degree T can be computed in polynomial time.

Let us assume that we know how to solve the TIP for such a powersmooth value T. Let $x; \quad$ be an instance of the SOIP and let us write φ for the isogeny corresponding to the representation . If we can compute the image of φ on $E_1[T]$, then we can apply our solver of the TIP to solve the SOIP. Using the suborder representation, we can apply SuborderEvaluation to compute the image of three cyclic subgroups of order T through φ (the ideals corresponding to the subgroups can be computed in polynomial time because we know End(E_1) and we can solve DLP e ciently over the T-torsion). Once we have the image of three subgroups, it easy to see that we can compute the image of any points with some DLP and pairing computations. Thus, we get a valid entry to the TIP and we use our solver to nd a solution.

For the other direction, let us assume that we have a solver for the SOIP. Let us take an instance of the TIP. By our assumption on the size of T, we can run GeneratingFamily$_{D(T)}$ to get a generating family $_1; \quad ; \ _n$ of $Z + D$ End(E_1) with norms dividing T. Since we know End(E_1) we can compute the embedding of $Z + D$ End(E_1) inside End(E_1). Since the embedding of $Z + D$ End(E_1) inside End(E_2) induced by φ is obtained by push-forward through φ of the embedding

$\mathbb{Z} + D\,\mathrm{End}(E_1) \hookrightarrow \mathrm{End}(E_1)$, it suffices to use the image of the basis P, Q given in input of the TIP to find the kernel of the endomorphism $\theta_1, \ldots, \theta_n \in \mathrm{End}(E_2)$ giving the generating family of the embedding $\mathbb{Z} + D\,\mathrm{End}(E_1) \hookrightarrow \mathrm{End}(E_2)$. Once, we have these kernels, we can derive the corresponding isogeny and this yields a valid suborder representation for θ. Now that we have this suborder representation, we can simply apply our solver of the SOIP to find a correct solution. □

All the results and algorithms from Section 4.3 were obtained with a prime degree D for simplicity. Yet, as we see from Proposition 2.3.16 and the explanations at the end of Section 4.3.2, nothing prevents us from having a suborder representation for composite degree isogenies. It is interesting to consider the case where D is not prime in the analysis of the SOIP because there are some cases where it is actually easy to solve. This happens, for instance, when D is powersmooth.

A polynomial time algorithm to solve the composite SOIP when D is powersmooth. The algorithm below is inspired by the inversion mechanism underlying the trapdoor one-way function that we introduced in Chapter 6. Let us fix an element $x = (D, E_1, E_2) \in L_{\mathrm{isog}}$ where each prime-power factor of D is in $O(\log(p))$. If we write θ for an isogeny of degree D between E_1, E_2, we are going to describe informally an algorithm to compute $\ker\hat{\theta}$ in $\mathrm{poly}(\log(p))$. Since $\mathrm{End}(E_1)$ is known and D is power-smooth, an ideal representation for $x \in L_{\mathrm{isog}}$ can be easily derived from $\ker\theta$ (or equivalently from $\ker\hat{\theta}$).

Let $D = \prod_{i=1}^{m} \ell_i^{e_i}$, it suffices to get $\ker\hat{\theta} \cap E_2[\ell_i^{e_i}]$ for each i to be able to reconstruct $\ker\hat{\theta}$. Let us fix an $i \in [1, m]$. The main idea introduced in the inversion mechanism in Section 6.2.3 is that if $\alpha = [d] + \theta \circ [\alpha_0] \circ \hat{\theta}$, then the equality $\ker(\alpha) \cap E_2[\ell_i^{e_i}] = \ker\hat{\theta} \cap E_2[\ell_i^{e_i}]$ depends only on ℓ and $\mathbb{Z}[\alpha_0]$. In particular, when ℓ_i is inert in $\mathbb{Z}[\alpha_0]$, then we have $\ker(\alpha) \cap E_2[\ell_i^{e_i}] = \ker\hat{\theta} \cap E_2[\ell_i^{e_i}]$. It is clear that such an α_0 always exists and that it can be computed in $O(\log(p) + \log(D))$. Once, the correct α_0 is found, D_{α_0} can be expressed as a linear combination of the generating family obtained from $1, \theta_1, \ldots, \theta_n$. With the coefficients of this linear combination, it suffices to evaluate $E_2[\ell_i^{e_i}]$ through the $\theta_1, \ldots, \theta_n$ and solve a few DLPs to obtain $\ker\hat{\theta} \cap E_2[\ell_i^{e_i}]$. This algorithm has to be repeated at most $O(\log(D))$ times to obtain the full description of $\ker\hat{\theta}$.

On the prime vs. composite case. Isogenies of degree $D_1 D_2$ can be decomposed as two isogenies of respective degrees D_1 and D_2. Thus, we know that the ideal that we look for can be decomposed as $I_1 \cdot I_2$ where $n(I_i) = D_i$. The local-global principle tells us that each coprime part behaves independently, and so there does not seem to be any reason why finding I_2 from the suborder representation for $D_1 D_2, E_1, E_2$ should be different from solving Problem 7.1.1 when the degree is simply D_2. Once we know I_2, it is easy to see that recovering I_1 reduces to an instance of Problem 7.1.1 of degree D_1. This informal reasoning justifies that taking D composite should only make Problem 7.1.1 easier to solve. The efficient algorithm that we described above in the case of powersmooth D leads to the same conclusion. Indeed, in this algorithm, we clearly recover each coprime part of the isogeny $\hat{\theta}_I$ independently.

The generic case: a heuristic quantum subexponential algorithm.
This paragraph presents informally the best-known algorithms to solve Problem 7.1.1. We will implicitly focus on the prime case, which appears to be the hardest case, as argued in the previous paragraph. We start by classical algorithms and worst-case complexity estimates before introducing a subexponential quantum algorithm, which is the best known generic method to solve Problem 7.1.1.

We start by analyzing the complexity of the brute-force algorithm. In full generality, for a given D, the brute force will take $O(\min(p, D))$. The idea is that since $\mathrm{End}(E_1)$ is part of the suborder representation, it suffices to enumerate through all $\mathrm{End}(E_1)$ ideals of norm D until IdealVerification succeeds. This is the reasoning we outlined in Section 5.3.2. There are $O(D)$ such ideals, but since there are only $O(p)$ curves, we expect to have to test at most $O(p)$ of them. Thus, the generic complexity of the brute force is $O(\min(D, p))$.

Another way to solve the problem generically is by computing $\mathrm{End}(E_2)$ (see Proposition 7.1.3). Without using the proof as a hint, the complexity is $\tilde{O}(p^{1/2})$ for classical computers and $\tilde{O}(p^{1/4})$ for quantum computers, as we explained already (see Section 5.3.2 for instance).

Now, let us look at the algorithm described above for powersmooth D in the generic case. Indeed, the algorithm remains correct and valid for any value of D. The only problem is that it becomes exponentially hard for a generic D. First, we need to be able to perform operations over the D-torsion. The smallest field of definition for the D-torsion can have degree in $\varphi(D)$ over $\mathbb{F}_p$. In that case, any operation over the D-torsion will have exponential complexity. Even assuming that the degree of definition is logarithmic in p, D, we still need to perform a D-isogeny computation from its kernel. When D is prime, the best known algorithm is the one we introduced in Section 4.1, and it has complexity $O(\sqrt{D})$. Thus, the complexity is exponential in the worst case.

We conclude by introducing a quantum algorithm with sub-exponential complexity in D. For that, we use the result from [KMPW21] that a one-way function $f : E \to F$ can be inverted at $f(e)$ by solving an instance of the hidden shift problem when there is a group action $? : G \times E \to E$ for which there exists a malleability oracle: an efficient way to evaluate the function $g \mapsto f(g ? e)$ on any $g \in G$. The hidden shift problem can be solved in quantum subexponential time. The authors from [KMPW21] proposed a key recovery attack on an imbalanced version of the SIDH scheme by using the group action of $(\mathrm{End}(E_1)/D\,\mathrm{End}(E_1))^\times$ on the set of cyclic subgroups of order D. This set is in correspondence with cyclic ideals of norm D inside $\mathrm{End}(E_1)$, and so we can invert the function $I \mapsto E/E[I]$ in subexponential time if we have a malleability oracle. In [KMPW21], the authors showed that this malleability oracle could be obtained as soon as the image of a big enough torsion-group was given through the secret isogeny. With our algorithm SuborderEvaluation, we presented a way to use the suborder representation to evaluate θ_I on any torsion subgroup. As a consequence, we can evaluate θ_I on any subgroup of powersmooth suborder, and this is more than enough to obtain a malleability oracle with the ideas of [KMPW21]. Thus, we can apply the reduction from [KMPW21] and get a sub-exponential quantum method to solve Problem 7.1.1.

Remark 7.1.6. The existence of a sub-exponential attack is inevitable as soon as one non-trivial endomorphism $\theta : E_2 \to E_2$ is revealed. The attack stems

from the existence of a group action of $\mathrm{Cl}(\mathbb{Z}[\ ])$ on the set of $\mathbb{Z}[\ ]$-orientations (see [Wes21] for instance). With the knowledge of , one can apply the idea (rst introduced by Biasse, Jao and Sankar [BJS14] in the special case where $\mathbb{Z}[\ ] = \mathbb{Z}[\sqrt{-p}]$) that the algorithm from Childs et al. [CJS14] can be adapted to nd a path of powersmooth degree between two $\mathbb{Z}[\ ]$-oriented curves. When this algorithm is applied between E_2 and E_1, a curve of known endomorphism ring, the path obtained in output allows the attacker to compute the endomorphism ring of E_2. This algorithm has sub-exponential complexity in $\log(\mathbb{Z}[\ ])$ as it reduces to an instance of the hidden shift problem.

Further analysis of the security problem. Even after seeing our analysis, the hardness of the SOERP may still come as a surprise to a reader familiar with isogeny-based cryptography. In particular, the fact that we reveal several endomorphisms of E_2 might seem like a very troublesome thing to do. This concern is legitimate: the algorithm from [EHL+20] to compute the endomorphism ring of any supersingular curve that we outlined in Section 2.2.1 is based on the principle that knowing two distinct non-trivial endomorphisms is enough to recover the full endomorphism ring in polynomial-time. However, this was only true under the conjecture that the order generated by two random non-trivial endomorphisms had a good probability of being Bass, which implied that they are contained in a few maximal orders. The endomorphisms that we reveal in the suborder representation are not random cycles. By design, the suborder they generate is not Bass and we know that it is contained in an exponential number of maximal orders (this number is equal to the number of D-isogenies by Lemma 2.3.15). As such, when using the endomorphisms of the suborder representation, the algorithm described in Section 2.2.1 is essentially the brute force attack where each ideal of norm D is tested.

Readers might also be concerned with the quaternion alternate path problem. A way to break the SOERP would be to use the embedding of $\mathbb{Z} + D\,\mathrm{End}(E_1)$ inside $\mathrm{End}(E_2)$ to compute a path from E_2 to a curve E_0 of a known endomorphism ring. Following the (now standard) blueprint that underlies most of the algorithms in this work, such an attack would be divided in two steps: rst a computation over the quaternions (analog to KLPT) and then a conversion through the Deuring Correspondence to obtain an isogeny connecting E_2 to E_0 (analog to IdealToIsogenyFromKLPT or IdealToIsogenyFromEichler). This supposed attack would have to work over orders of non-trivial Brandt invariant rather than maximal orders to exploit the suborder representation. It appears that the rst part of this method can be made to work over non-Gorenstein orders. In fact, the IdealSuborderEichlerNorm from Chapter 3 is exactly the analog of KLPT for orders of the form $\mathbb{Z} + D\mathcal{O}$. However, the fact that the Brandt-Invariant is non-trivial appears like a serious obstacle to the second part of the proposed attack. Indeed, as the number of curves admitting an embedding of $\mathbb{Z} + D\mathcal{O}$ inside their endomorphism ring is big, it becomes hard to tell which pairs of curves are connected by any ideal of the form $(\mathbb{Z} + D\mathcal{O}) \setminus J$ (which was not the case for maximal orders because we have almost a 1-to-1 correspondence between curves and maximal orders). Thus, it seems implausible that one may nd a path between E_2 and a given curve E_0 in that manner. Another way of seeing this is that since $\mathbb{Z} + D\mathcal{O}$ is a generic suborder shared by a lot of curves, we cannot compute anything that will be speci c to a given curve from

the knowledge of $\mathbb{Z} + D_O\mathcal{O}$ only.

7.2 A new NIKE based on a generalization of SIDH for large prime degrees.

We present here pSIDH (prime-SIDH) a new Non-interactive Key Exchange (NIKE) scheme. It is based on a SIDH-style isogeny diagram (see Section 6.1.1, Figure 6.1 and Figure 7.1) but with prime degrees.

Validation of public keys. The difference between a NIKE and a simple key exchange protocol is the validation of public keys. If we cannot verify that the public key sent by another participant is well-formed, we might be exposed to attacks by malicious adversaries who would create bad keys in order to recover our secret by observing the failures of the key exchange. For instance, this is what is done in [GPST16]. SIDH public key validation is an active topic of research [GPST16, FP22, UXT+22, DFDGZ21]. The active attacks by malicious adversaries can be prevented by providing a proof that the public key has been constructed correctly, or we can use the Fujisaki-Okamoto transform [FO99]. The former is rather costly in the case of SIDH [DFDGZ21] and the latter has several downsides (e.g. we cannot build a NIKE). A NIKE is a key exchange that has a validation mechanism to verify correctness of the public keys. The Elliptic Curve Diffie-Hellman protocol is an example of a very simple NIKE (but it is not post-quantum). The absence of such feature is one of the downsides of SIDH in comparison to the CSIDH protocol [CLM+18]. In fact, CSIDH is one of the few examples of practical post-quantum secure NIKE. Our protocol pSIDH is an alternate solution based on different isogeny-based assumptions than CSIDH.

The pSIDH protocol. The principle of our key exchange is quite simple, as it is an adaptation of SIDH for prime degrees. For secret keys, we propose to use ideal representations and then take suborder representations as public keys. The key exchange will be made possible with the `SuborderEvaluation` algorithm from Section 4.3.4. In terms of security, the pSIDH key recovery problem is exactly the SOIP and the NIKE is secure under the hardness of a decisional variant of the SOIP similarly to SIDH with the CSSI and SSDDH problems introduced in [JDF11]. We stress that we leave efficiency considerations to future work and merely show that the scheme can be executed in polynomial-time.

To adapt SIDH to the setting of two prime degrees D_A, D_B, we need a new method to compute the codomain of the push-forward isogenies (since the Vélu formulas and the algorithms from Section 4.1 are not practical for prime degrees). If the ideal representations are secret keys and the suborder representations are public keys, the computation of the common key $j(E)$ can be done as follows. Given an ideal I of norm D_A and the suborder $\mathbb{Z} + D_B\mathcal{O}$, it is possible to find an element $\alpha \in (\mathbb{Z} + D_B\mathcal{O}_0) \cap I$ of norm $D_A S$ where S is a powersmooth integer with the algorithm `IdealSuborderEichlerNorm` (Algorithm 14). The embedding $\iota_B : \mathbb{Z} + D_B\mathcal{O}_0 \hookrightarrow \mathrm{End}(E_B)$, is obtained by pushing forward the embedding of $\mathbb{Z} + D_B\mathcal{O}_0$ inside $\mathrm{End}(E_0)$ through φ_B and so we have $\iota_B(\alpha) = \varphi_A \circ [\alpha] \circ \widehat{\varphi_A}$ where φ_A has degree S. Thus, using ι_B, the suborder

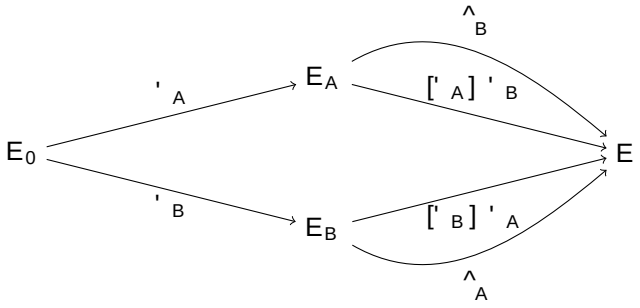


Figure 7.1: pSIDH-isogeny diagram.

representation of φ_B, we can use SuborderEvaluation to compute $\ker \hat{\varphi}_A$ and $\hat{\varphi}_A$. The codomain of $\hat{\varphi}_A$ is isomorphic to E, and so the common secret $j(E)$ can be derived from that.

These ideas are summarized in Figure 7.1 and the full description of the key exchange mechanism is given as pSIDHKeyExchange. The key generation algorithm pSIDHKeyGen is also described as Algorithm 38. To be able to run this algorithm in polynomial-time, we need to be able to compute efficiently isogenies of degree m_A and to be able to manipulate the full $\deg m_A$ torsion. This is why we take the degree of m_A as a divisor of a powersmooth integer T. To be able to apply SuborderEvaluation, we also need that T is coprime to the degree of the endomorphisms of the suborder representation (so we take T coprime to ℓ).

The public parameters should include a prime p and a starting curve E_0, together with a description of $\mathrm{End}(E_0)$.

Algorithm 38 pSIDHKeyGen(D)

Input: A prime number $D \approx p$.

Output: The pSIDH public key $pk = E$, and the pSIDH secret key $sk = I$ where $\mathcal{O}$ is a suborder representation and I an ideal representation for $(D; E_0; E) \in L_{p\text{-isog}}$.

1: Sample I as a random $\mathcal{O}_0$-ideal of norm D.
2: Compute $\mathcal{O} = \mathrm{IdealToSuborder}(I)$ and set E as the domain of the endomorphisms in $\mathcal{O}$.
3: return $pk; sk = (E; \mathcal{O}); I$.

Proposition 7.2.1. (UPHA) pSIDHKeyExchange terminates in expected $O(\mathrm{poly}(\log(pDD')))$.

Proof. Since $B = O(\mathrm{poly}(\log(pDD')))$, T can be chosen with a smoothness bound equal in $O(\mathrm{poly}(\log(pDD')))$. Thus, the final computation of $\mathcal{O}$ can be done in $O(\mathrm{poly}(\log(pD'D)))$. The remaining computations terminate in expected $O(\mathrm{poly}(\log(pD'D)))$ due to Propositions 3.4.4 and 3.4.6 and Propositions 2.2.4, 4.3.6 and 4.3.8. □

Algorithm 39 pSIDHKeyExchange(I, D', E', α)

Input: I an ideal of degree D and a prime $D' \nmid D, p$. A curve E' and a suborder representation α.
Output: A j-invariant or $\perp$.

1: Parse $\alpha = (O, \iota'_1, \ldots, \iota'_n)$.
2: Compute $\iota_1, \ldots, \iota_n = \text{GeneratingFamily}(O_0, D')$.
3: **if** $\neg\text{SuborderVerification}_M((D', E_0, E'), \alpha)$ or $O \neq O_0$ **then**
4: Return $\perp$.
5: **end if**
6: Take a powersmooth integer T coprime to ℓ.
7: Compute $L = \text{ConnectingIdeal}(O_0, O)$
and $J = \text{RandomEquivalentPrimeIdeal}(L)$ with $J \perp L$.
8: Compute $\beta = \text{IdealSuborderEichlerNorm}_{M(T)}(D', J, I)$.
9: Factorize $T = \prod_{i=1}^{m} \ell_i^{e_i}$.
10: Set $G = \langle 0_{E_0} \rangle$.
11: **for** $i \in [1, m]$ **do**
12: Compute $J_i = O_0\langle \overline{\beta}, \ell_i^{e_i} \rangle$.
13: $G = G + \text{SuborderEvaluation}(\alpha, D', J_i)$.
14: **end for**
15: Compute $\varphi : E' \to E'/G$.
16: **return** $j(E'/G)$.

Proposition 7.2.2. Let $D_A, D_B \nmid p$ be two distinct prime numbers. If $E_A, \alpha_A, I_A = \text{pSIDHKeyGen}(D_A)$ and $E_B, \alpha_B, I_B = \text{pSIDHKeyGen}(D_B)$, then

$$\text{pSIDHKeyExchange}(I_A, D_B, E_B, \alpha_B) = \text{pSIDHKeyExchange}(I_B, D_A, E_A, \alpha_A).$$

Proof. Let us write φ'_A, φ'_B the isogenies corresponding to the two ideals I_A, I_B. Then, the quaternion element $\overline{\alpha_A^{-1}\beta\alpha_A}$ obtained at Step 8 during the execution of $\text{pSIDHKeyExchange}(I_A, D_B, E_B, \alpha_B)$ corresponds to the endomorphism $\theta_{0;A} = \varphi'_A \circ (\mathbb{Z} + D_B\,\text{End}(E_0)) \cap I_A \to \text{End}(E_B)$. Since it is contained in $(\mathbb{Z} + D_B\,\text{End}(E_0)) \cap I_A$, this endomorphism is equal to $\alpha \circ [\varphi'_B] \circ \varphi'_A$ where $\hat{\varphi}_A = [\varphi'_B] \circ \hat{\varphi}_{A;0}$ for some isogeny $\varphi_{A;0} : E_0 \to E_A$. In particular, the codomain of $\hat{\varphi}_A$ is isomorphic to the codomain of $[\varphi'_B] \circ \varphi'_A$. We can make the same reasoning by swapping A and B and by definition of push-forward isogenies and Proposition 4.3.8, the two j-invariants obtained at the end of the two executions of pSIDHKeyExchange are equal. □

Security. By design, we have the algorithm SuborderVerification to validate public keys, and so we obtain a NIKE. For key validation, the public parameters for pSIDH also include a value $M = p^k - 1$ as in Proposition 4.3.5. By design, the pSIDH key recovery problem is simply the SOIP (Problem 7.1.1). To prove security of our key exchange, we need a decisional variant, which we call the pSSDDH (prime supersingular DDH) problem (see Problem 7.2.3).

Problem 7.2.3. (pSSDDH) Let $D_A, D_B \nmid p$ be two distinct prime numbers and $E_A, \alpha_A, I_A = \text{pSIDHKeyGen}(D_A)$ and $E_B, \alpha_B, I_B = \text{pSIDHKeyGen}(D_B)$. The problem is to distinguish between the two distributions:

1. $(E_A; \ _A); (E_B; \ _B); E_{AB}$ where $\mathrm{End}(E_{AB}) = \mathcal{O}_R(I_A \setminus I_B)$.
2. $(E_A; \ _A); (E_B; \ _B); E_C$ where E_C is a random curve $N_A N_B$-isogenous to E_0.

With the pSSDDH problem, we can state the security of the key agreement protocol we just outlined. The proof mimics the one made in [JDF11].

Proposition 7.2.4. Under the pSSDDH assumption, the key-agreement protocol made of pSIDHKeyGen and pSIDHKeyExchange is session-key secure in the authenticated-links adversarial model of Canetti and Krawczyk [CK01].

7.2.1 About efficiency

We have proven (at least heuristically) that all our new algorithms can be executed in polynomial time. However, this does not prove anything on the concrete efficiency. For instance, it would be interesting to compare pSIDH with other existing isogeny-based key exchanges. The only thing that we can claim with certainty is that pSIDH will be a lot slower than SIDH. In fact, we will rather estimate the complexity of pSIDH by comparing it to that of SQISign. This comparison is relevant for two reasons: we can take the same size of prime p (and measure relative efficiency by counting the number of operations over $\mathbb{F}_{p^2}$) and the bottlenecks should be the same. We elaborate on that below.

Our analysis in Section 7.1 indicates that the only security constraint on the prime p is that it needs to be large enough to prevent the exponential attacks against the endomorphism ring problem. Once p has been fixed, the hardness of our new SOIP depends on the value of D. The main attack against the SOIP that we introduce in Section 7.1 has quantum sub-exponential complexity in D. So we can expect the value of D to be significantly larger than p. This gap between p and D will also induce a gap between the performances of SQISign and the performances of pSIDH. Based on empirical observations, we can predict that the bottleneck in our algorithms is going to be the same as the bottleneck in SQISign's signature: executions of the idealToIsogenyFromEichler sub-algorithm. Our method requires performing a number of arithmetic operations over $\mathbb{F}_{p^2}$ that is linear in the length of the isogeny to be translated. For SQISign the length is equal to $O(\log(p)) = O( )$ where is the security parameter. For pSIDH, the size estimates from Chapter 3 show that the length is in $O(\log(pD))$. Thus, we can expect pSIDH to be slower than the signature computation in SQISign (a more concrete analysis is required to obtain a more precise estimate of the relative efficiencies). Since SQISign is orders of magnitude slower than SIDH we conclude that it will be the same for pSIDH.

7.3 Potential for other cryptographic applications

We have introduced a new NIKE scheme, pSIDH, as a way to illustrate the possibilities offered by our new isogeny representation. When making the comparison with SIDH, the two main advantages of our construction are the different security assumption and the non-interactive key validation mechanism. These two properties probably do not make up for the huge efficiency gap between SIDH

and pSIDH (see Section 7.2.1) but they could be important for more complicated primitives. As such, pSIDH should only be considered as a first example of what can be done with our suborder representation. We discuss below other potential applications. We propose directions to explore for future work rather than concrete protocols.

Adaptation of protocols based on SIDH. A lot of isogeny-based primitives are based on the mechanism underlying the SIDH key exchange. We can mention n-party key exchange [AJJS19], signatures [YAJ+17] built upon the SIDH identification scheme from [JDF11], oblivious transfers [BOBN19, dSGOPS20] and oblivious PRF [BKW20]. It is natural to ask if we can adapt them to the setting of pSIDH.

A multi-party key exchange can easily be designed in the SIDH setting. It suffices to take coprime degrees $D_1, D_2, \dots, D_n$, and the commutative diamond in Figure 7.1 can be extended to an n-dimensional commutative diagram that leads naturally to a multi-party key exchange. The main problem with this protocol in the setting of SIDH is security, as it is under serious threat of the most recent advances on torsion point attacks from [KMP+20] (the construction is broken as soon as $n \geq 6$). It seems plausible to adapt this multi-party key exchange to the setting of pSIDH using the successive suborders $\mathbb{Z}[\sqrt{-} + D_i D_j \mathcal{O}$, $\mathbb{Z}[\sqrt{-} + D_i D_j D_k \mathcal{O}, \dots$. In terms of security, this n-party pSIDH could be addressing some shortcomings of the SIDH version. Indeed, as explained in Section 7.1, the composite version of the SOIP (Problem 7.1.1) appears to be reducing to the prime case, which tends to suggest that the multi-party key exchange could be as secure as the two-party version. Remains to see how exactly the successive suborders can be computed from the suborder representations. We leave that to future work.

The flexibility offered by pSIDH could also be useful in a simpler setting. Let us assume that there are three parties, Alice, Bob and Charlie, who want to agree on three keys (one for each pair of parties). In the setting of SIDH, they will need at least 4 public keys. Indeed, if Alice has a key of degree D_A and Bob a key of degree D_B, then Charlie will need a key of degree D_B (resp. D_A) to interact with Alice (resp. Bob). In pSIDH, each party can select a different degree, and so they need only 3 public keys.

Contrary to the multi-party key exchange, the adaptation of SIDH signatures to the setting of pSIDH seems like a complicated task. It would require a zero-knowledge ideal-representation proof of knowledge, which seems hard to build. However, if it is possible to build one, the suborder representation appear like a good starting point, so there could be more to that story.

The OT protocols that we mentioned should not be complicated to adapt to pSIDH given that they mostly require a DH-like commutative diagram. However, it is not clear that using pSIDH would be more interesting than SIDH for that primitive.

The oblivious PRF from [BKW20] appears like a more interesting application. First, verifiability is a big issue for this primitive and the construction proposed in [BKW20] includes some zero-knowledge isogeny proof-of-knowledge, which are quite expensive and not very compact in the setting of SIDH. Given that verifying computations is inherently a lot easier with pSIDH, it might prove a good match. Second, [BKM+21] have presented some attacks against

the SIDH-based OPRF from [BKW20]. These attacks might be avoided with a pSIDH variant. Of course, as for the n-party key exchange, new algorithmic tools are needed before we can hope to obtain the analog of the OPRF in the setting of pSIDH, and it requires some more work.

Group action. The sub-exponential quantum attack that we presented in Section 7.1 was based on the existence of a group action on the set of ideals of norm D. After a quick glance, it seems like this group action could also be cryptographically relevant and be used to instantiate the increasing list of group-action based protocols in the literature. It is not exactly clear that this new group action could be more interesting than the one based on CSIDH [CLM+18], but it is probably worth studying further to better understand the differences between the two. One hope is that the structure of the underlying group could be easier to compute. The signature scheme CSI-FiSh [BKV19] is a good example of the interest of having computed this structure, but their solution, based on the CSIDH group action, does not scale well asymptotically. If we could do the same with our new group action but without the scaling issue, it could be interesting.

Zero-knowledge proof of suborder representation knowledge. We mentioned several times already the interest of zero-knowledge proofs of isogeny knowledge. We know there exist somewhat practical instantiations in the setting of SIDH and CSIDH. We explained and argued that it seems complicated to do the same with ideal representations. The next natural question is whether we can hope to do it for the new suborder representations. Proving the knowledge of several endomorphisms of a given norm might be feasible, but making the additional verification that they generate a specific quaternion order might prove a lot more arduous. Alas, there does not seem to be an easy way to do that.

Trapdoor mechanism from endomorphisms revelation. One of the main novelties behind our suborder representation construction is the revelation of suborders of rank 4 contained inside endomorphism rings of supersingular curves. Until our work, revealing more than one non-trivial endomorphisms has always been considered as dangerous, but we conjecture with the hardness of the SOIP that it is not problematic when done carefully. It might be possible to exploit this mechanism for further applications. For instance, we can look at the trapdoor one-way function (TOWF) of the SETA scheme from [DFFdSG+21]. In this primitive, the trapdoor is some endomorphism of the public key curve. In the instantiation proposed in [DFFdSG+21], the endomorphism ring of the public key curve is typically computed during key generation, but we could imagine a situation where one participant P_1 generates a curve E (and computes its endomorphism ring along the way) before revealing a well-chosen endomorphism of E to another participant P_2. Then, P_2 could use this endomorphism to perform some protocols (for instance the SETA-TOWF) without knowing anything else on the curve E.

It seems tempting to try to build IBE from this setting. For instance, the master public key could be a curve E with the master secret key as $\mathrm{End}(E)$, identities would be isogenies from E to curves E_{id} and the corresponding secret

key would be an endomorphism of E_{id} that could be used as a SETA secret key. Unfortunately, it seems hard to choose these secret keys in a way that would prevent an adversary who has access to several of them to recover enough information to generate secret keys for himself. Even though IBE appears to be out of reach from this idea, lesser primitives could still be achievable.

Bibliography

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In International Conference on the Theory and Applications of Cryptographic Techniques, pages 418–433. Springer, 2002.

[ABLS07] Noga Alon, Itai Benjamini, Eyal Lubetzky, and Sasha Sodin. Non-backtracking random walks mix faster. Communications in Contemporary Mathematics, 9(04):585–603, 2007.

[ACC+20] Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Aaron Hutchinson, Amir Jalali, David Jao, Koray Karabina, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Geovandro Pereira, Joost Renes, Vladimir Soukharev, and David Urbanik. Supersingular isogeny key encapsulation, 2020.

[ACL+22] Sarah Arpin, Mingjie Chen, Kristin E Lauter, Renate Scheidler, Katherine E Stange, and Ha TN Tran. Orienteering with one endomorphism. arXiv preprint arXiv:2201.11079, 2022.

[ACVCD+19] Gora Adj, Daniel Cervantes-Vazquez, Jesus-Javier Chi-Domnguez, Alfred Menezes, and Francisco Rodrguez-Henrquez. On the cost of computing isogenies between supersingular elliptic curves. In Carlos Cid and Michael J. Jacobson Jr., editors, Selected Areas in Cryptography – SAC 2018, pages 322–343, Cham, 2019. Springer International Publishing.

[AFMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In International Conference on the Theory and Application of Cryptology and Information Security, pages 411–439. Springer, 2020.

[AIL+21] Laia Amoros, Annamaria Iezzi, Kristin Lauter, Chloe Martindale, and Jana Sotakova. Explicit connections between supersingular isogeny graphs and bruhat–tits trees. Cryptology ePrint Archive, 2021.

[AJJS19] Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. Practical supersingular isogeny group key agreement. IACR Cryptol. ePrint Arch., 2019:330, 2019.

[AJK+16] Reza Azarderakhsh, David Jao, Kassem Kalach, Brian Koziel, and Christopher Leonardi. Key compression for isogeny-based cryptosystems. In Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography, pages 1{10. ACM, 2016.

[Ank52] Nesmith Cornett Ankeny. The least quadratic non residue. Annals of mathematics, pages 65{72, 1952.

[Arp22] Sarah Arpin. Adding level structure to supersingular elliptic curve isogeny graphs. arXiv preprint arXiv:2203.03531, 2022.

[BCL08] Reinier Bröker, Denis Charles, and Kristin Lauter. Evaluating large degree isogenies and applications to pairing based cryptography. In International Conference on Pairing-Based Cryptography, pages 100{112. Springer, 2008.

[BCP97] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. J. Symbolic Comput., 24(3-4):235{265, 1997. Computational algebra and number theory (London, 1993). https://www.math.ru.nl/ ~bosma/pubs/JSC1997Magma.pdf

[BCS97] Peter Bürgisser, Michael Clausen, and Mohammad Amin Shokrollahi. Algebraic complexity theory, volume 315 of Grundlehren der mathematischen Wissenschaften. Springer, 1997.

[BDFLS20] Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. ANTS XIV, 2020.

[Bel08] Juliana V Belding. Number theoretic algorithms for elliptic curves. University of Maryland, College Park, 2008.

[BJS14] Jean-Francois Biasse, David Jao, and Anirudh Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In International Conference on Cryptology in India, pages 428{442. Springer, 2014.

[BKM+21] Andrea Basso, Peter Kutas, Simon-Philipp Merz, Christophe Petit, and Antonio Sanso. Cryptanalysis of an oblivious prf from supersingular isogenies. Cryptology ePrint Archive, 2021.

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. Csi- sh: E cient isogeny based signatures through class group computations. In International Conference on the Theory and Application of Cryptology and Information Security, pages 227{247. Springer, 2019.

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In International Conference on the Theory and Application of Cryptology and Information Security, pages 520{550. Springer, 2020.

[BOBN19] Paulo Barreto, Glaucio Oliveira, Waldyr Benits, and Anderson Nascimento. Supersingular isogeny oblivious transfer. In Anais do XIX Simposio Brasileiro de Seguranca da Informac-ao e de Sistemas Computacionais, pages 99{112. SBC, 2019.

[Bos20] Alin Bostan. Computing the N-th term of a q-holonomic sequence. Preprint, 2020. https://specfun.inria.fr/bostan/NthQhol-s.pdf .

[Boy08] Xavier Boyen. The uber-assumption family. In Steven D. Galbraith and Kenneth G. Paterson, editors, Pairing-Based Cryptography - Pairing 2008, Second International Conference, Egham, UK, September 1-3, 2008. Proceedings, volume 5209 of Lecture Notes in Computer Science, pages 39{56. Springer, 2008.

[Brz83] Juliusz Brzezinski. On orders in quaternion algebras. Communications in algebra, 11(5):501{522, 1983.

[BS07] William D Banks and Igor E Shparlinski. Integers with a large smooth divisor. Integers, 7:A17, 2007.

[BS20] Xavier Bonnetain and Andre Schrottenloher. Quantum security analysis of CSIDH. In Advances in Cryptology - EUROCRYPT 2020, pages 493{522, 2020.

[Cas91] J. W. S. Cassels. Lectures on Elliptic Curves, volume 24 of London Mathematical Society Student Texts. Cambridge University Press, 1991.

[Cer04] Juan Marcos Cervino. On the correspondence between supersingular elliptic curves and maximal quaternionic orders. arXiv preprint math/0404538, 2004.

[CFMR+19] Antoine Casanova, Jean-Charles Faugere, Gilles Macario-Rat, Jacques Patarin, Ludovic Perret, and Jocelyn Ryckeghem. GeMSS: a great multivariate short signature. NIST Post-Quantum Cryptography Standardization, 2019.

[CH17] Craig Costello and Huseyin Hisil. A simple and compact algorithm for SIDH with arbitrary degree isogenies. In ASIACRYPT (2), volume 10625 of Lecture Notes in Computer Science, pages 303{329, 2017. https://eprint.iacr.org/2017/504 .

[CJL+17] Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. Efficient Compression of SIDH Public Keys, pages 679{706. Springer International Publishing, 2017.

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. Journal of Mathematical Cryptology, 8(1):1{29, 2014.

[CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In International conference on the theory and applications of cryptographic techniques, pages 453{474. Springer, 2001.

[CK19] Leonardo Cob and David Kohel. Orienting supersingular isogeny graphs. Number-Theoretic Methods in Cryptology 2019, 2019.

[CLG09] Denis X. Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. Journal of Cryptology, 22(1):93{113, Jan 2009.

[CLM+18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In International Conference on the Theory and Application of Cryptology and Information Security, pages 395{427. Springer, 2018.

[CMN21] Craig Costello, Michael Meyer, and Michael Naehrig. Sieving for twin smooth integers with solutions to the prouhet-tarry-escott problem. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 272{301. Springer, 2021.

[Cor08] Giuseppe Cornacchia. Su di un metodo per la risoluzione in numeri interi dell'equazione $\sum_{h=0}^{n} c_h x^{n-h} y^h = p$. Giornale di Matematiche di Battaglini, 46:33{90, 1908.

[Cos19] Craig Costello. B-SIDH: supersingular isogeny Die-Hellman using twisted torsion, 2019. https://ia.cr/2019/1145 .

[Cou06] Jean Marc Couveignes. Hard homogeneous spaces. IACR Cryptology ePrint Archive, 2006:291, 2006.

[Cox11] David A Cox. Primes of the form x2+ ny2: Fermat, class field theory, and complex multiplication, volume 34. John Wiley & Sons, 2011.

[CPV20] Wouter Castryck, Lorenz Panny, and Frederik Vercauteren. Rational isogenies from irrational endomorphisms. Advances in Cryptology{EUROCRYPT 2020, 12106:523, 2020.

[CS18] Craig Costello and Benjamin Smith. Montgomery curves and their arithmetic. Journal of Cryptographic Engineering, 8(3):227{240, 2018.

[CSCDJRH20] Jorge Chavez-Saab, Jesus-Javier Chi-Dominguez, Samuel Jaques, and Francisco Rodriguez-Henriquez. The SQALE of CSIDH: square-root velu quantum-resistant isogeny action with

low exponents. Technical report, Cryptology ePrint Archive, Report 2020/1520, 2020. https://eprint. iacr. org . . . , 2020.

[CSV21] Sara Chari, Daniel Smertnig, and John Voight. On basic and bass quaternion orders. Proceedings of the American Mathematical Society, Series B, 8(2):11{26, 2021.

[Dam10] Damgard. On protocols. http://www.cs.au.dk/%7eivan/Sigma.pdf , 2010.

[Deu41] Max Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkorper. Abhandlungen aus dem Mathematischen Seminar der Universitat Hamburg, 14(1):197{272, Dec 1941.

[DFDGZ21] Luca De Feo, Samuel Dobson, Steven D Galbraith, and Lukas Zobernig. Sidh proof of knowledge. Cryptology ePrint Archive, 2021.

[DFFdSG+21] Luca De Feo, Tako Boris Fouotsa, Cyprien Delpech de Saint Guilhem, Peter Kutas, Antonin Leroux, Christophe Petit, Javier Silva, and Benjamin Wesolowski. Seta: Supersingular encryption from torsion attacks. In ASIACRYPT, 2021.

[DFG19] Luca De Feo and Steven D Galbraith. Seasign: Compact isogeny signatures from class group actions. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 759{789. Springer, 2019.

[DFJP14] Luca De Feo, David Jao, and Jerôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. Journal of Mathematical Cryptology, 8(3):209{247, 2014.

[DFKL+20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. Sqisign: compact post-quantum signatures from quaternions and isogenies. In International Conference on the Theory and Application of Cryptology and Information Security, pages 64{93. Springer, 2020.

[DFLW22] Luca De Feo, Antonin Leroux, and Benjamin Wesolowski. New algorithms for the deuring correspondence: Sqisign twice as fast. Cryptology ePrint Archive, 2022.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaner. Security of the at-shamir transformation in the quantum random-oracle model. In Annual International Cryptology Conference, pages 356{383. Springer, 2019.

[DG16] Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over F_p. Designs, Codes and Cryptography, 78(2):425{440, February 2016.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. IEEE transactions on Information Theory, 22(6):644{654, 1976.

[DKL18] Jean-Marie De Koninck and Patrick Letendre. New upper bounds for the number of divisors function. arXiv preprint arXiv:1812.09950, 2018.

[Dor87] David R Dorman. Global orders in definite quaternion algebras as endomorphism rings for reduced cm elliptic curves. Theorie des nombres (Quebec, PQ, 1987), pages 108–116, 1987.

[dSGOPS20] Cyprien Delpech de Saint Guilhem, Emmanuela Orsini, Christophe Petit, and Nigel P Smart. Semi-commutative masking: A framework for isogeny-based protocols, with an application to fully secure two-round isogeny-based ot. In International Conference on Cryptology and Network Security, pages 235–258. Springer, 2020.

[EB92] M Eichler and J Brzezinski. On the imbeddings of imaginary quadratic orders in definite quaternion orders. Journal für die reine und angewandte Mathematik, 426:91–106, 1992.

[EHL+18] Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In Jesper Buus Nielsen and Vincent Rijmen, editors, Advances in Cryptology – EUROCRYPT 2018, pages 329–368, Cham, 2018. Springer International Publishing.

[EHL+20] Kirsten Eisenträger, Sean Hallgren, Chris Leonardi, Travis Morrison, and Jennifer Park. Computing endomorphism rings of supersingular elliptic curves and connections to path-finding in isogeny graphs. Open Book Series, 4(1):215–232, 2020.

[Eic36] Martin Eichler. Untersuchungen in der zahlentheorie der rationalen quaternionenalgebren. Journal für die reine und angewandte Mathematik, 174:129–159, 1936.

[Eic38] Martin Eichler. Über die Idealklassenzahl total definiter Quaternionenalgebren. Mathematische Zeitschrift, 43(1):102–109, 1938.

[FHK+19] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-Fourier lattice-based compact signatures over NTRU. NIST Post-Quantum Cryptography Standardization, 2019.

[FKM21] Tako Boris Fouotsa, Peter Kutas, and Simon-Philipp Merz. On the isogeny problem with torsion point information. IACR Cryptol. ePrint Arch., 2021:153, 2021.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Annual international cryptology conference, pages 537–554. Springer, 1999.

[FP22] Tako Boris Fouotsa and Christophe Petit. A new adaptive attack on sidh. In Cryptographers' Track at the RSA Conference, pages 322{344. Springer, 2022.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identi cation and signature problems. In Conference on the theory and application of cryptographic techniques, pages 186{194. Springer, 1986.

[GHS02] Steven D Galbraith, Florian Hess, and Nigel P Smart. Extending the ghs weil descent attack. InInternational Conference on the Theory and Applications of Cryptographic Techniques, pages 29{44. Springer, 2002.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems.Journal of the ACM (JACM), 38(3):690{728, 1991.

[GPS17] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identi- cation protocols and signature schemes based on supersingular isogeny problems. InASIACRYPT, 2017.

[GPST16] Steven D Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In International Conference on the Theory and Application of Cryptology and Information Security, pages 63{91. Springer, 2016.

[IK21] Henryk Iwaniec and Emmanuel Kowalski. Analytic number theory, volume 53. American Mathematical Soc., 2021.

[JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, International Workshop on Post-Quantum Cryptography { PQCrypto 2011, pages 19{34, 2011.

[JDF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, Post-Quantum Cryptography, pages 19{34, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[JMV09] D. Jao, S. D. Miller, and R. Venkatesan. Expander graphs based on GRH with an application to elliptic curve cryptography. Journal of Number Theory, 129(6):1491{1504, 2009.

[Kan89] Masanobu Kaneko. Supersingular j-invariants as singular moduli mod p. Osaka Journal of Mathematics, 26(4):849{855, 1989.

[Kat76] N Katz. An overview of deligne's proof of the riemann hypothesis for varieties over nite elds. Mathematical developments arising from Hilbert problems (Proc. Sympos. Pure Math., XXVIII, Northern Illinois Univ., De Kalb, Ill., 1974), pages 275{305, 1976.

[Kat10] Jonathan Katz. *D*igital signatures. Springer Science & Business Media, 2010.

[KLPT14] David Kohel, Kristin E. Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion `-isogeny path problem. *I* ACR Cryptology ePrint Archive, 2014:505, 2014.

[KMP+20] Peter Kutas, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E Stange. Weak instances of SIDH variants under improved torsion-point attacks. *a*rXiv preprint arXiv:2005.14681, 2020.

[KMPW21] Peter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkamper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. In *A*nnual International Conference on the Theory and Applications of Cryptographic Techniques, pages 242{271. Springer, 2021.

[Koh96] David Kohel. *E*ndomorphism rings of elliptic curves over nite elds. PhD thesis, University of California, Berkeley, 1996.

[Lan87] Serge Lang. Elliptic functions. In *E*lliptic functions, pages 5{21. Springer, 1987.

[LB20] Jonathan Love and Dan Boneh. Supersingular curves with small noninteger endomorphisms. *O*pen Book Series, 4(1):7{22, 2020.

[Ler21] Antonin Leroux. A new isogeny representation and applications to cryptography. *C*ryptology ePrint Archive, 2021.

[Ler22] Antonin Leroux. An e ective lower bound on the number of orientable supersingular elliptic curves. *C*ryptology ePrint Archive, 2022.

[Lit28] John E Littlewood. On the class-number of the corpus p ($\sqrt{k}$). *P*roceedings of the London Mathematical Society, 2(1):358{372, 1928.

[LPS86] Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Hecke operators and distributing points on the sphere i. *C*ommunications on Pure and Applied Mathematics, 39(S1):S149{S186, 1986.

[LV15] Kristin Lauter and Bianca Viray. On singular moduli for arbitrary discriminants. *I*nternational Mathematics Research Notices, 2015(19):9206{9250, 2015.

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum atshamir. In *A*nnual International Cryptology Conference, pages 326{355. Springer, 2019.

[Mon87] Peter L. Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *M*athematics of Computation, 48(177):243{264, 1987.

[MP19] Chloe Martindale and Lorenz Panny. How to not break SIDH. Cryptology ePrint Archive, Report 2019/558, 2019. https://eprint.iacr.org/2019/558.

[MR18] Michael Meyer and Ste en Reith. A faster way to the CSIDH. In *INDOCRYPT*, volume 11356 of *L*ecture Notes in Computer Science, pages 137{152. Springer, 2018. https://ia.cr/2018/782.

[MS16] Dustin Moody and Daniel Shumow. Analogues of Velu's formulas for isogenies on alternate models of elliptic curves. *M*athematics of Computation, 85(300):1929{1951, 2016. https://ia.cr/2011/430.

[Mum66] David B. Mumford. On the equations de ning abelian varieties. I. *I*nventiones Mathematicae, 1(4):287{354, 1966. https://dash.harvard.edu/handle/1/3597241.

[NR19] Michael Naehrig and Joost Renes. Dual isogenies and their application to public-key compression for isogeny-based cryptography. In Steven D. Galbraith and Shiho Moriai, editors, *A*dvances in Cryptology { ASIACRYPT 2019, pages 243{272, Cham, 2019. Springer International Publishing.

[Onu21] Hiroshi Onuki. On oriented supersingular elliptic curves. *F*inite Fields and Their Applications, 69:101777, 2021.

[PDJ20] Geovandro C. C. F. Pereira, Javad Doliskani, and David Jao. x-only point addition formula and faster torsion basis generation in compressed sike. Cryptology ePrint Archive, Report 2020/431, 2020.

[Pei20] Chris Peikert. He gives C-sieves on the CSIDH. In *A*dvances in Cryptology - EUROCRYPT 2020, pages 463{492, 2020.

[Pet17] Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In *I*nternational Conference on the Theory and Application of Cryptology and Information Security, pages 330{353. Springer, 2017.

[Piz80] Arnold Pizer. An algorithm for computing modular forms on $_0(n)$. *J*ournal of Algebra - J ALGEBRA, 64:340{390, 06 1980.

[Piz90] Arnold K Pizer. Ramanujan graphs and Hecke operators. *B*ulletin of the American Mathematical Society, 23(1):127{137, 1990.

[Pol74] John M. Pollard. Theorems on factorization and primality testing. *M*athematical Proceedings of the Cambridge Philosophical Society, 76(3):521{528, 1974. https://doi.org/10.1017/S0305004100049252.

[PS18] Christophe Petit and Spike Smith. An improvement to the quaternion analogue of the l-isogeny path problem, 2018. Conference talk at MathCrypt.

[QKL+21] Victoria de Quehen, Peter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E Stange. Improved torsion-point attacks on sidh variants. In *A*nnual International Cryptology Conference, pages 432{470. Springer, 2021.

[Ren18] Joost Renes. Computing isogenies between Montgomery curves using the action of (0, 0). In *P*ost-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9{11, 2018, Proceedings, PQCrypto 2018, pages 229{247, 2018. https://ia.cr/2017/1198.

[RS06] Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies, 2006. https://ia.cr/2006/145.

[Sch95] Rene Schoof. Counting points on elliptic curves over nite elds. *J*ournal de theorie des nombres de Bordeaux, 7(1):219{254, 1995.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *S*IAM Journal on Computing, 26(5):1484{1509, 1997.

[Sil86] Joseph H. Silverman. *T*he Arithmetic of Elliptic Curves, volume 106 of *G*radute Texts in Mathematics. Springer-Verlag, 1986.

[Sim05] Denis Simon. Quadratic equations in dimensions 4, 5 and more. *P*reprint, 2005.

[Sto03] Carl Stormer. Quelques proprietes arithmetiques des integrales elliptiques et leurs applications a la theorie des fonctions entieres transcendantes. *A*cta Mathematica, 27:185{208, 1903.

[Sto10] Anton Stolbunov. Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. *A*dvances in Mathematics of Communications, 4(2):215, 2010.

[Str76] Volker Strassen. Einige Resultate uber Berechnungskomplexitat. *J*ahresbericht der Deutschen Mathematiker-Vereinigung, 78:1{8, 1976.

[Tes06] Edlyn Teske. An elliptic curve trapdoor system. *J*ournal of cryptology, 19(1):115{133, 2006.

[The20] The PARI Group, Universite de Bordeaux. *PARI/GP version 2.11.4*, 2020. available from http://pari.math.u-bordeaux.fr/.

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In *T*heory of Cryptography Conference, pages 192{216. Springer, 2016.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *A*nnual International Conference on the Theory and Applications of Cryptographic Techniques, pages 755{784. Springer, 2015.

[UXT+22] Rei Ueno, Keita Xagawa, Yutaro Tanaka, Akira Ito, Junko Takahashi, and Naofumi Homma. Curse of re-encryption: A generic power/em analysis on post-quantum kems. *I* ACR Transactions on Cryptographic Hardware and Embedded Systems, pages 296{322, 2022.

[Vel71] J. Velu. Isogenies entre courbes elliptiques. *C*omptes rendus de l'Academie des Sciences, Series A-B, 273:A238{A241, 1971.

[Ven15] Daniele Venturi. Zero-knowledge proofs and applications, 2015.

[Vig06] M-F Vigneras. *A*rithmetique des algebres de quaternions, volume 800. Springer, 2006.

[Voi13] John Voight. Identifying the matrix ring: algorithms for quaternion algebras and quadratic forms. In *Q*uadratic and higher degree forms, pages 255{298. Springer, 2013.

[Voi18] John Voight. *Q*uaternion Algebras. Springer Graduate Texts in Mathematics series, 2018.

[Voi21] John Voight. *Q*uaternion algebras. Springer Nature, 2021.

[Wat69] William C. Waterhouse. Abelian varieties over nite elds. *A*nnales scienti ques de l'Ecole Normale Superieure, 2(4):521{560, 1969.

[Wes21] Benjamin Wesolowski. Orientations and the supersingular endomorphism ring problem. Cryptology ePrint Archive, Report 2021/1583, 2021. https://ia.cr/2021/1583.

[Wes22] Benjamin Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *F*OCS 2021-62nd Annual IEEE Symposium on Foundations of Computer Science, 2022.

[Wig07] Carl Severin Wigert. *S*ur l'ordre de grandeur du nombre des diviseurs d'un entier. Almqvist & Wiksell, 1907.

[YAJ+17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. A post-quantum digital signature scheme based on supersingular isogenies. In *I*nternational Conference on Financial Cryptography and Data Security, pages 163{181. Springer, 2017.

[ZG85] Don Zagier and B. Gross. On singular moduli. *J*ournal Fur Die Reine Und Angewandte Mathematik - J REINE ANGEW MATH, 1985:191{220, 01 1985.

[ZSP+18] Gustavo H. M. Zanon, Marcos A. Simplicio, Geovandro C. C. F. Pereira, Javad Doliskani, and Paulo S. L. M. Barreto. Faster isogeny-based compressed key agreement. In Tanja Lange and Rainer Steinwandt, editors, *P*ost-Quantum Cryptography, pages 248{268. Springer International Publishing, 2018.