Constructing the Deuring correspondence with applications to supersingular isogeny-based cryptography

Ray, Dimitrij

Award date:
2018

Technische Universiteit Eindhoven University of Technology

Department of Mathematics and Computer Science
Coding Theory and Cryptology Group

Constructing the Deuring Correspondence with Applications to Supersingular Isogeny-Based Cryptography

Master's Thesis

Dimitrij Ray

Supervisors:
prof. dr. Tanja Lange
dr. Chloe Martindale
Lorenz Panny, MSc

Eindhoven, July 2018

Abstract

The Diffie-Hellman key exchange is a public key cryptosystem which enables two parties to generate a shared secret. The cryptosystem is secure under the assumption that the discrete logarithm problem is computationally infeasible to solve given proper parameters. However, an adversary with access to a sufficiently powerful quantum computer may perform an attack using Shor's algorithm and solve the discrete logarithm problem in polynomial time. While currently (known) existing quantum computers are not yet powerful enough to perform such attacks, a lot of work is being done in research to develop algorithms capable of withstanding attacks from quantum computers. One such algorithm is the supersingular isogeny Diffie-Hellman key exchange algorithm (SIDH) proposed by Jao and De Feo (2011).

Part of the public information required in the SIDH algorithm is an 'initial' supersingular elliptic curve E_0 with coefficients in a finite field of the form $\mathbb{F}_{p^2}$ where p is a prime. Constructing such a curve is possible in polynomial time; however, constructing a random curve with the required properites is potentially hard. Costello, Longa, and Naehrig (2016) proposed to use $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_{p^2}$ where $p = 2^{372}3^{239} - 1$ thus simplified the implementation of SIDH.

An algorithm by Kohel, Lauter, Petit, and Tignol (2014) can be used to construct a random curve suitable for use in SIDH by constructing an isogeny from an ideal of a known endomorphism ring of a curve such as E_0. The problem which this algorithm solves is called the *constructive Deuring correspondence problem*. The algorithm is of interest due to its use in an attack by Galbraith, Petit, Shani, and Ti (2016), which breaks SIDH provided one can give an explicit description of the endomorphism ring of E_0 and another public curve E_A. This algorithm is also used by Galbraith, Petit, and Silva (2017) in their signature scheme.

This thesis seeks to investigate whether it is possible to achieve security and practicality by providing a way to construct a random initial curve. This project aims to give an implementation of the algorithm and study possible alternatives or optimizations to the algorithm. Under some weak conditions, this project is able to provide a successful implementation of the algorithm in Sage.

Acknowledgements

I wish to thank my supervisors: Tanja Lange, for introducing me to the vast expanse that is cryptology and offering her guidance and advice during this thesis project and internship; Chloe Martindale and Lorenz Panny, for their multitude of assistance in both theoretical and practical aspects throughout this thesis project in form of crash courses in algebra and programming help, and for always standing by whenever I entered the dreaded panic mode. I also thank Christophe Petit, who offered his insights to his algorithm and gave helpful hints during his visit to the TU/e. I would also like to thank Ruud Pellikaan, whose lecture back in Bandung gave me the idea of studying post-quantum cryptography.

I express deep gratitude for the friends I have acquired along my journey in the Netherlands: fellow international students, the Indonesian Student Association in Eindhoven (PPI/e), choristers of Vokollage and Dolce Bocca Choir, without whom I would not have lasted more than a day in the Netherlands. I would like to express special thanks to members of the Nederlands Studenten Kamerkoor 2018 project *Voyná i Mir / Oorlog en Vrede* for providing me with tremendous emotional support, companionship, unforgettable memories, and Dutch lessons (whether I want it or not).

I would like to thank my family for supporting my decision to go abroad and pursue my master's degree, especially my grandmother, who gave her blessings despite presumably knowing that she would never see me again. This master's thesis is dedicated to her memory.

Finally, I would like to thank the Indonesian Endowment Fund for Education (LPDP-RI) for fully supporting my master's degree through their scholarship program.

Contents

Chapter 1

Introduction

The Diffie-Hellman key exchange algorithm was first introduced by Whitfield Diffie and Martin Hellman in 1976 [7]. The algorithm was the first public-key cryptosystems to be published. The idea of the algorithm is that two parties can generate the same shared secret by combining their own private information with public information which are obtained either through agreement by both parties or sent from one party to another. Such an algorithm is immensely useful; for instance, it allows two parties to communicate using a symmetric-key cryptosystem without having to agree on a key in private.

Developments in quantum computing, however, threaten the security of the Diffie-Hellman key exchange. The algorithm is considered secure due to the fact that the best generic attacks run in exponential time. An adversary possessing a powerful enough quantum computer, however, may use an algorithm by Peter Shor [22] to break the protocol in polynomial time. This led to the development of cryptosystems that are believed to be resistant to attacks from quantum computers, commonly known as post-quantum cryptosystems.

Jao and De Feo proposed a Diffie-Hellman-like key exchange algorithm using isogenies of supersingular elliptic curves, commonly referred to as *Supersingular Isogeny Diffie-Hellman key exchange algorithm* (SIDH) [13]. At the time of this writing, the best known quantum attack for their algorithm still runs in exponential time. One of the parameters involved in the protocol, which is the crux of this thesis, is an initial supersingular elliptic curve E_0 over a finite field $\mathbb{F}_{p^2}$ where p is a prime such that the number of points in the curve is $(p+1)^2$. Constructing such a curve is solvable in polynomial time; however, constructing a *random* curve with the required properties is potentially hard. Thus, a proposal by Costello, Longa, and Naehrig [6] to use a specific curve simplifies the protocol's implementation.

One way to construct a random curve is using an algorithm first described by Kohel, Lauter, Petit, and Tignol [15]. The algorithm uses the observation that the endomorphism ring of a supersingular elliptic curve is isomorphic to an order in a quaternion algebra and that ideals in such algebra corresponds to isogenies. This algorithm is of interest due to its use in an attack by Galbraith, Petit, Shani, and Ti [9] which exploits the endomorphism rings of the initial ring E_0 and a public curve E_A. It is also used constructively as part of a signature scheme proposed by Galbraith, Petit, and Silva [10]. This thesis project aims to implement this algorithm as described in [10] using Sage [21].

Our main contribution is giving an implementation of the Kohel-Lauter-Petit-Tignol algorithm in Sage and a potential improvement described in Section 3.3. This thesis gives more details on the more practical aspects of the implementation which might not be readily available in Sage, such as constructing a random ideal of an order in a quaternion algebra. To the best of our knowledge, no other attempts at implementing this algorithm in Sage have been reported.

This thesis consists of 5 chapters. This first chapter is a general introduction to the problem and its setting, most of which will be elaborated in further chapters. Chapter 2 is devoted to some preliminaries on supersingular elliptic curves and the SIDH algorithm. Chapter 3 builds upon the materials in Chapter 2 and elaborates more on the nature of endomorphism rings of supersingular

elliptic curves, the Deuring correspondence, and the aforementioned algorithm by Kohel, Lauter, Petit, and Tignol. We also describe a potential improvement to the algorithm in Chapter 3. Chapter 4 provides the reader with some details of our implementation of the algorithm and a brief report on its performance. Finally, we give our conclusions and our suggestions for future research in Chapter 5.

Chapter 2

Supersingular Elliptic Curves and the SIDH Algorithm

Before we begin our discussion on the constructive Deuring correspondence, we will first introduce the necessary notions related to supersingular elliptic curves, especially isogenies and the endomorphism ring of an elliptic curve. We will also briefly explain the SIDH protocol in order to give some context on the relevance of implementing the constructive Deuring correspondence.

2.1 Supersingular elliptic curves

We begin by defining elliptic curves.

Definition 2.1. An elliptic curve over a field K is a nonsingular projective curve of genus one with a specified base point O. Equivalently, an elliptic curve is a nonsingular curve given by the (affine) Weierstrass equation

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{2.1}$$

where $a_1, \ldots, a_6 \in K$, together with a point at infinity O. [23]

When the field K is not of characteristic 2 or 3, one might perform the following admissible change of variables to Equation 2.1

$$(x, y) \mapsto \left(\frac{x - 3a_1^2 - 12a_2}{36}, \ \frac{y - 3a_1x}{216} - \frac{a_1^3 + 4a_1a_2 - 12a_3}{24} \right)$$

to obtain the short Weierstrass equation

$$y^2 = x^3 + Ax + B. \tag{2.2}$$

where $A, B \in K$. Unless otherwise noted, we will use the form in Equation 2.2 throughout this thesis.

When an elliptic curve is written in short Weierstrass form, we define the *discriminant* Δ and the *j-invariant* of a curve as

$$\Delta = -16(4A^3 + 27B^2) \quad \text{and} \quad j = -1728\frac{(4A)^3}{\Delta}.$$

A curve is *singular* if and only if its discriminant $\Delta = 0$; hence, elliptic curves are precisely the curves given by Equation 2.2 with $\Delta \neq 0$. The j-invariant can be used to identify curves that are isomorphic over an algebraic closure $\bar{K}$ of K: two elliptic curves are isomorphic over $\bar{K}$ if and only if both curves have the same j-invariant. We will see later that the j-invariant is important to the SIDH protocol.

2.1.1 The group law

There exists a group law on points of an elliptic curve, which is easily described using the *chord-and-tangent rule*. Let E be an elliptic curve over a field K where the characteristic of K is different from 2 or 3.

Point negation. Let P be a point on E. The negation of a point P, denoted $-P$, is constructed as follows: draw a line between P and O, which will intersect the curve on a third point. This is the point $-P$.

Point addition. Let P and Q be distinct points on E. The *sum* of these points, $P+Q$, is then constructed as follows:

1. Draw a line passing through P and Q. Since E is a cubic equation, this line will intersect the curve in a unique third point R (counted with multiplicities) on E.

2. The point $P+Q$ is defined to be $-R$.

Point doubling. Let P be a point on E. The *double* of this point, denoted $P+P$ or $[2]P$, is constructed as follows:

1. Draw a tangent line through P. This line will intersect the curve in another point R on E.

2. The point $[2]P$ is then defined to be $-R$.

Similarly, one can also define the *scalar multiple* or the *multiplication-by-m map* of a point P. Let m be an integer, then

$$[m]P = \underbrace{P + P + \cdots + P}_{m \text{ times}}.$$

Given an elliptic curve defined by a short Weierstrass equation, it is possible to compute the results of the operations above explicitly. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be points on an elliptic curve and let $x(P)$ denote the x-coordinate of P. We then have the following formulas:

$$P + Q = \left(\lambda^2 - x_1 - x_2,\ \lambda(x_1 - x(P+Q)) - y_1\right),$$

$$[2]P = \left(\left(\frac{3x_1^2 + A}{2y_1}\right)^2 - 2x_1,\ \left(\frac{3x_1^2 + A}{2y_1}\right)(x_1 - x([2]P)) - y_1\right),$$

$$-P = (x_1, -y_1),$$

where $P \neq \pm Q$ and $\lambda = (y_2 - y_1)/(x_2 - x_1)$. Under point addition, the set of points on an elliptic curve forms an abelian group.

We note that the chord-and-tangent rule is a geometric description of the group operation of points in an elliptic curve. For the proof that the points form an abelian group and a more algebraic description of the group law, we refer the reader to [23].

2.1.2 Isogenies

Now that we have described the group law for points on an elliptic curve, we can define isogenies, which are a special kind of rational maps between curves.

Definition 2.2. Let E_1 and E_2 be elliptic curves. A *morphism* is a rational map from E_1 to E_2 that is defined at every point and preserves the group structure.

Definition 2.3. Let E_1 and E_2 be elliptic curves. An *isogeny* from E_1 to E_2 is a morphism

$$\phi : E_1 \to E_2$$

such that $\phi(O_{E_1}) = O_{E_2}$. Two elliptic curves E_1 and E_2 are *isogenous* if there exists an isogeny from E_1 to E_2 where $\phi(E_1) \neq \{O_{E_2}\}$.

A morphism of curves is either constant or surjective [23]. Hence, an isogeny ϕ satisfies either

$$\phi(E_1) = E_2 \quad \text{or} \quad \phi(E_1) = \{O_{E_2}\}.$$

A non-constant isogeny $\phi : E_1 \to E_2$ induces an injection of function fields

$$\phi^* : \bar{K}(E_2) \to \bar{K}(E_1)$$

where $\bar{K}(E)$ denotes the function field of the curve E over $\bar{K}$. The *degree* of the isogeny ϕ, denoted $\deg \phi$, is the degree of the extension $\bar{K}(E_1)/\phi^*(\bar{K}(E_2))$. The degree of the zero isogeny $[0]$ is defined to be 0. An isogeny may be separable, inseparable, or purely inseparable depending on the corresponding property of the extension. For this project, we only consider separable isogenies. In particular, ϕ is a separable isogeny, then

$$\deg \phi = \# \ker(\phi) \ \ [23].$$

The degree of a composition of isogenies is *multiplicative*. Let $\phi : E_1 \to E_2$ and $\psi : E_2 \to E_3$, then

$$\deg(\psi \circ \phi) = \deg(\psi) \deg(\phi).$$

This latter property of degrees of isogenies is especially important. We will later see that one of the steps in SIDH involves computing isogenies of a large degree, making naïve computations of the isogeny time-consuming. However, provided we know the kernel and the desired degree of the isogeny, we can simply compute compositions of isogenies until the desired degree is reached due to the multiplicative property and the following propositions.

Proposition 2.1. *Let E be an elliptic curve and let Φ be a finite subgroup of E. There is a unique elliptic curve E' (up to isomorphism) and a separable isogeny $\phi : E \to E'$ satisfying* $\ker \phi = \Phi$.

Proof. See [23]. □

Proposition 2.2. *Let $\phi : E_1 \to E_2$ and $\psi : E_1 \to E_3$ be nonconstant isogenies and assume that ϕ is separable. If* $\ker \phi \subset \ker \psi$, *then there is a unique isogeny $\lambda : E_2 \to E_3$ satisfying $\psi = \lambda \circ \phi$.*

Proof. See [23]. □

An especially important example of an isogeny is the multiplication-by-m map $[m]$. With the exception of $[0]$, this map is surjective, and hence its degree is nonzero. In fact, the degree of this map is simply

$$\deg\, [m] = m^2.$$

We now define the m-torsion subgroup of an elliptic curve.

Definition 2.4. Let E/K be an elliptic curve, $m \in \mathbb{Z}$, $m \geq 1$. The *m-torsion subgroup* of E, denoted by $E[m]$, is the set

$$E[m] = \ker\, [m] = \{P \in E(\bar{K}) : [m]P = O\}.$$

It turns out that this subgroup has a nice algebraic structure, given in the following proposition.

Proposition 2.3. *Let E/K be an elliptic curve and let $m \in \mathbb{Z}$ with $m \neq 0$.*

1. *If* $\text{char}(K) = 0$ *or* $p = \text{char}(K) > 0$ *and* $p \nmid m$*, then*

$$E[m] \cong \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}.$$

2. *If* $\text{char}(K) = p > 0$*, then one of the following is true.*

   (a) $E[p^e] = \{O\}$ *for all* $e \in \mathbb{Z}_{>0}$

   (b) $E[p^e] \cong \mathbb{Z}/p^e\mathbb{Z}$ *for all* $e \in \mathbb{Z}_{>0}$.

Proof. See [23]. □

Recall that the set of points on an elliptic curve forms an abelian group under point addition. Isogenies are *homomorphisms* between abelian groups. Let E_1 and E_2 be elliptic curves. The set of all isogenies between E_1 and E_2 is denoted $\text{Hom}(E_1, E_2)$. Let $\phi, \psi \in \text{Hom}(E_1, E_2)$. We define the sum of two isogenies by

$$(\phi + \psi)(P) = \phi(P) + \psi(P)$$

for $P \in E_1$. The set of all homomorphisms between the curves E_1 and E_2, denoted $\text{Hom}(E_1, E_2)$ forms a group under addition [23].

The curves E_1 and E_2 need not be distinct. If $E_1 = E_2 = E$, we call the homomorphism an *endomorphism* and we denote the set of all endomorphisms of an elliptic curve E as $\text{End}(E)$. Moreover, the group $\text{End}(E)$ forms a (not necessarily commutative) ring: the addition law is as previously described, while the multiplication law is simply a composition of isogenies, defined by

$$\phi\psi(P) = (\phi \circ \psi)(P)$$

where $P \in E$. Throughout this thesis, we will refer to this ring as the *endomorphism ring* of E.

The following theorem shows that the endomorphism ring of an elliptic curve also has a neat algebraic structure.

Theorem 2.1. *Let E be an elliptic curve defined over a field K. The endomorphism ring of E is either:*

1. *the ring $\mathbb{Z}$,*

2. *an order in an imaginary quadratic field, or*

3. *a maximal order in a quaternion algebra.*

If $\text{char}(K) = 0$*, only the first two are possible.*

Proof. See [23] and [25]. □

We are finally ready to define supersingular elliptic curves.

Definition 2.5. A *supersingular elliptic curve* is an elliptic curve whose endomorphism ring is isomorphic to an order in a quaternion algebra.

Remark. The name *supersingular* has nothing to do with the existence of a singular point on the curve. The fact that it is an elliptic curve implies that it has no such point. In this instance, the word *singular* refers to the fact that supersingular elliptic curves are sparse or unusual.

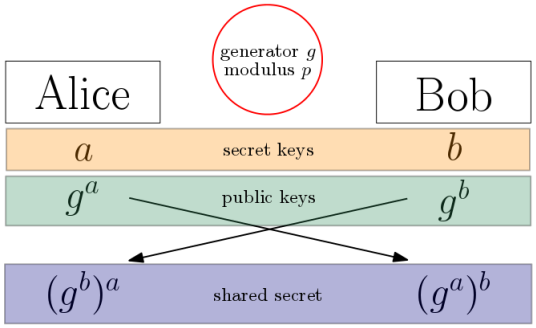

Figure 2.1: Illustration of the Diffie-Hellman key exchange

2.2 The SIDH algorithm

2.2.1 The Diffie-Hellman key exchange

The original key exchange algorithm was introduced in [7]. The algorithm involves two parties who exchange publicly available information. This information is combined with each individual party's private information to create a shared secret between the two parties. One description of the algorithm is as follows.

Setup Both parties agree on a prime modulus p and a generator g of $\mathbb{F}_p^*$.

Key Generation Alice chooses a private key $a \in \mathbb{Z}$, $0 < a < p - 1$. Bob does the same and chooses a private key $b \in \mathbb{Z}$, $0 < b < p - 1$. Alice then computes g^a modulo p and sends it to Bob. Bob also computes g^b modulo p and sends it to Alice.

Key Exchange Alice computes $(g^b)^a$ modulo p. Bob computes $(g^a)^b$ modulo p. The number $(g^a)^b = (g^b)^a$ is Alice and Bob's shared secret.

An illustration of the algorithm as previously described is shown in Figure 2.1.

Although the Diffie-Hellman key exchange as described above uses the group $\left(\mathbb{F}_p^*, \cdot\right)$, the algorithm is easily adaptable to other cyclic (sub)groups. One such group is, unsurprisingly, the group of points on an elliptic curve under the group law described in Section 2.1.1. This group is the basis of the appropriately named Elliptic-Curve Diffie Hellman (ECDH) protocol.

Unfortunately, the Diffie-Hellman key exchange is not quantum-resistant. The security of this protocol depends on the computational infeasibility of solving the *discrete logarithm problem*. Given a cyclic group $(G, \cdot)$, a generator g, and an element $h \in G$, the problem asks for an integer m such that $g^m = h$. Given that the algorithm is implemented in such a way such that there are no protocol failures and the parameters are suitable, the best known generic attack for the Diffie-Hellman key exchange runs in $O(\sqrt{\pi n/2})$ without parallelization, where n is the size of the input; for instance, the number of $\mathbb{F}_p$-rational points on the elliptic curve [20], i.e. the attack runs in exponential time. However, if the adversary has access to a sufficiently powerful quantum computer, they can then run Shor's algorithm, which solves the discrete logarithm problem with time complexity polynomial in $\log n$. Therefore, once a practical quantum computer is available, all cryptosystems that rely on the hardness of the discrete logarithm problem are broken.

All is not lost, however, since Jao and De Feo proposed a quantum-resistant algorithm for key exchange. Their scheme uses the supposed hardness of computing large degree isogenies between supersingular elliptic curves. The best known quantum attack against this protocol takes $O(p^{1/6})$ time [13], thus making this algorithm a candidate for post-quantum key exchange.

2.2.2 The Jao-De Feo algorithm

We now describe Jao and De Feo's algorithm [13].

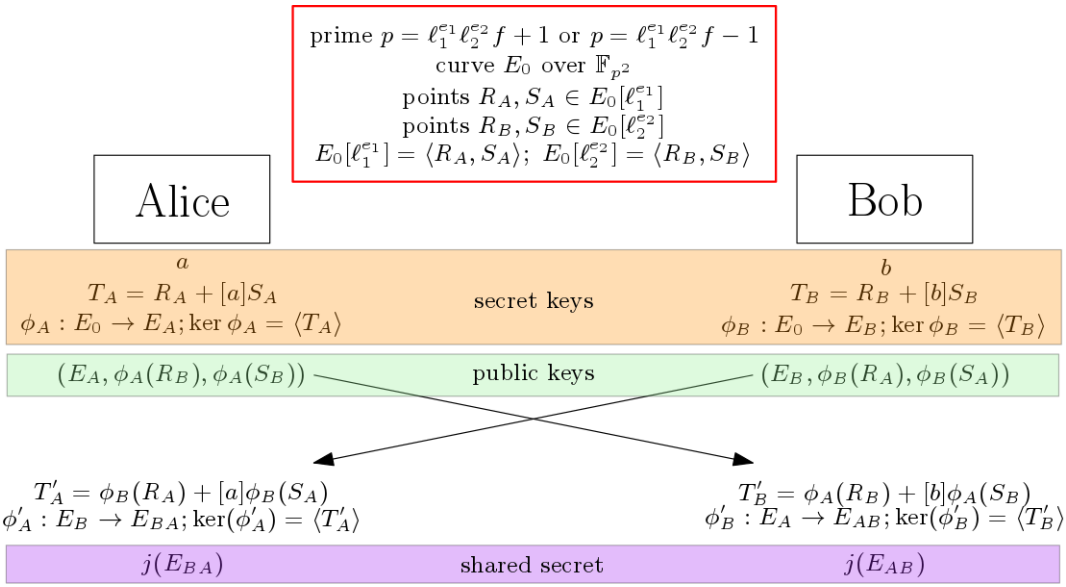

Figure 2.2: Illustration of the SIDH algorithm

Setup Both parties agree on distinct small primes ℓ_1 and ℓ_2 and exponents e_1 and e_2 such that $\ell_1^{e_1} \approx \ell_2^{e_2} \approx 2^\lambda$ where λ is the desired security parameter. A cofactor $f \in \mathbb{N}$ is then chosen such that either $p = \ell_1^{e_1}\ell_2^{e_2}f + 1$ or $p = \ell_1^{e_1}\ell_2^{e_2}f - 1$ is prime. Both parties then agree on an elliptic curve E_0 over $\mathbb{F}_{p^2}$ such that the number of $\mathbb{F}_{p^2}$-rational points on E_0 is equal to $(\ell_1^{e_1}\ell_2^{e_2}f)^2$. Finally, both parties fix points $R_A, S_A \in E_0[\ell_1^{e_1}]$ such that $\langle R_A, S_A\rangle = E_0[\ell_1^{e_1}]$ and two points $R_B, S_B \in E_0[\ell_2^{e_2}]$ such that $\langle R_B, S_B\rangle = E_0[\ell_2^{e_2}]$.

Key Generation Alice first chooses a secret integer $0 \leq a < \ell_1^{e_1}$, computes $T_A = R_A + [a]S_A$, then computes an isogeny $\phi_A : E \to E_A$ where $\ker(\phi_A) = \langle T_A\rangle$. Alice then publishes $(E_A, \phi_A(R_B), \phi_A(S_B))$. Similarly, Bob chooses a secret integer $0 \leq b < \ell_2^{e_2}$, computes $T_B = R_B + [b]S_B$, then computes an isogeny $\phi_B : E \to E_B$ where $\ker(\phi_B) = \langle T_B\rangle$. Bob then publishes $(E_B, \phi_B(R_A), \phi_B(S_A))$.

Key Exchange Alice computes $T'_A = \phi_B(R_A) + [a]\phi_B(S_A)$ and an isogeny $\phi'_A : E_B \to E_{BA}$ where $\ker(\phi'_A) = \langle T'_A\rangle$. Similarly, Bob computes $T'_B = \phi_A(R_B) + [b]\phi_A(S_B)$ and an isogeny $\phi'_B : E_A \to E_{AB}$ where $\ker(\phi'_B) = \langle T'_B\rangle$. Their shared secret is $j(E_{BA}) = j(E_{AB})$.

A diagram of this algorithm is provided in Figure 2.2.

It is not obvious from the description of this algorithm that the shared secret is the same for both Alice and Bob, i.e. that $j(E_{AB}) = j(E_{BA})$. Indeed, the elliptic curves E_{AB} and E_{BA} computed by Alice and Bob at the end of the algorithm are likely not the same curve. However, from Figure 2.2 one might observe that

$$\ker(\phi'_A \circ \phi_A) = \ker(\phi'_B \circ \phi_B) = \langle T_A, T_B\rangle.$$

Hence, by Proposition 2.1, the curves E_{AB} and E_{BA} are isomorphic and therefore $j(E_{AB}) = j(E_{BA})$ [13].

The security of SIDH depends on the difficulty of the following problem.

Definition 2.6 (SIDH isogeny problem). Let (E, R_A, S_A, R_B, S_B) be SIDH public parameters and let Alice's public information (E_A, R'_B, S'_B) be given. Compute an isogeny $\phi : E \to E_A$ of degree $\ell_1^{e_1}$ such that $\phi(R_B) = R'_B$ and $\phi(S_B) = S'_B$. [11]

The most common way to analyze the difficulty of this problem is by considering the *ℓ-isogeny graph*, defined as follows. Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected graph, where the vertex set $\mathcal{V}$ is the

set of all possible j-invariants over $\bar{K}$. On an ℓ-isogeny graph, there exists an edge between two vertices v_1, v_2 if and only if there exists an isogeny of degree ℓ from a curve with j-invariant v_1 to a curve with j-invariant v_2. Supersingular curves form a connected component, an $(\ell+1)$-regular graph, in the full isogeny graph. In fact, it is a Ramanujan graph: it has "good mixing properties" and there is a "short" path between any two vertices in the graph. [11]

The process of computing Alice's (and similarly Bob's) secret isogeny ϕ_A of degree $\ell_1^{e_1}$ can be seen as an e_1-step pseudorandom walk in an ℓ_1-isogeny graph due to multiplicativity of isogeny degrees and Propositions 2.1 and 2.2. The above properties of a Ramanujan graph ensure that if we have an isogeny with a large enough degree ℓ^e, it is computationally expensive to find the path that the random walk took in the ℓ-isogeny graph given only the endpoints – and hence, it is infeasible to solve the SIDH isogeny problem given only the public information. For a more extensive discussion on the security of SIDH, see [13] and [11].

Chapter 3

Constructive Deuring Correspondence

The problem of finding a supersingular curve with a specified number of points over $\mathbb{F}_p$ or $\mathbb{F}_{p^2}$, where p is prime, is solvable in polynomial time using the algorithm described in [2]. However, the algorithm is deterministic. When $p \equiv 3 \mod 4$, for instance, the algorithm will always output the curve

$$y^2 = x^3 - x$$

as a result. In this chapter, we will describe an algorithm to compute a random elliptic curve satisfying the requirements for SIDH.

In the previous chapter, it has been stated that the endomorphism ring of a supersingular elliptic curve is isomorphic to an order in a quaternion algebra. One might suppose that if we can somehow construct a random such order, we are able to obtain a random curve. This is possible, for instance, using [4]. However, the aforementioned algorithm runs in $O(p^{1+\epsilon})$, and is hence impractical for the choice of SIDH curves, where p has size approximately 2^λ for some big λ. The problem of constructing the endomorphism ring itself is considered a hard problem [19]. Fortunately, it is still possible to use this information and instead look at isogenies which correspond to *ideals* of a maximal order in a quaternion algebra using Deuring's correspondence.

3.1 Quaternion algebra and Deuring's correspondence

We first begin by describing quaternion algebras and some of their properties which will be relevant for the following parts of this thesis.

Definition 3.1. A *quaternion algebra* B over a field K not of characteristic 2 is an algebra with basis 1, $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ for B as a K-vector space, such that

$$\mathbf{i}^2 = a,\ \mathbf{j}^2 = b, \quad \text{and } \mathbf{k} = \mathbf{ij} = -\mathbf{ji}$$

for some fixed a, $b \in K^*$. This quaternion algebra is denoted $\left(\frac{a,b}{K}\right)$. [25]

A famous example of a quaternion algebra is *Hamilton's quaternions* $\mathbb{H} = \left(\frac{-1,-1}{\mathbb{R}}\right)$ first described by William Rowan Hamilton, who allegedly came up with the idea while walking along the Royal Canal in Dublin with his wife [25]. This quaternion algebra satisfies

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1.$$

The properties of the basis elements as stated in Definition 3.1 give us a multiplication table to be used as reference when multiplying elements in a quaternion algebra, shown in Figure 3.1.

$\times$	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
1	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
$\mathbf{i}$	$\mathbf{i}$	a	$\mathbf{k}$	$a\mathbf{j}$
$\mathbf{j}$	$\mathbf{j}$	$-\mathbf{k}$	b	$-b\mathbf{i}$
$\mathbf{k}$	$\mathbf{k}$	$-a\mathbf{j}$	$b\mathbf{i}$	$-ab$

Figure 3.1: Multiplication table for quaternion algebra $\left(\frac{a,b}{K}\right)$. The row index indicates the first factor. The column index indicates the second factor.

This immediately introduces a caveat: in general, quaternion multiplication is **not commutative**, which is apparent from the multiplication table. Hence, we exercise caution when multiplying two elements in a quaternion algebra. A special case is when $\alpha \in K$ and $\beta \in B$; in this case, indeed $\alpha\beta = \beta\alpha$.

Let α be an element in a quaternion algebra. We can then express α in the following way:

$$\alpha = t + x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$$

where t, x, y, and z are elements of K. The *standard involution* or *conjugation* $\bar{\alpha}$ of α is defined to be

$$\bar{\alpha} = t - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}.$$

The conjugation is clearly K-linear. It is also obvious from the definition that $\bar{1} = 1$ and $\overline{\bar{\alpha}} = \alpha$. Furthermore, it can be shown that $\overline{\alpha\beta} = \bar{\beta}\bar{\alpha}$ for all α, β in the quaternion algebra.

Definition 3.2. Let α be an element of a quaternion algebra. The reduced norm and the reduced trace of α are

$$\mathrm{nrd}(\alpha) = \alpha\bar{\alpha} \quad \text{and} \quad \mathrm{trd}(\alpha) = \alpha + \bar{\alpha},$$

respectively.

For the sake of brevity, throughout this thesis we will use *norm* and *trace* to refer to the reduced norm and reduced trace.

Let $B = \left(\frac{a,b}{K}\right)$ and $\alpha \in B$. Writing α as above, we can give explicit formulas for the norm and the trace as follows:

$$\mathrm{nrd}(\alpha) = \alpha\bar{\alpha} = (t + x\mathbf{i} + y\mathbf{j} + z\mathbf{k})(t - x\mathbf{i} - y\mathbf{j} - z\mathbf{k}) = t^2 - ax^2 - by^2 + abz^2,$$

$$\mathrm{trd}(\alpha) = \alpha + \bar{\alpha} = t + x\mathbf{i} + y\mathbf{j} + z\mathbf{k} + t - x\mathbf{i} - y\mathbf{j} - z\mathbf{k} = 2t.$$

We observe from these explicit formulas that $\mathrm{nrd}(\alpha)$ and $\mathrm{trd}(\alpha)$ are both in K. It is easily seen that the trace is additive, since for $\alpha, \beta \in B$:

$$\mathrm{trd}(\alpha + \beta) = \alpha + \beta + \overline{\alpha + \beta} = \alpha + \beta + \bar{\alpha} + \bar{\beta} = \alpha + \bar{\alpha} + \beta + \bar{\beta} = \mathrm{trd}(\alpha) + \mathrm{trd}(\beta).$$

On the other hand, the norm is multiplicative:

$$\mathrm{nrd}(\alpha\beta) = \alpha\beta\overline{\alpha\beta} = \alpha\beta\bar{\beta}\bar{\alpha} = \mathrm{nrd}(\beta)\alpha\bar{\alpha} = \mathrm{nrd}(\alpha)\,\mathrm{nrd}(\beta).$$

Here we use the fact that $\mathrm{nrd}(\alpha) \in K$ for all $\alpha \in B$, and thus commutes with any element of the quaternion algebra.

Finally, we define the notion of an order in a quaternion algebra.

Definition 3.3. Let B be a $\mathbb{Q}$-vector space. A *lattice* $M \subset B$ is a finitely generated $\mathbb{Z}$-submodule with $M\mathbb{Q} = B$. [25]

Definition 3.4. Let B be a finite-dimensional $\mathbb{Q}$-algebra. An *order* $\mathcal{O} \subset B$ is a lattice that is also a subring of B. An order is *maximal* if it is not properly contained in another order. [25]

p	(a, b)
$3 \mod 4$	$(-1, -p)$
$5 \mod 8$	$(-2, -p)$
$1 \mod 8$	$(-q, -p)$; $q = 3 \mod 4$ and $\left(\frac{p}{q}\right) = -1$

Figure 3.2: Possible choices for (a, b) of the quaternion algebra over different possible values of p.

We recall that the endomorphism ring of a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$ is isomorphic to an order in a quaternion algebra. The quaternion algebra in question is the quaternion algebra $\left(\frac{a,b}{\mathbb{Q}}\right)$ ramified at p and at infinity, denoted $B_{p,\infty}$. The actual values of a and b vary with p, as shown in Figure 3.2. For convenience, throughout the following discussions we will only consider the case $p \equiv 3 \pmod 4$, although most of the results contained are applicable for other values of p. [8]

A fundamental result due to Deuring showed that the endomorphism ring is isomorphic to a maximal order in the quaternion algebra $B_{p,\infty}$. It is also true that for every maximal order in B, there exists a supersingular elliptic curve whose endomorphism ring is isomorphic to it [25]. This correspondence between the endomorphism ring of an elliptic curve and a maximal order in a quaternion algebra is called the *Deuring correspondence*. Furthermore, let $\mathcal{O}$ be a maximal order in B and I be any left $\mathcal{O}$-ideal. Such an ideal I corresponds to an isogeny

$$\phi_I : E \to E_I \quad \text{where } \ker \phi_I = \{P \in E : \alpha(P) = O \ \forall \alpha \in I\} = \bigcap_{\alpha \in I} \ker \alpha. \tag{3.1}$$

We also have the following proposition.

Proposition 3.1. *Let E be an elliptic curve. Let $\mathcal{O}$ be a maximal order in a quaternion algebra B isomorphic to the endomorphism ring of E. For every isogeny $\phi : E \to E'$, there exists a left $\mathcal{O}$-ideal I and an isomorphism $\rho : E_I \to E'$ such that $\phi = \rho\phi_I$.*

Proof. See [25]. □

Let $\mathcal{O}'$ be a maximal order in a quaternion algebra. The problem of constructing an elliptic curve whose endomorphism ring is isomorphic to $\mathcal{O}'$ is called the *constructive Deuring correspondence* problem [19]. The following problem is equivalent [10]: given a curve E with a known endomorphism ring $\mathcal{O}$ and a left $\mathcal{O}$-ideal I, construct an isogeny from E to another curve E' whose kernel is I. Proposition 3.1 guarantees that every curve E' isogenous with E can be constructed this way provided one finds the appropriate ideal.

There exists a polynomial-time algorithm to solve the constructive Deuring correspondence problem due to Kohel, Lauter, Petit, and Tignol [15]. We will describe the algorithm, its restrictions, and the implementation in the coming sections.

3.2 The Kohel-Lauter-Petit-Tignol algorithm

From now on, we let p be a prime with $p \equiv 3 \pmod 4$. Let $\mathcal{O}$ be a maximal order of the quaternion algebra $B = \left(\frac{-1,-p}{\mathbb{Q}}\right)$. We first note that the multiplication table given in Figure 3.1 now simplifies to

$\times$	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
1	1	$\mathbf{i}$	$\mathbf{j}$	$\mathbf{k}$
$\mathbf{i}$	$\mathbf{i}$	-1	$\mathbf{k}$	$-\mathbf{j}$
$\mathbf{j}$	$\mathbf{j}$	$-\mathbf{k}$	$-p$	$p\mathbf{i}$
$\mathbf{k}$	$\mathbf{k}$	$\mathbf{j}$	$-p\mathbf{i}$	$-p$

and the norm formula can now be written

$$\mathrm{nrd}(\alpha) = a^2 + b^2 + p(c^2 + d^2)$$

for $\alpha = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} \in B$. We also define the reduced norm of a left $\mathcal{O}$-ideal I as

$$\mathrm{nrd}(I) = \gcd(\{\mathrm{nrd}(\alpha) : \alpha \in I\}).$$

Finally, we define powersmooth numbers.

Definition 3.5. Let $n \in \mathbb{N}$ and $s \in \mathbb{R}^+$. The number n is said to be *s-powersmooth* if any divisor p^k of n for a prime p satisfies $p^k \leq s$.

We now describe the algorithm due to Kohel, Lauter, Petit and Tignol. Let $\mathcal{O}_0$ be the maximal order that is generated as a $\mathbb{Z}$-module as

$$\mathcal{O}_0 = \left\langle 1, \mathbf{i}, \frac{1+\mathbf{k}}{2}, \frac{\mathbf{i}+\mathbf{j}}{2} \right\rangle \subseteq B.$$

The order $\mathcal{O}_0$ is isomorphic to the endomorphism ring of the curve

$$E_0 : y^2 = x^3 + x. \quad [10]$$

We assume that we are already given a left $\mathcal{O}_0$-ideal. The idea of the algorithm is to first construct an ideal with a prescribed powersmooth norm, then use Deuring's correspondence to compute an isogeny from E_0 which corresponds to the ideal. We give an overview of the algorithm as described in [10] as follows.

Input: the curve E_0; the maximal order $\mathcal{O}_0$; a left $\mathcal{O}_0$-ideal I.
Output: the curve E' isomorphic to E/I.

1. Compute the ideal:
   (a) Compute an element $\delta \in I$ and an ideal $I' = I\bar{\delta}/\mathrm{nrd}(I)$ of some prime norm N.
   (b) Fix a powersmoothness bound $s = (7/2)\log p$ and an odd s-powersmooth number S. Find $\beta \in I'$ with norm NS.
   (c) Output $J = I'\bar{\beta}/N$.
2. Compute the isogeny:
   (a) Write the norm of J as its prime factorization $\mathrm{nrd}(J) = \prod_{i=1}^{r} \ell_i^{e_i}$ and write $J = \langle \alpha_1, \alpha_2, \alpha_3, \alpha_4 \rangle$.
   (b) Let $\varphi_0 = [1]_{E_0}$. For every $1 \leq i \leq r$:
      i. Compute a basis (P_i, Q_i) of $E_0[\ell_i^{e_i}]$.
      ii. For every generator α_k of J, compute $\alpha_k(P_i)$ and $\alpha_k(Q_i)$.
      iii. Find a point R_i of order ℓ_i such that $\alpha_k(R_i) = O$ for all k. This point generates $\ker \phi_I \cap E_0[\ell_i^{e_i}]$.
      iv. Compute an isogeny ϕ_i with kernel generated by $\varphi_{i-1}(R_i)$, then compute the composition $\varphi_i = \phi_i \varphi_{i-1}$.
3. Compute E' using $\varphi_r : E_0 \to E'$.

We will now go over the main steps of the algorithm.

Constructing an ideal of prime norm

The first step in the algorithm is to construct a left ideal I', which is *equivalent* to the input ideal I but with a prime norm N; that is, we are looking for an element $q \in B$ such that $I' = Iq$ and $\mathrm{nrd}(I') = N$. The method of its construction, as well as its existence, is given by the following lemma.

Lemma 3.1. *Let I be a left $\mathcal{O}$-ideal of reduced norm $\mathrm{nrd}(I)$ and δ an element of I, then $I\gamma$, where $\gamma = \bar{\delta}/\mathrm{nrd}(I)$ is a left $\mathcal{O}$-ideal of norm $\mathrm{nrd}(\delta)/\mathrm{nrd}(I)$.*

Proof. See [15] □

Note that in the lemma, the norm of the new ideal need not be prime. The choice of finding an ideal with a prime norm is to simplify the next steps. This explains the first step of searching a random element until the desired ideal I' has prime norm.

Constructing an ideal of powersmooth norm

Our next goal is to construct an element of I' with a specified norm NS, where S is an odd powersmooth number. If such an element is found, we can use Lemma 3.1 to construct another ideal $J = I'\bar{\beta}/N$. The norm of this ideal is indeed powersmooth, since

$$\mathrm{nrd}(J) = \mathrm{nrd}(I) \cdot \frac{\mathrm{nrd}(\beta)}{N} = \frac{N^2 S}{N^2} = S.$$

The powersmooth norm of J will be important since we will require a factorization of $\mathrm{nrd}(J)$ to solve the discrete logarithm problem in the Deuring correspondence step. Unlike the previous step, we cannot simply pick a random $\beta \in I'$, since this element β has to have norm NS.

The process of obtaining β of a particular norm involves solving a sum-of-squares problem. We first describe the general problem and an algorithm to solve it.

Cornacchia's algorithm

Cornacchia's algorithm solves the following general problem: given positive integers d and m such that $\gcd(d, m) = 1$, determine integers (x, y) such that

$$x^2 + dy^2 = m.$$

The algorithm proceeds as follows.

1. Put $r_0 = m$ and $r_1^2 = -d \pmod{m}$, where $0 \leq r_1 \leq m/2$
2. Compute $r_{i+2} = r_i \pmod{r_{i+1}}$ recursively like in the Euclidean algorithm until an r_k where $r_k^2 < m$ is found.
3. If $(m - r_k^2)/d$ is a square integer, return $\left(r_k, \sqrt{(m - r_k^2)/d}\right)$.

This algorithm will always return a primitive solution (solutions where $\gcd(x, y) = 1$) if it exists, provided one tries all possible square roots of $-d \pmod{m}$. Otherwise, one might attempt to solve the equation

$$x^2 + dy^2 = \frac{m}{g^2}$$

for some square g^2 such that m/g^2 is an integer. If a solution is found, the solution to the original equation is then (gx, gy).

We refer the reader to [1], [3], and [17] for further discussion regarding Cornacchia's algorithm, as well as its proof of correctness.

The search for β

The first method to construct β, which we attempt during the course of investigating this algorithm, is to construct an element $\beta \in \mathcal{O}_0$, then do a brute force search for all β with norm NS such that $I'\bar{\beta} \subseteq N\mathcal{O}_0$. Constructing such a β can be done by writing $\beta = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ and solving for the quadratic norm equation

$$a^2 + b^2 + p(c^2 + d^2) = NS.$$

This can be done by first picking a random pair of integers (c, d) and then solving

$$a^2 + b^2 = NS - p(c^2 + d^2)$$

for (a, b) using Cornacchia's algorithm. This process is repeated until a suitable β is found. However, this solution is not efficient; we postpone the discussion for this result until the next chapter.

A better solution, as described in [10], is to write $\beta = \beta_1\beta_2'$, each with norms NS_1 and S_2 respectively, where S_1 and S_2 are powersmooth numbers – the reason for use of this notation shall become clear in a moment. To construct each β, we first write $I' = N\mathcal{O}_0 + \mathcal{O}_0\alpha$, where $\alpha \in I'$ such that $\gcd(N^2, \mathrm{nrd}(\alpha)) = N$. The element β_1 is then constructed just as above: pick a random pair of integers (c, d) and solve

$$a^2 + b^2 = NS_1 - p(c^2 + d^2)$$

for (a, b).

The search for β_2' is more complicated. We first find an element β_2 of the form $C\mathbf{j} + D\mathbf{k}$ which solves the following equation of ideals:

$$(\mathcal{O}_0\beta_1)\beta_2 = (\mathcal{O}_0\alpha) \mod N\mathcal{O}_0. \tag{3.2}$$

The next proposition analyzes the probability of solving this equation.

Proposition 3.2. *Let $\alpha \in I'$, $\beta_1 \in \mathcal{O}_0$, and $\beta_2 \in \mathbb{Z}\mathbf{j} + \mathbb{Z}\mathbf{k}$. Consider the equation of ideals*

$$(\mathcal{O}_0\beta_1)\beta_2 = (\mathcal{O}_0\alpha) \mod N\mathcal{O}_0.$$

1. *If N is inert, the equation is always solvable.*
2. *If N is split, it is solvable with probability $\dfrac{N^2 - 2N + 3}{(N+1)^2}$.*

We will prove Proposition 3.2 with the aid of the following lemmas.

Lemma 3.2. *The quotient ring $\mathcal{O}_0/N\mathcal{O}_0$ is a quaternion algebra over $\mathbb{Z}/N\mathbb{Z}$.*

Proof. The ring properties of $\mathcal{O}_0/N\mathcal{O}_0$ follow from the fact that it is a quotient ring. To prove that the quotient ring is an algebra, we now prove its compatibility with scalars. First, we observe that the elements of $\mathcal{O}_0/N\mathcal{O}_0$ can be written

$$\mathbb{Z}/N\mathbb{Z} + \mathbb{Z}/N\mathbb{Z}\mathbf{i} + \mathbb{Z}/N\mathbb{Z}\left(\frac{1+\mathbf{k}}{2}\right) + \mathbb{Z}/N\mathbb{Z}\left(\frac{\mathbf{i}+\mathbf{j}}{2}\right).$$

Indeed, for any $\lambda, \mu \in \mathbb{Z}/N\mathbb{Z}$ and any $x, y \in \mathcal{O}_0/N\mathcal{O}_0$, we have that $\lambda x \in \mathcal{O}_0/N\mathcal{O}_0$ and $(\lambda x)(\mu y) = (\lambda\mu)(xy)$. Therefore, $\mathcal{O}_0/N\mathcal{O}_0$ is an algebra over $\mathbb{Z}/N\mathbb{Z}$.

We now have the following change of coordinates from $\mathcal{O}_0/N\mathcal{O}_0$-coordinate to $(1, \mathbf{i}, \mathbf{j}, \mathbf{k})$-coordinate:

$$\zeta: \quad (\alpha, \beta, \gamma, \delta) \mapsto \left(\alpha + \frac{\gamma}{2}, \beta + \frac{\delta}{2}, \frac{\delta}{2}, \frac{\gamma}{2}\right)$$

and its inverse, the change of coordinates from $(1, \mathbf{i}, \mathbf{j}, \mathbf{k})$-coordinate to $\mathcal{O}_0/N\mathcal{O}_0$-coordinate:

$$\zeta^{-1}: \quad (a, b, c, d) \mapsto (a - d, b - c, 2d, 2c).$$

Since N is a prime, these changes of coordinates are well-defined. Moreover, since they are invertible, they are bijective. It follows that we can write the elements of $\mathcal{O}_0/N\mathcal{O}_0$ simply as

$$\mathbb{Z}/N\mathbb{Z} + \mathbb{Z}/N\mathbb{Z}\mathbf{i} + \mathbb{Z}/N\mathbb{Z}\mathbf{j} + \mathbb{Z}/N\mathbb{Z}\mathbf{k}.$$

Finally, it is clear that $\mathbf{i}^2 \equiv -1 \pmod{N}$, $\mathbf{j}^2 \equiv -p \pmod{N}$, and $\mathbf{k} = \mathbf{ij} = -\mathbf{ji}$. It follows that $\mathcal{O}_0/N\mathcal{O}_0$ is the quaternion algebra $\left(\frac{-1, -p}{\mathbb{Z}/N\mathbb{Z}}\right)$. □

Lemma 3.3. *The quotient ring $\mathcal{O}_0/N\mathcal{O}_0$ is isomorphic to the matrix ring $M_2(\mathbb{Z}/N\mathbb{Z})$.*

Proof. The quotient ring $\mathcal{O}_0/N\mathcal{O}_0$ is not a division ring, since elements of norm divisible by N are not units in this ring. By [5], Theorem 4.20, the quotient ring is isomorphic to $M_2(\mathbb{Z}/N\mathbb{Z})$. □

Corollary 3.1. *The quotient ring $\mathcal{O}_0/N\mathcal{O}_0$ has $N + 1$ nontrivial left ideals.*

Proof. The left ideals of the matrix ring $M_2(\mathbb{Z}/N\mathbb{Z})$ are all principal [16]. The nontrivial left ideals of the ring are the left ideals generated by 2×2 matrices of rank 1. Since swapping rows of the matrix does not change the generated ideal, we need only consider matrices of the form

$$\begin{bmatrix} a & b \\ 0 & 0 \end{bmatrix}$$

where $a, b \in \mathbb{Z}/N\mathbb{Z}$ where a and b are not both zero. There are $N^2 - 1$ such matrices. Since matrices obtained by multiplication with $(\mathbb{Z}/N\mathbb{Z})^*$ also generate the same ideal, it follows that there are $(N^2 - 1)/(N - 1) = N + 1$ different nontrivial left ideals of $M_2(\mathbb{Z}/N\mathbb{Z})$. Hence, there are $N + 1$ different nontrivial left ideals in the quotient ring $\mathcal{O}_0/N\mathcal{O}_0$. □

Lemma 3.4. *Let R be the ring $\mathbb{Z} + \mathbb{Z}\mathbf{i}$ and let $\mathcal{L}$ be the set of all nontrivial left $\mathcal{O}_0$-ideals. The map*

$$\begin{aligned} \rho: \quad \mathcal{L} \times (R/NR)^* &\to \mathcal{L} \\ (I, \beta) &\mapsto I\beta \end{aligned}$$

is a group action whose kernel is $(\mathbb{Z}/N\mathbb{Z})^*$.

1. *If N is split in R, the group action has an orbit of size $N - 1$ and two fixed points.*
2. *If N is inert in R, the group action has only one orbit.*

Proof. From Corollary 3.1, the left ideals of $\mathcal{O}_0/N\mathcal{O}_0$ are principal. Let $I \in \mathcal{L}$ be generated by an element α and let $\beta \in (R/NR)^*$. Clearly,

$$\rho(I, \beta) = I\beta = (\mathcal{O}_0\alpha)\beta = \mathcal{O}_0(\alpha\beta) \pmod{N\mathcal{O}_0}$$

which is in $\mathcal{L}$. It also follows that for all $I \in \mathcal{L}$, $I \cdot 1 = I$ and

$$\rho(\rho(I, \beta), \gamma) = I\beta\gamma = \rho(I, \beta\gamma)$$

for $\beta, \gamma \in (R/NR)^*$. Hence ρ is a group action. The fact that $(\mathbb{Z}/N\mathbb{Z})^*$ is the kernel of this group action follows from the fact that these are the only elements of $(R/NR)^*$ which commute with generators of any left ideal $I \in \mathcal{L}$.

Let N be split in R. We can then write $N = (a + b\mathbf{i})(a - b\mathbf{i})$ where $a, b \in \mathbb{Z}$. We observe that the ideal generated by $a + b\mathbf{i}$ and the ideal generated by $a - b\mathbf{i}$ are fixed points under ρ, since these elements commute with $(R/NR)^*$. Any other ideal in $\mathcal{L}$ is only stabilized by $(\mathbb{Z}/N\mathbb{Z})^*$, again due

to the fact that they commute with the generator. Moreover, any element of R/NR other than multiples of $a + b\mathbf{i}$ and $a - b\mathbf{i}$ are units. Therefore, by the orbit-stabilizer theorem, the size of the orbits of these ideals is

$$\frac{|(R/NR)^*|}{|(\mathbb{Z}/N\mathbb{Z})^*|} = \frac{N^2 - 1 - 2(N-1)}{N-1} = \frac{(N-1)^2}{N-1} = N - 1.$$

Since there are $N + 1$ elements in $\mathcal{L}$, we conclude that the group action has an orbit of size $N - 1$ and two fixed points.

Let N be inert in R. It follows that every element of R/NR is a unit. Therefore, by the orbit-stabilizer theorem, the size of the orbit of any ideal in $\mathcal{L}$ is

$$\frac{|(R/NR)^*|}{|(\mathbb{Z}/N\mathbb{Z})^*|} = \frac{N^2 - 1}{N-1} = N + 1.$$

It follows that the group action has only one orbit. □

We are now ready to prove Proposition 3.2.

Proof of Proposition 3.2. Solving Equation 3.2 for β_2 is equivalent to finding an element β_2 whose action sends $\mathcal{O}_0\beta_1$ to $\mathcal{O}_0\alpha$. We are looking for an element of the form $\mathbb{Z}\mathbf{j} + \mathbb{Z}\mathbf{k} = (\mathbb{Z} + \mathbb{Z}\mathbf{i})\mathbf{j}$. Since $\mathbf{j}$ is a unit, it simply permutes the left ideals that are not the fixed points.

If N is inert, from Lemma 3.4, there is only one orbit. Hence, there will always be an element whose action sends $\mathcal{O}_0\beta_1$ to $\mathcal{O}_0\alpha$. It follows that Equation 3.2 is always solvable.

If N is split, Equation 3.2 is guaranteed to be solvable if the ideals involved in both sides of the equation are not the fixed points. From Lemma 3.4, there are $N - 1$ ideals that are not the fixed points. Hence, the probability that this occurs is

$$\left(\frac{N-1}{N+1}\right)^2.$$

Otherwise, the two fixed points are permuted to each other by $\mathbf{j}$, since

$$\begin{aligned}(\mathcal{O}_0(a + b\mathbf{i}))(\mathbb{Z} + \mathbb{Z}\mathbf{i})\mathbf{j} &= \mathcal{O}_0(a + b\mathbf{i})\mathbf{j} \pmod{N\mathcal{O}_0} \\ &= \mathcal{O}_0(a\mathbf{j} - b\mathbf{j}\mathbf{i}) \pmod{N\mathcal{O}_0} \\ &= \mathcal{O}_0(a - b\mathbf{i}) \pmod{N\mathcal{O}_0}.\end{aligned}$$

Hence, Equation 3.2 is also solvable provided the ideals involved in both sides of the equation are the different fixed points. This case occurs with probability

$$\frac{2}{(N+1)^2}.$$

We conclude that the probability that Equation 3.2 is solvable is

$$\left(\frac{N-1}{N+1}\right)^2 + \frac{2}{(N+1)^2} = \frac{N^2 - 2N + 3}{(N+1)^2}.$$

□

Remark. Proposition 3.2 shows that Equation 3.2 is solvable with high probability. Indeed, even when N is split, the probability of being guaranteed a solution approaches 1 as $N \to \infty$.

Once such a β_2 has been found, we find an element β_2' such that $\beta_2' = \lambda\beta_2 \mod N\mathcal{O}_0$ and $\mathrm{nrd}(\beta_2') = S_2$ for some $\lambda \in (\mathbb{Z}/N\mathbb{Z})^*$. This is possible by tweaking the previous norm equation to accommodate the new information in the following way. We want this β_2' to be of the form

$$\beta_2' = v + w\mathbf{i} + x\mathbf{j} + y\mathbf{k}.$$

Since β_2' also needs to satisfy $\text{nrd}(\beta_2') = S_2$, we have to solve the following norm equation:

$$v^2 + w^2 + p(x^2 + y^2) = S_2. \tag{3.3}$$

Also, the condition that $\beta_2' = \lambda\beta_2 \pmod{N\mathcal{O}_0}$ is equivalent to stating that (after an appropriate change of basis):

$$\begin{aligned} v &= aN \\ w &= bN \\ x &= \lambda C + cN \\ y &= \lambda D + dN, \end{aligned}$$

for some a, b, c, $d \in \mathbb{Z}$. Substituting these values for v, w, x, and y in Equation 3.3 yields

$$N^2(a^2 + b^2) + p\left((\lambda C + cN)^2 + (\lambda D + dN)^2\right) = S_2. \tag{3.4}$$

To solve this equation for a, b, c, and d, we first consider Equation 3.4 modulo N to obtain the following:

$$p\lambda^2(C^2 + D^2) = S_2 \pmod{N},$$

and solve for λ, provided that $S_2/(p(C^2 + D^2))$ is a quadratic residue modulo N. If this is not the case, the issue is easily remedied by multiplying S_2 by small primes. Once the λ is found, we consider Equation 3.4 modulo N^2 which yields

$$p\lambda^2(C^2 + D^2) + 2p\lambda N(Cc + Dd) = S_2 \pmod{N^2}.$$

From this equation, we can pick a random d, and then solve for c (or vice versa). Rearranging Equation 3.3 gives

$$a^2 + b^2 = \frac{S_2 - p\left((\lambda C + cN)^2 + (\lambda D + dN)^2\right)}{N^2}$$

which we can solve for (a, b) using Cornacchia's algorithm. Note that due to our choice of λ, c, and d, the right hand side of this equation is an integer. Solving for (λ, a, b, c, d) yields the desired (v, w, x, y) by substitution.

Remark. The algorithm as described in [10] instead sets

$$\beta_2' = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$$

where a, b, c, d are directly obtained from Equation 3.4. We found that this neither gives β_2' with the correct norm nor satisfies $\beta_2' = \lambda\beta_2 \pmod{N\mathcal{O}_0}$, hence the need to substitute back to (v, w, x, y).

Computing the isogeny

The last step in the algorithm is computing the actual isogeny. Recall from the correspondence 3.1 that we need to find the kernel of the isogeny, that is, the set of points P such that $\alpha(P) = O$ for all $\alpha \in J$, our output ideal. We first need to clarify what we mean by $\alpha(P)$. There is an isomorphism of quaternion algebras

$$\begin{aligned} \theta : \quad B_{p,\infty} &\to \text{End}(E_0) \otimes \mathbb{Q} \\ (1, \mathbf{i}, \mathbf{j}, \mathbf{k}) &\mapsto ([1], \phi, \pi, \phi\pi) \end{aligned}$$

where $\phi : (x, y) \mapsto (-x, \iota y)$ is the "square root of -1" map, and $\pi : (x, y) \mapsto (x^p, y^p)$ is the Frobenius map. Given an element $\alpha \in J$, write $\alpha = a_1 + a_2\mathbf{i} + a_3\mathbf{j} + a_4\mathbf{k}$. Let $P(x, y)$ be a point. We then have:

$$\alpha(P) = [a_1]P + [a_2]\pi(P) + [a_3]\phi(P) + [a_4]\phi(\pi(P)). \tag{3.5}$$

The strategy described in [24] and used in [10] is to compute the elements of the kernel in $E_0[\ell_i^{e_i}]$ and compose them Chinese remainder theorem-style. To do so, since $E_0[\ell_i^{e_i}]$ is 2-dimensional, we look for two basis points P_i and Q_i. We then compute $\alpha(P_i)$ and $\alpha(Q_i)$ for every α in the basis of J. It is likely that such α contains coefficients with 2 in the denominator. This issue is dealt with by writing

$$\alpha = \frac{(\alpha'_1 + \alpha'_2\mathbf{i} + \alpha'_3\mathbf{j} + \alpha'_4\mathbf{k})}{2}$$

and performing *point division*: compute points P'_i and Q'_i such that $[2]P'_i = P_i$ and $[2]Q'_i = Q_i$, respectively. Although generally point division is not uniquely defined, it suffices to choose a point in this computation, since for any choice of P'_i (and respectively Q_i),

$$2\alpha(P'_i) = \alpha([2]P'_i) = \alpha(P_i).$$

Therefore, instead of computing as in Equation 3.5, we compute

$$\alpha(P_i) = [a'_1]P'_i + [a'_2]\pi(P'_i) + [a'_3]\phi(P'_i) + [a'_4]\phi(\pi(P'_i))$$

and

$$\alpha(Q_i) = [a'_1]Q'_i + [a'_2]\pi(Q'_i) + [a'_3]\phi(Q'_i) + [a'_4]\phi(\pi(Q'_i)).$$

Using all of these information, we compute a point R_i on $E_0[\ell_i^{e_i}]$ which satisfies $\alpha(R_i) = O$ for all $\alpha \in J$ using linear algebra. We then compute an isogeny with kernel generated by $\varphi_{i-1}(R_i)$, where $\varphi_0 = [1]_{E_0}$. We proceed through all i, constructing the isogeny step-by-step by composition, and at the end we have constructed an isogeny corresponding to the output ideal J.

3.3 A potential improvement

Having described the main features of the Kohel-Lauter-Petit-Tignol algorithm and given the details on most of the steps involved, we now describe a potential improvement to the algorithm. Recall that a step in the algorithm involved constructing an element β of norm NS, where S is a powersmooth number, such that $I'\bar{\beta} \subseteq N\mathcal{O}_0$. Since I' has norm N, we can write

$$I' = N\mathcal{O}_0 + \mathcal{O}_0\alpha$$

where $\alpha \in I'$ such that $\gcd(\mathrm{nrd}(\alpha), N^2) = N$. Therefore the requirement that $I'\bar{\beta} \subseteq N\mathcal{O}_0$ is equivalent to

$$(\mathcal{O}_0\alpha)\bar{\beta} = \mathbf{0} \pmod{N\mathcal{O}_0}$$

where $\mathbf{0}$ is the zero ideal. The equation of ideals is then equivalent to

$$\alpha\bar{\beta} = 0 \pmod{N\mathcal{O}_0}$$

which has $\beta = \alpha \pmod{N\mathcal{O}_0}$ as a solution. We can rewrite this solution as

$$\beta = \alpha + Nu + Nv\mathbf{i} + Nw\mathbf{j} + Nx\mathbf{k}$$

for some u, v, w, $x \in \frac{1}{2}\mathbb{Z}$. Since we insist that $\mathrm{nrd}(\beta) = NS$, this means

$$\mathrm{nrd}(\alpha + Nu + Nv\mathbf{i} + Nw\mathbf{j} + Nx\mathbf{k}) = NS$$

which yields

$$\mathrm{nrd}(\alpha) + \mathrm{trd}(\alpha)Nu - (\alpha - \bar{\alpha})N(v\mathbf{i} + w\mathbf{j} + x\mathbf{k}) + N^2(u^2 + v^2 + pw^2 + pk^2) = NS.$$

Since the right-hand side is an integer and $\beta \in I'$, the expression

$$(\alpha - \bar{\alpha})N(v\mathbf{i} + w\mathbf{j} + x\mathbf{k})$$

must also be an integer. Writing $\alpha = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ and simplifying the expression yields

$$2N(-bv - pcw - pdx + p(cx - dw)\mathbf{i} + (dv - bx)\mathbf{j}(bw - cv)\mathbf{k}).$$

Hence, the coefficients of $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ must be 0. This gives us a system of linear equations

$$\begin{cases} p(cx - dw) = 0 \\ dv - bx = 0 \\ bw - cv = 0 \end{cases}$$

which gives a family of solutions $(v, w, x) = \lambda(b, c, d)$ for some λ.

We speculate that this extra knowledge regarding β can be used to either improve or provide an alternative to finding the desired β in the Kohel-Lauter-Petit-Tignol algorithm. For instance, plugging the family of solutions back into the norm equation for $\mathrm{nrd}(\beta) = NS$ and completing the squares gives an equation of the form

$$a\lambda'^2 + bu'^2 = M$$

for which one can solve for integral λ' and u' using a generalized version of Cornacchia's algorithm given in [18].

Chapter 4

Implementation

We are now ready to describe the technical details of our implementation of the Kohel-Lauter-Petit-Tignol algorithm. We implemented the algorithm in Sage, which we choose due to its open source nature. This comes with the drawback that there are some functions that are not built-in or incomplete, which motivated most of the subroutines described in this chapter. At the end of this chapter, we discuss the limitations of our approach and some possible optimizations.

4.1 Implementation details

4.1.1 Enumerating powersmooth numbers

We first pre-compute powersmooth numbers S_1 and S_2 satisfying $S_1 > p \log p$ and $S_2 > p^3 \log p$ [10] where p is the prime used in the field of definition $\mathbb{F}_{p^2}$ of the curve. Let s be the powersmooth bound and let ℓ_i be the i-th odd prime. We first compute S_1 and S_2 recursively. For S_1:

1. Set $S_1 = \ell_1^{e_1}$, where $e_1 = \lfloor (\lfloor \log_{\ell_1} s \rfloor)/2 \rfloor)$ and set $i = 2$.
2. While $S_1 \leq p \log p$ and $e_i > 0$, replace S_1 by $S_1 \cdot \ell_i^{e_i}$ where
$$e_i = \left\lfloor \frac{\lfloor \log_{\ell_i} s \rfloor}{2} \right\rfloor.$$
Increment i.

Similarly, for S_2:

1. Set $S_2 = \ell_1^{e_1}$, where $e_1 = \lceil (\lfloor \log_{\ell_1} s \rfloor)/2 \rceil$ and set $i = 2$.
2. While $S_2 \leq p^3 \log p$ and $e_i > 0$, replace S_2 by $S_2 \cdot \ell_i^{e_i}$ where
$$e_i = \left\lceil \frac{\lfloor \log_{\ell_i} s \rfloor}{2} \right\rceil.$$
Increment i.

If, after encountering an i such that $e_i = 0$, the lower bound for either S_1 or S_2 is not yet satisfied, we generate a larger s-powersmooth number by successively multiplying the resulting number by small prime powers. For example, when the powersmoothness bound is set to 10, for each successive call to the generator it tries to multiply the previous number with numbers in the following order:
$$3, 3^2, 3^0 \cdot 5, 3 \cdot 5, 3^2 \cdot 5, 3^0 \cdot 5^0 \cdot 7, \ldots .$$

If the lower bound for either S_1 or S_2 is not yet reached after arriving at the largest possible s-powersmooth number, we simply increase s. In practice, we change s to $1.5s$.

For our implementation, we tweaked the S_1 and S_2 resulting from this computation to have as many common divisors as possible with $p + 1$.

4.1.2 Generating a random ideal

Step 1 of the algorithm requires a random ideal I as an input. We can construct this random ideal by constructing a random upper-triangular integer matrix $\mathbf{U}$. We then put the generators of $\mathcal{O}_0$ in a vector $\mathbf{b}$ and compute $\mathbf{x} = \mathbf{U}\mathbf{b}$. Finally, check whether $\mathbf{x}$ generates an ideal.

The determinant of $\mathbf{U}$ is necessarily a square.

Proposition 4.1. *Let* $\mathbf{U}$ *be a matrix and* $\mathbf{b}$ *a vector of generators of* $\mathcal{O}_0$*. If* $\mathbf{U}\mathbf{b}$ *generates an ideal, then* $\det(\mathbf{U})$ *is a square.*

Proof. Let I be the ideal generated by $\mathbf{U}\mathbf{b}$. Since $\mathcal{O}_0$ is a lattice, I is a sublattice. Moreover, since $\mathcal{O}_0$ is a maximal order, the left and right orders of I, $\mathcal{O}_L$ and $\mathcal{O}_R$, satisfy

$$\mathcal{O}_L(I) = \mathcal{O}_R(I) = \mathcal{O}_0$$

and I is invertible ([25], Proposition 16.1.2). Hence,

$$\mathrm{nrd}(I)^2 = [\mathcal{O}_L(I) : I] = [\mathcal{O}_0 : I].$$

From [25], Section 9.6.3, $[\mathcal{O}_0 : I]$ is the determinant of the change-of-basis matrix from $\mathcal{O}_0$ to I. The matrix $\mathbf{U}$ is precisely this change-of-basis matrix, and hence

$$\det(\mathbf{U}) = \mathrm{nrd}(I)^2.$$

□

The following corollary follows directly from Proposition 4.1.

Corollary 4.1. *If* $\mathbf{U}\mathbf{b}$ *generates an ideal* I*, then*

$$\mathrm{nrd}(I) = \sqrt{\det(\mathbf{U})}.$$

Using the previous statements, we can pick an ideal of small norm and construct an ideal of that norm using a matrix $\mathbf{U}$ of the correct determinant.

Now that we have established a necessary condition for $\mathbf{U}$, all that remains is to analyze the probability that $\mathbf{U}\mathbf{b}$ actually generates an ideal. From [12], we have that the number of different lattices of index n^2 is

$$\frac{(n^2)^4 - 1}{n^2 - 1} = (n^2 + 1)(n^4 + 1) \in O(n^6).$$

If we now restrict n to be a prime, by Lemma 3.2, the quotient ring $\mathcal{O}_0/n\mathcal{O}_0$ is a quaternion algebra. Moreover, it is not a division ring, since there exists a nonzero element with norm 0. Hence, from Lemma 3.3 $\mathcal{O}_0/n\mathcal{O}_0$ is isomorphic to the matrix ring $M_2(\mathbb{Z}/n\mathbb{Z})$. Since all ideals in $\mathcal{O}_0/n\mathcal{O}_0$ are principal [16], there exists a bijection χ from the set of $\mathcal{O}_0/n\mathcal{O}_0$-ideals to the set of $\mathcal{O}_0$-ideals of norm n given by

$$\chi((\mathcal{O}_0/n\mathcal{O}_0)\alpha) = n\mathcal{O}_0 + \mathcal{O}_0\alpha$$

where α is an element of the $\mathcal{O}_0$-ideal satisfying $\gcd(\mathrm{nrd}(\alpha), n^2) = n$. From Corollary 3.1, there are $n+1$ nontrivial ideals in $\mathcal{O}_0/n\mathcal{O}_0$. It follows that there are $n+1$ nontrivial $\mathcal{O}_0$-ideals of norm n. The probability of finding an ideal of norm n is

$$\frac{n+1}{(n^2+1)(n^4+1)} \approx \frac{1}{n^5}.$$

Therefore we expect this construction to run in at most $O(n^5)$ time.

The bijection χ can be used to construct ideals of norm n^2. Suppose I_1 and I_2 are two different ideals of prime norm n and

$$I_1 = n\mathcal{O}_0 + \mathcal{O}_0\alpha_1; \quad I_2 = n\mathcal{O}_0 + \mathcal{O}_0\alpha_2,$$

the set

$$I' = n^2\mathcal{O}_0 + \mathcal{O}_0(\alpha_1\alpha_2)$$

is an ideal of norm n^2. "Composing" the non-integral generator of the ideals of norm n in this manner allows us to construct ideals of norm n^e for some e.

4.1.3 Constructing an ideal of prime norm

Once the ideal I is constructed, we turn to step 1a of the algorithm, which is the computation of an ideal with a prime norm. Let $m = \lceil \log p \rceil$ and let $\{b_1, b_2, b_3, b_4\}$ be the generators of I. We perform an exhaustive search for a 4-tuple $(x_1, x_2, x_3, x_4) \in [-m, m]^4$ of integers until we find an element δ, where

$$\delta = x_1 b_1 + x_2 b_2 + x_3 b_3 + x_4 b_4$$

which satisfies that $N := \operatorname{nrd}(\delta)/\operatorname{nrd}(I)$ is a prime [10]. Once such an element δ is found, we construct the ideal $I' = I\bar{\delta}/\operatorname{nrd}(I)$. From Lemma 3.1, the I' is indeed a left $\mathcal{O}_0$-ideal of prime norm N.

4.1.4 Constructing an ideal of powersmooth norm

We now turn to step 1b of the algorithm, which involves the computation of an element β. We present two alternatives for this step.

Alternative 1: Direct computation of β

Our initial implementation attempt involves searching for an element β which satisfies

$$I'\bar{\beta} \subseteq N\mathcal{O}_0$$

and $\operatorname{nrd}(\beta) = NS$ where S is a powersmooth number using Cornacchia's algorithm. We note that this is equivalent to solving the following equation of ideals:

$$I'\bar{\beta} = \mathbf{0} \pmod{N\mathcal{O}_0} \tag{4.1}$$

where $\mathbf{0}$ is the zero ideal. While giving an argument for the feasibility of solving Equation 3.2, we established in Corollary 3.1 that there are $N+1$ nontrivial left $\mathcal{O}_0/N\mathcal{O}_0$-ideals. There are precisely 2 trivial ideals: the unit ideal and the zero ideal. Therefore, the probability that $I'\bar{\beta} = \mathbf{0} \pmod{N\mathcal{O}_0}$ is $1/(N+3)$ and therefore the probability of solving Equation 4.1 for β of the right norm is at most $1/(N+3)$. We can conclude that this simplified version of the algorithm runs in $O(N)$. We note that from [10], we can expect N to be of size $O(\sqrt{p})$, and hence this algorithm will run in exponential time for most SIDH parameters.

Alternative 2: Computing β_1 and β_2'

We now turn to the strategy described in [10]. Recall that this involves solving an equation of ideals

$$(\mathcal{O}_0\beta_1)\beta_2 = \mathcal{O}_0\alpha \pmod{N\mathcal{O}_0}$$

for $\beta_2 = C\mathbf{j} + D\mathbf{k}$. It is suggested in [15] that we use the isomorphism between $\mathcal{O}_0/N\mathcal{O}_0$-ideals and $M_2(\mathbb{Z}/N\mathbb{Z})$. Our approach differs than that of [15]. We use a more elementary approach. We observe that two principal left ideals are equal if their generators differ by a left multiplication with a unit; in our case, this is equivalent to solving

$$\beta_1\beta_2 = u\alpha \pmod{N\mathcal{O}_0}$$

or equivalently, the homogeneous equation

$$\beta_1\beta_2 - u\alpha = 0 \pmod{N\mathcal{O}_0}$$

for β_2 and u, where u is a unit in $\mathcal{O}_0/N\mathcal{O}_0$. Writing $u = u_1 + u_2\mathbf{i} + u_3\mathbf{j} + u_4\mathbf{k}$, $\beta_1 = b_1 + b_2\mathbf{i} + b_3\mathbf{j} + b_4\mathbf{k}$, and $\alpha = a_1 + a_2\mathbf{i} + a_3\mathbf{j} + a_4\mathbf{k}$, we have the following homogeneous system of equations modulo N:

$$\begin{bmatrix} -pb_3 & -pb_4 & -a_1 & a_2 & pa_3 & pa_4 \\ -pb_4 & pb_3 & -a_2 & -a_1 & -pa_4 & pa_3 \\ b_1 & -b_2 & -a_3 & a_4 & -a_1 & -a_2 \\ b_2 & b_1 & -a_4 & -a_3 & a_2 & -a_1 \end{bmatrix} \begin{bmatrix} C \\ D \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Since the requirement that β_2 and u are units is not represented in the linear system and the system is underdetermined, there needs to be a set of criteria to determine the validity of a solution. We claim that the solvability of the system depends on the dimension of the solution space and the nature of the solutions for β_2.

Proposition 4.2. *Let β_1 and α be the generators of the ideals $(\mathcal{O}_0\beta_1)$ and $(\mathcal{O}_0\alpha)$, respectively. Solving the equation of ideals*

$$(\mathcal{O}_0\beta_1)\beta_2 = (\mathcal{O}_0\alpha) \pmod{N\mathcal{O}_0}$$

for $\beta_2 = \mathbb{Z}\mathbf{j} + \mathbb{Z}\mathbf{k}$ is equivalent to solving the linear system of equations

$$\beta_1\beta_2 = u\alpha \pmod{N\mathcal{O}_0}.$$

for units β_2 and u. If the solution space of the system is a 4-dimensional $\mathbb{Z}/N\mathbb{Z}$-vector space, there is always a valid solution. If the solution space of the system is 3-dimensional, a family of valid solutions exist if and only if the nonzero solutions for β_2 are generated by a unit.

Proof. We first write all solutions to the system as $(\mathbf{w}|\mathbf{v})$, where $\mathbf{w}$ corresponds to (C, D) and $\mathbf{v}$ to the rest. The solution space of this system can be written as a direct sum of solutions of the following form:

$$(\mathbf{0}|\mathbf{v}') \oplus (\mathbf{w}'|\mathbf{v}'')$$

where $\mathbf{0}$ represents the zero solution for (C, D) and $\mathbf{w}'$ represents nonzero solutions for (C, D). We will compute the dimension of the solution space by computing the dimension of each term of the direct sum.

Computing the dimension of solutions of the form $(\mathbf{0}|\mathbf{v}')$ is equivalent to computing the nullity of the following matrix:

$$\begin{bmatrix} a_1 & -a_2 & -pa_3 & -pa_4 \\ a_2 & a_1 & pa_4 & -pa_3 \\ a_3 & -a_4 & a_1 & a_2 \\ a_4 & a_3 & -a_2 & a_1 \end{bmatrix}.$$

Assuming that a_1 and a_2 are not both zero, denote by $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$, and $\mathbf{c}_4$ the first, second, third, and fourth column of the matrix. It can be shown that the columns satisfy the following relations:

$$\begin{cases} \mathbf{c}_1 = \xi\mathbf{c}_3 + \eta\mathbf{c}_4 \\ \mathbf{c}_2 = -\eta\mathbf{c}_3 + \xi\mathbf{c}_4 \end{cases}$$

where $\xi = \dfrac{a_3a_1 - a_4a_2}{a_1^2 + a_2^2}$ and $\eta = \dfrac{a_3a_2 + a_1a_4}{a_1^2 + a_2^2}$. From the rank-nullity theorem, we conclude that the nullity of the matrix, and hence the dimension of solutions of the form $(\mathbf{0}|\mathbf{v}')$, is 2.

We now turn to the nonzero solutions. This immediately rules out that the solution space is of dimension 0. There are two cases to consider: the case where $\mathcal{O}_0\beta_1$ is a fixed point of the group action described in Lemma 3.4 and the case where it is not. We first consider the fixed point case. If this is the case, then either $\mathcal{O}_0\beta_1 = \mathcal{O}_0(a + b\mathbf{i})$ or $\mathcal{O}_0\beta_1 = \mathcal{O}_0(a - b\mathbf{i})$. From the result in Section 3.2 we established that if the ideals involved in both sides of the equation are the different fixed point ideals, any β_2 will be a solution of the system. Hence, the solution space in this case is 2-dimensional, bringing the total of the solution space dimension to 4. Otherwise, since the solution must be nonzero, the solution space must be 1-dimensional, bringing the total of the solution space dimension to 3.

It now remains to see what happens to the solution space when only one of the ideals involved the equation is a fixed point. The solution space for nonzero β_2 is one-dimensional, therefore it is enough to check whether the basis of this space is a unit. If the element β_2 were a unit, by definition of orbits, it will simply permute elements in the same orbit. Hence, to allow for solutions where the ideals involved are in different orbits, β_2 cannot be a unit. □

We note that by Proposition 3.2, Equation 3.2 is immediately solvable with high probability. Therefore, the complexity of this step is dominated by solving for $(C, D, u_1, u_2, u_3, u_4)$. This can be accomplished by algorithms such as the Gaussian elimination, which runs in $O(n^3)$ time where n is the number of variables.

4.1.5 Computing the isogeny

The implementation of the step 2, the computation of the isogeny, is rather straightforward. The first step in computing the isogeny is writing down the norm $\mathrm{nrd}(J)$ of the output ideal J as its prime factorization

$$\mathrm{nrd}(J) = \prod_i \ell_i^{e_i}.$$

Since $\mathrm{nrd}(J)$ is powersmooth, factorization will not be computationally expensive. In particular, if one chooses the powersmooth numbers as in Section 4.1.1, one already knows the factorization of $\mathrm{nrd}(J)$, which simplifies this step.

Next, we compute a basis for the $\ell_i^{e_i}$-torsion groups. We initially pick two random points P_i and Q_i in $E_0[\ell_i^{e_i}]$ with the correct order. We then check whether these two points form a basis for the $\ell_i^{e_i}$-torsion group according to the following proposition.

Proposition 4.3. *Let P and Q be points on E_0 of order ℓ^e. If P and Q do not span $E_0[\ell^e]$, then $[\ell^{e-1}]P$ and $[\ell^{e-1}]Q$ are dependent.*

Proof. Fix an isomorphism $E_0[\ell^e] \cong \mathbb{Z}/\ell^e\mathbb{Z} \times \mathbb{Z}/\ell^e\mathbb{Z}$. Let $\{P^*, Q^*\}$ be a basis for $E_0[\ell^e]$. Considering $E_0[\ell^e]$ as a $\mathbb{Z}/\ell^e\mathbb{Z}$-module, we assume that the points P and Q are given as vectors with respect to this basis. We write (m, n) for $[m]P^* + [n]Q^*$.

Let $P = (a, b)$ and $Q = (c, d)$. Suppose there is a point $R = (\lambda, \mu) \in E_0[\ell^e]$ which cannot be written as a linear combination of P^* and Q^*. The following system of linear equations modulo ℓ^e will have no solution:

$$\begin{cases} ax + by = \lambda \\ cx + dy = \mu. \end{cases}$$

which implies that the determinant of the matrix

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

is not a unit in $\mathbb{Z}/\ell^e\mathbb{Z}$. It follows that $ad - bc \equiv 0 \pmod{\ell}$. Write $ad = bc + k\ell$ for some $k \in \mathbb{Z}$.

Consider now the points $P' = [\ell^{e-1}]P = (\ell^{e-1}a, \ell^{e-1}b)$ and $Q' = [\ell^{e-1}]Q = (\ell^{e-1}c, \ell^{e-1}d)$. We have

$$[d]P' = (\ell^{e-1}ad, \ell^{e-1}bd)$$
$$[b]Q' = (\ell^{e-1}bc, \ell^{e-1}bd).$$

Since $ad = bc + k\ell$,

$$[d]P' = (\ell^{e-1}ad, \ell^{e-1}bd) = (\ell^{e-1}(bc + k\ell), \ell^{e-1}bd) = (\ell^{e-1}bc, \ell^{e-1}bd).$$

Hence, $[d]P' = [b]Q'$, and therefore $[\ell^{e-1}]P$ and $[\ell^{e-1}]Q$ are dependent. □

From this proposition, it follows that it is enough to check whether $[\ell_i^{e_i-1}]P_i$ and $[\ell_i^{e_i-1}]Q_i$ are independent by solving the discrete logarithm problem. The points $[\ell_i^{e_i-1}]P_i$ and $[\ell_i^{e_i-1}]Q_i$ are of order ℓ_i. Since the primes ℓ_i are small, it is feasible to solve the discrete logarithm problem.

It is possible that there does not exist a point of order $\ell_i^{e_i}$ in $\mathbb{F}_{p^2}$. This issue can be handled by temporarily lifting E_0 to a larger extension field containing the appropriate point and using the kernel polynomial $\psi(x)$ to define the isogeny over $\mathbb{F}_{p^2}$ [14].

Finally, we turn to the computation of the point R_i such that $\alpha(R_i) = O$ for every generator α of J. We do this by first writing $\alpha(P_i) = [A]P_i + [B]Q_i$ and $\alpha(Q_i) = [C]P_i + [D]Q_i$. The integers A, B, C, D are determined by solving a generalized discrete logarithm problem. Again, since ℓ_i is small and $\ell_i^{e_i}$ is powersmooth, it is feasible to solve the discrete logarithm problem using index calculus methods. We then construct the matrix

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

whose nullspace is the set of points $R_i' \in E_0[\ell_i^{e_i}]$ where $\alpha(R_i') = 0$. Once we have the nullspaces for each matrix corresponding to each generator α of J, we intersect the nullspaces and choose a point R_i of order $\ell_i^{e_i}$ in the intersection. Such a point R_i will be the generator of a generating set of the kernel of the output isogeny in $E_0[\ell_i^{e_i}]$ with which we perform the composition of isogenies as described in Section 3.2.

4.2 Performance

We now describe the results of our experiments with implementing the Kohel-Lauter-Petit-Tignol algorithm in Sage. For our experiments, we use input ideals of norm 3 and small primes $p = 431$ and $p = 1619$. A summary of the performance of our successful implementations is given in Figure 4.1. We discuss two main issues with our implementation: the choice for S_2 and the n-torsion points and the choice of prime p.

The first issue we encountered during implementation is the fact that choosing S_2 as described in Section 4.1.1 gives a rather abysmal success rate when solving the norm equation

$$a^2 + b^2 = \frac{S_2 - p\left((\lambda C + cN)^2 + (\lambda D + dN)^2\right)}{N^2}$$

despite satisfying the lower bound $p^3 \log p$ given in [10]. This is due to the fact that an inadequate choice for S_2 causes the right-hand side of the equation to be negative, which in turn renders the norm equation unsolvable. Thus, for our implementation, this always involves multiplying S_2 by small primes in increasing order, and in most cases, raising the powersmoothness bound. For example, our initial choice for S_2 when $p = 431$ is

$$16174233075 = 3 \times 5^2 \times 7 \times 11 \times 13 \times 17 \times 19 \times 23 \times 29$$

which has low success rate for solving the norm equation. Our final choice for S_2 in this case is

$$8948537162565 = 3 \times 5 \times 7 \times 11 \times 13 \times 19 \times 23 \times 29 \times 31 \times 37 \times 41$$

which has a more reasonable success rate, leading to successful results shown in Figure 4.1. The choice of adding the larger primes 31, 37, and 41 instead of increasing the power of the small primes (e.g. increasing 3 to 3^2) is due to the fact that in this case, increasing the power of the small primes leads to larger extension fields than the largest extensions shown in Figure 4.1. We

p	S_1	S_2	Largest extension	Running time of ideals step (sec.)	Running time of isogenies step (sec.)
431	4515	8948537162565	$GF(431^{84})$	0.47	443.11
431	4515	8948537162565	$GF(431^{84})$	0.45	407.32
431	4515	8948537162565	$GF(431^{84})$	0.43	460.69
1619	17017	621058354640325	$GF(1619^{84})$	0.48	718.34

Figure 4.1: Performance summary of our implementation of the Kohel-Petit-Lauter-Tignol algorithm.

suspect that this low success rate is due to our choice for solutions C and D when solving the ideal equation and the choice of c and d before computing the norm equation. The remedy to this situation might be choosing these parameters such that the sum-of-squares

$$(\lambda C + cN)^2 + (\lambda D + dN)^2$$

is minimized. Another possible solution to this situation is to construct a random input ideal of a larger norm, since the norm of the constructed ideal I' with prime norm is $N = \text{nrd}(\delta)/\text{nrd}(I)$, where I is the input ideal. Choosing a random input ideal of a larger norm might help minimize N. We also note that the lower bound in [10] is asymptotic, hence this problem might also be resolved by increasing p.

The second issue is the involvement of the n-torsion points. The total running time of the algorithm is directly influenced by the n-torsion points involved in the computation of the isogeny. This is due to the fact that the search for these points often requires changing the field of definition of the curve to a larger extension field. Therein lies the limitation imposed by the powersmooth assumption of this algorithm: it is likely that $\gcd(\text{nrd}(J)), p + 1)$ is small for some choices of p. In such cases, the running time of the isogeny computation part of the algorithm will be severely impacted due to the need to compute over large extension fields instead of simply over $\mathbb{F}_{p^2}$, as shown on Figure 4.1. Hence, it may be of interest to try to replace this condition, for instance using smooth norms instead of powersmooth ones, although we remark that this might not be easy due to the number of steps which require either factoring or solving the discrete logarithm problem. Another approach to resolving this issue would be to use primes such that $p + 1$ is powersmooth, and choose S_1 and S_2 such that $\gcd(S_1 S_2, p + 1)$ is as large as possible.

The Sage program used in this thesis is available at `https://github.com/dimitrijray/masters-thesis`. To run the program, download the program and run it directly using Sage. Readers using the SageMath VirtualBox should copy and paste the contents of the program into an empty jupyter notebook and simply choose “Run All”.

Chapter 5

Conclusions

We have described in detail the Kohel-Lauter-Petit-Tignol algorithm for constructing Deuring's correspondence in relation to the initial curve in SIDH and supplied extra explanation for some of the steps involved in the algorithm. We have also suggested that extra information to the nature of solutions β of the ideal equation $I\beta = \mathbf{0} \pmod{N\mathcal{O}_0}$ might be useful as a supplement to the original algorithm.

Other than describing the algorithm itself, we have also given our implementation of the algorithm in Sage. Some practical details need to be addressed due to the unavailability of required subroutines in Sage, which we have described. There are two main practical issues with the implementation. First, the powersmooth number S_2 often exceeds the initial powerswmoothness bound of $(7/2)\log p$. Second, the common factors of the norm of the output ideal from the first step of this algorithm might be small, which caused the running time of the isogeny computation step to suffer from the need to compute over large extension fields. This is due to the powersmooth assumptions of the algorithm.

5.1 Future work

For future research, we recommend to first optimize the choices made while solving the norm equation to find an element of norm S_2, such that the sum-of-squares described in Section 4.2 is minimized. Otherwise, one can also optimize and perhaps find a stricter condition for S_1 and S_2.

Another possibility for future research is to try and replace the powersmoothness assumption from this algorithm, for instance by smoothness. This might be difficult, since the powersmoothness assumption is required due to a lot of the steps in the isogeny construction requiring the use of the discrete logarithm problem. Therefore, one might be interested in looking for ways to construct the isogeny without having to resort to such methods.

Finally, one might also be interested in looking at the possible improvement given in Section 3.3 and investigate whether incorporating such information into the existing algorithm is possible.

Bibliography

[1] Julius Magalona Basilla. "On the solution of $x^2 + dy^2 = m$". In: *Proceedings of the Japan Academy, Series A, Mathematical Sciences* 80.5 (May 2004), pp. 40–41. DOI: 10.3792/pjaa.80.40. URL: https://doi.org/10.3792/pjaa.80.40.

[2] Reinier Bröker. "Constructing supersingular elliptic curves". In: *J. Comb. Number Theory* 1.3 (2009), pp. 269–273. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.304.7992&rep=rep1&type=pdf.

[3] Joe Buhler and Stan Wagon. "Basic algorithms in number theory". In: *Algorithmic Number Theory. Lattices, Number Fields, Curves and Cryptography* (2008), pp. 25–68. URL: http://www.math.leidenuniv.nl/~psh/ANTproc/02buhler.pdf.

[4] Ilya Chevyrev and Steven D. Galbraith. "Constructing supersingular elliptic curves with a given endomorphism ring". In: *LMS Journal of Computation and Mathematics* 17.A (2014), pp. 71–91. DOI: 10.1112/s1461157014000254. URL: https://doi.org/10.1112/s1461157014000254.

[5] Keith Conrad. *Quaternion algebras*. 2016. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.692.9806&rep=rep1&type=pdf.

[6] Craig Costello, Patrick Longa and Michael Naehrig. "Efficient Algorithms for Supersingular Isogeny Diffie-Hellman". In: *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*. 2016, pp. 572–601. DOI: 10.1007/978-3-662-53018-4_21. URL: https://doi.org/10.1007/978-3-662-53018-4%5C_21.

[7] Whitfield Diffie and Martin E. Hellman. "New directions in cryptography". In: *IEEE Trans. Information Theory* 22.6 (1976), pp. 644–654. DOI: 10.1109/TIT.1976.1055638. URL: https://doi.org/10.1109/TIT.1976.1055638.

[8] Kirsten Eisenträger, Sean Hallgren, Kristin E. Lauter, Travis Morrison and Christophe Petit. "Supersingular Isogeny Graphs and Endomorphism Rings: Reductions and Solutions". In: *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part III*. 2018, pp. 329–368. DOI: 10.1007/978-3-319-78372-7_11. URL: https://doi.org/10.1007/978-3-319-78372-7%5C_11.

[9] Steven D. Galbraith, Christophe Petit, Barak Shani and Yan Bo Ti. "On the Security of Supersingular Isogeny Cryptosystems". In: *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*. 2016, pp. 63–91. DOI: 10.1007/978-3-662-53887-6_3. URL: https://doi.org/10.1007/978-3-662-53887-6%5C_3.

[10] Steven D. Galbraith, Christophe Petit and Javier Silva. "Identification Protocols and Signature Schemes Based on Supersingular Isogeny Problems". In: *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*. 2017, pp. 3–33. DOI: 10.1007/978-3-319-70694-8_1. URL: https://doi.org/10.1007/978-3-319-70694-8%5C_1.

[11] Steven D. Galbraith and Frederik Vercauteren. "Computational problems in supersingular elliptic curve isogenies". In: *IACR Cryptology ePrint Archive* 2017 (2017), p. 774. URL: http://eprint.iacr.org/2017/774.

[12] Boris Gruber. "Alternative formulae for the number of sublattices". In: *Acta Crystallographica Section A: Foundations of Crystallography* 53.6 (1997), pp. 807–808.

[13] David Jao and Luca De Feo. "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies". In: *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, November 29 - December 2, 2011. Proceedings*. 2011, pp. 19–34. DOI: 10.1007/978-3-642-25405-5_2. URL: https://doi.org/10.1007/978-3-642-25405-5%5C_2.

[14] David Kohel. "Endomorphism rings of elliptic curves over finite fields". PhD thesis. University of California, Berkeley, 1996. URL: http://iml.univ-mrs.fr/~kohel/pub/thesis.pdf.

[15] David Kohel, Kristin Lauter, Christophe Petit and Jean-Pierre Tignol. "On the quaternion ℓ-isogeny path problem". In: *LMS Journal of Computation and Mathematics* 17.A (2014), pp. 418–432. URL: http://eprint.iacr.org/2014/505.

[16] Henryk Minc. "Left and Right Ideals in the Ring of 2×2 Matrices". In: *The American Mathematical Monthly* 71.1 (1964), pp. 72–75. ISSN: 00029890, 19300972. URL: http://www.jstor.org/stable/2311311.

[17] François Morain and Jean-Louis Nicolas. "On Cornacchia's algorithm for solving the diophantine equation $u^2+dv^2 = m$". In: *Projet* 1000 (1990). URL: https://pdfs.semanticscholar.org/f0b1/4badbba3534b994f3330edef9efb2c98f5e6.pdf.

[18] Abderrahmane Nitaj. "L'algorithme de Cornacchia". In: *Expositiones Mathematicae* 13 (1995), pp. 358–365. URL: https://nitaj.users.lmno.cnrs.fr/cornacchia.ps.

[19] Christophe Petit and Kristin E. Lauter. "Hard and Easy Problems for Supersingular Isogeny Graphs". In: *IACR Cryptology ePrint Archive* 2017 (2017), p. 962. URL: http://eprint.iacr.org/2017/962.

[20] John M. Pollard. "Monte Carlo Methods for Index Computation (mod p)". In: *Mathematics of Computation* 32.143 (1978), pp. 918–924. ISSN: 00255718, 10886842. URL: http://www.jstor.org/stable/2006496.

[21] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 8.2)*. 2018. URL: http://www.sagemath.org.

[22] Peter W. Shor. "Algorithms for quantum computation: Discrete logarithms and factoring". In: *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*. Ieee. 1994, pp. 124–134.

[23] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. Springer New York, 2009. DOI: 10.1007/978-0-387-09494-6. URL: https://doi.org/10.1007/978-0-387-09494-6.

[24] David Urbanik and David Jao. "SoK: The Problem Landscape of SIDH". In: *Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop, APKC@AsiaCCS, Incheon, Republic of Korea, June 4, 2018*. 2018, pp. 53–60. DOI: 10.1145/3197507.3197516. URL: http://doi.acm.org/10.1145/3197507.3197516.

[25] John Voight. *Quaternion algebras*. v.0.9.11. 2018. URL: https://math.dartmouth.edu/~jvoight/quat/quat-book-v0.9.11.pdf.