Karlsruhe Institute of Technology

Optimizations of Isogeny-based Key Exchange

KIT – Karlsruher Institut für Technologie

Paul Reichert
Bachelor's thesis

Submitted at July 13th, 2020

Responsible supervisor:	Prof. Dr. Jörn Müller-Quade
Supervisors:	Marcel Tiepelt
	Dr. Stefan Kühnlein

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die Arbeit selbständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des Karlsruher Instituts für Technologie zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung beachtet habe.

Abstract

Supersingular Isogeny Diffie-Hellman (SIDH) is a key exchange scheme that is believed to be quantum-resistant. It is based on the difficulty of finding a certain isogeny between given elliptic curves. Over the last nine years, optimizations have been proposed that significantly increased the performance of its implementations. Today, SIDH is a promising candidate in the US National Institute for Standards and Technology's (NIST's) post-quantum cryptography standardization process.

This work is a self-contained introduction to the active research on SIDH from a high-level, algorithmic lens. After an introduction to elliptic curves and SIDH itself, we describe the mathematical and algorithmic building blocks of the fastest known implementations.

Regarding elliptic curves, we describe which algorithms, data structures and trade-offs regarding elliptic curve arithmetic and isogeny computations exist and quantify their runtime cost in field operations. These findings are then tailored to the situation of SIDH. As a result, we give efficient algorithms for the performance-critical parts of the protocol.

Notation

Algebra and algebraic geometry

$\mathbb{N}$	The ring of natural numbers excluding zero
$\mathbb{Z}$	The ring of integers
$\mathbb{R}$	The field of real numbers
$\mathbb{C}$	The field of complex numbers
$\mathbb{F}_q$	The field with $q \in \mathbb{N}$ elements, if existent
$\overline{K}$	K being a field, the algebraic closure of K
$K[X_1, \ldots, X_n]$	The polynomial ring in $X_1, \ldots, X_n$
$K(X_1, \ldots, X_n)$	The fraction field of $K[X_1, \ldots, X_n]$
$\langle P_1, \ldots, P_k \rangle$	$P_1, \ldots, P_k$ being elements of a group, the subgroup they generate
$G[n]$	The n-torsion subgroup of an abelian group G
$[n]$	The multiplication-by-n map in an abelian group
C/K	A curve C defined over the field K
$E(L)$	E/K being an elliptic curve and $K \subseteq L \subseteq \overline{K}$, the subgroup of L-rational points

Supersingular Isogeny Diffie-Hellman protocol

For the exact way the parameters have to be chosen and which conditions they must fulfill, see Chapter 2.

General conventions

The fact that SIDH is symmetric with regard to the parties Alice and Bob is reflected in the notation as follows.

$\cdot_A$	Variables essentially relevant to Alice
$\cdot_B$	Variables essentially relevant to Bob
X	Either A or B, representing a party of the protocol
Y	Either A or B, representing the party which is complementary to X

Public parameters

ℓ_X, e_X	A small prime and a positive integer yielding a prime power ${\ell_X}^{e_X}$
n_X	Shorthand for ${\ell_X}^{e_X}$
p	A prime satisfying $p = n_A n_B f \pm 1$ for some $f \in \mathbb{N}$ coprime to $n_A n_B$
E_0	The base curve, a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$
P_X, Q_X	Basis points of $E_0[n_X]$, the torsion subgroup used for X's secret key.

Protocol

(a_X, b_X)	Secret of X: an element of order n_X of $\mathbb{Z}/n_X\mathbb{Z} \times \mathbb{Z}/n_X\mathbb{Z}$
R_X	The kernel point $[a_X]P_X + [b_X]Q_X$
N_X	The subgroup of E_0 generated by R_X
φ_X, E_X	The separable isogeny $E_0 \to E_X$ with kernel N_X and its codomain E_X
P'_X, Q'_X	The basis points $\varphi_Y(P_X), \varphi_Y(Q_X)$ of $E_Y[n_X]$
R'_X	The kernel point $\varphi_Y(R_X)$
ψ_X, E_{XY}	The separable isogeny $E_Y \to E_{XY}$ with kernel $\langle R'_X \rangle$ and its codomain E_{XY}

Contents

Preface

The world has never been more connected than now. People communicate with others of their kind, digital, over long distances, every day. The ubiquity of communication, often sensible, has led to a quick spread of cryptography into our everyday lives, even if many people rely on it without even noticing.

Cryptography has become one of the pillars of how we all exchange messages. These days, however, when the availability of highly scalable quantum computers seems to be only a matter of years, most of modern asymmetric cryptography has become vulnerable. Already today, in the absence of competitive quantum computers, it is vulerable, because the gist of cryptography is the promise that no one in the near future will be able to break it with a reasonable probability.

At the same time, as far as we can tell today, asymmetric cryptography is not lost. The US National Institute of Standards and Technology (NIST) has initiated a post-quantum cryptography standardization process and 17 candidates have been submitted to the second round of that process.

One of the submissions is Supersingular Isogeny Key Encapsulation (SIKE), at whose heart lies a key exchange protocol called Supersingular Isogeny Diffie-Hellman (SIDH). Its comparatively short key sizes make it helpful in certain situations. As a protocol, it is relatively slow, but the performance has been improved significantly in the last few years.

We start with the introduction of elliptic curves and relevant algebro-geometric concepts in their environment. Next, we briefly introduce SIDH and its underlying security assumptions. In the following chapters, we discuss algorithms that helped to obtain efficient implementations on three levels: fields, elliptic curves and the high-level protocol.

This work is hoped to be a useful introduction to the algorithmic optimizations behind modern SIDH implementations. It is supposed to enable the reader to understand, on a high level, how the optimizations work. Especially out of scope are low-level implementation and cryptanalysis

issues, except when they are relevant for the algorithm design.

While the thesis is intended to be self-contained, the reader is assumed to be comfortable with basic abstract algebra and to have an understanding of basic cryptographic concepts such as the concept of asymmetric cryptography.

Acknowledgements

First of all I would like to thank my supervisor Marcel Tiepelt from the faculty of computer science not only for coming up with this fruitful topic when I was desperately in search for one in the intersection of algebra and cryptography, but also for taking the time and advising me in the process of writing. Many parts of the thesis would look different, arguably less helpful, without his advice.

I also want to express my gratitude to Stefan Kühnlein, who was my co-supervisor from the faculty of mathematics. His encouraging attitude and exciting algebra lectures and the willingness to help unbureaucratically with many organizational issues regarding my parallel maths Bachelor have undoubtedly contributed to the fact that I'm now here finishing my Bachelor's degree.

Last but not least, I thank my parents, my fellow students and my mates from the students' dormitory for their support, whether emotional or material, during my studies. Having people that accept you as you are and that support you in hard times, even if it's about an unexpected and urgent 12 ares garden clearance, is really a privilege.

1 Elliptic curves

This chapter is about the mathematical preliminaries, namely the study of elliptic curves. Most stated facts have been taken from Silverman's book [Silo9].

1.1 The affine and projective plane

Let K be a field and denote its algebraic closure by $\overline{K}$ for the rest of the section.

We begin with some algebraic geometry: the definition of affine and projective spaces and their points.

1.1.1 The affine plane

Definition 1.1 (Affine plane) Let K be a field. The *affine plane*[1] over K is the set of 2-tuples

$$\mathbb{A}^2(K) \coloneqq \{P = (x_1, x_2) : x_1, x_2 \in K\}\,.$$

Its elements are called *(K-rational) points.* ◇

The affine plane $\mathbb{A}^2(K)$ is really just the set K^2, but it is equipped with a certain topology in algebraic geometry. The topology is not relevant for our purposes, though.

1.1.2 The projective plane

Consider the equivalence equation on $K^3 \setminus \{(0,0,0)\}$ given by

$$(x_1, x_2, x_3) \sim (\lambda x_1, \lambda x_2, \lambda x_3)$$

1 The affine plane over K is the affine 2-space over K, according to the definition of Silverman [Silo9, Section I.1].

for all $\lambda \in K \setminus \{0\}$. It identifies all points which are on the same line through the origin. The equivalence class of (x_1, x_2, x_3) is denoted by $[x_1 : x_2 : x_3]$.

Definition 1.2 (Projective plane) Let K be a field. The *projective plane*[2] over K is the set

$$\mathbb{P}^2(K) \coloneqq \left\{[x_1 : x_2 : x_3] : (x_1, x_2, x_3) \in K^3 \setminus \{(0,0,0)\}\right\}.$$

Its elements are called *(K-rational) points.* ◇

The affine plane $\mathbb{A}^2(K)$ is considered a subset of the projective plane $\mathbb{P}^2(K)$ by virtue of the inclusion sending (x_1, x_2) to $[x_1 : x_2 : 1]$. The remaining points are of the form $[x_1 : x_2 : 0]$ and are called *points at infinity.*

1.2 Algebraic curves

Let $f \in K[X, Y]$ be a polynomial. It can be evaluated at a point $(x, y) \in \mathbb{A}^2(\overline{K})$ in the obvious way. We are interested in the affine solutions $P \in \mathbb{A}^2(\overline{K})$ of

$$f(P) = 0.$$

The following definition is used to handle this geometric object.

Definition 1.3 (Affine algebraic curve) Let K be a field and $\overline{K}$ its algebraic closure. An *affine algebraic curve*[3] *C defined over K* is the zero set of an irreducible polynomial $f \in K[X, Y]$ in $\mathbb{A}^2(\overline{K})$. We write

$$C/K : f(x, y) = 0.$$

If L is a subfield of $\overline{K}$, the set of L-rational points is denoted by $C(L)$. ◇

Example 1.4 Let $f \coloneqq Y^2 - X \in \mathbb{Q}[X, Y]$. It defines (over $\mathbb{Q}$) an affine algebraic curve

$$C_a/\mathbb{Q} : y^2 = x.$$

2 The projective plane over K is the projective 2-space over K, according to the definition of [Silo9, Section I.2].

3 Silverman [Silo9, Section II.1] only defines projective algebraic curves. They are projective varieties of dimension one. Silverman's definition is equivalent to the one given below. The notion of an affine algebraic curve is obtained by removing the points at infinity.

Indeed, the fundamental theorem on homomorphisms implies that

$$\mathbb{Q}[X, Y]/(Y^2 - X) \cong \mathbb{Q}[Y].$$

Because $\mathbb{Q}[Y]$ is an integral domain, $Y^2 - X$ is a nonzero prime element in $\mathbb{Q}[X, Y]$. Because $\mathbb{Q}[X, Y]$ is an integral domain, in which every nonzero prime element is irreducible, we conclude that $Y^2 - X$ is irreducible.

There are $\mathbb{Q}$-rational points on C such as $(1, 1)$, but even though C is defined over $\mathbb{Q}$, the point $(2, \sqrt{2})$ lies on C and is not $\mathbb{Q}$-rational. ◇

It is relevant for the later study of elliptic curves to view algebraic curves as subsets of the *projective* plane. It seems plausible to define them again as zero sets of polynomials. However, since homogeneous coordinates are not unique, "zero sets" on the projective plane are not well-defined for all polynomials. For example, $XYZ - 1$ is zero at $(1, 1, 1)$ but not at $(2, 2, 2)$, although both represent the same projective point $[1 : 1 : 1]$.

There are, however, polynomials with a well-defined zero set.

Definition 1.5 (Homogeneous polynomial) Let K be a field and $\overline{K}$ its algebraic closure. A polynomial $f \in K[X, Y, Z]$ is *homogeneous*[4] *of degree d* if for all $\lambda \in \overline{K}$ and $x, y, z \in \overline{K}$,

$$f(\lambda x, \lambda y, \lambda z) = \lambda^d f(x, y, z).$$ ◇

Now we can define projective algebraic curves.

Definition 1.6 (Projective algebraic curve) Let K be a field and $\overline{K}$ its algebraic closure. A *projective algebraic curve*[5] C *defined over* K is the zero set of a homogeneous irreducible polynomial $f \in K[X, Y, Z]$ in $\mathbb{P}^2(\overline{K})$. We write

$$C/K : f(x, y, z) = 0.$$ ◇

Indeed, if $\lambda \neq 0$, the left and right hand side of the equation in Definition 1.5 are both zero or none of them is zero. Furthermore, setting $z = 1$ (thus, removing points at infinity), an affine

4 This is a special case of Silverman [Silo9, Chapter I.2].

5 This definition is equivalent to the one given by Silverman [Silo9, Section II.1] in the case of the projective plane. Note that the dimension of a projective variety (I.2) equals the dimension of the underlying affine curve as an affine variety (I.1), which in turn is one if and only if it can be defined by a single irreducible polynomial.

algebraic curve remains.

Conversely, an *affine* algebraic curve gives rise to a *projective* one by *homogenization* of the polynomial $f \in K[X, Y]$. Homogenization is the operation of multiplying all monomial summands of f with powers of Z until they all have the same degree. For example, the homogenization of $XY + 2X + 3$ is $XY + 2XZ + 3Z^2$.

Example 1.7 The affine algebraic curve from Example 1.4

$$C_a/\mathbb{Q} : y^2 = x$$

gives rise to the projective algebraic curve

$$C_p/\mathbb{Q} : y^2 = xz.$$

We know that $Y^2 - X$ is irreducible and it follows that $Y^2 - XZ$ is: Assume for contradiction that $f := Y^2 - XZ = g \cdot h$ for two non-units $g, h \in \mathbb{Q}[X, Y, Z]$ of degree 1. Both factors are homogeneous because f is. Consider the dehomogenization map

$$H^{-1} \colon \mathbb{Q}[X, Y, Z] \to \mathbb{Q}[X, Y], \quad X \mapsto X, \quad Y \mapsto Y, \quad Z \mapsto 1.$$

Applying H^{-1}, we obtain a factorization of the irreducible polynomial

$$Y^2 - X = H^{-1}(f) = H^{-1}(g) \cdot H^{-1}(h).$$

Irreducibility of the left hand side implies that, without loss of generality, $H^{-1}(g) = 1$. Because g is homogeneous and has degree 1, it follows that $g = Z$. However, $g = Z$ does not divide $f = Y^2 - XZ$. This is a contradiction, so $Y^2 - XZ$ must be irreducible. ◇

When talking about a *curve*, a projective algebraic curve is meant. For the sake of simplicity, projective algebraic curves are also sometimes specified by the corresponding affine equation.

1.3 Morphisms

Definition 1.8 (Rational map on the projective plane) Let $K \subseteq L \subseteq \overline{K}$ a tower of fields. If f_x, f_y and f_z are same-degree homogeneous polynomials from $L[X, Y, Z]$, denote by $\varphi = \left[f_x : f_y : f_z\right]$ the partial function from $\mathbb{P}^2(K)$ to $\mathbb{P}^2(K)$ defined as follows.

The function φ is *regular* at a point $P = [x : y : z] \in \mathbb{P}^2(K)$ if and only if there is a $\lambda \in K(X, Y, Z)$ such that $g_x = \lambda f_x$, $g_y = \lambda f_y$ and $g_z = \lambda f_z$ are same-degree homogeneous polynomials and $\varphi(P) \coloneqq \big[g_x(x, y, z) : g_y(x, y, z) : g_z(x, y, z)\big]$ is specified by well-formed homogeneous coordinates. The points of definition of φ are its regular points.

We call $\big[f_x : f_y : f_z\big]$ a *rational map* on $\mathbb{P}^2(K)$ that is *defined over* L. ◇

Definition 1.9 (Rational map of curves) Let K be a field and C_1 and C_2 curves defined over K. A *rational map defined over* L from C_1 to C_2 is a partial function from C_1 to C_2 that is the restriction $\big[f_x : f_y : f_z\big]\,|_{C_1}$ of a rational map defined over L on $\mathbb{P}^2(K)$ with at least one regular point in C_1. A *morphism* from C_1 to C_2 is a rational function from C_1 to C_2 that is regular everywhere.

A rational map is considered to be defined over $\overline{K}$ if L is not given explicitly. ◇

1.4 Elliptic curves

In the rest of the thesis we are only interested in elliptic curves over a field with characteristic $p = 0$ or $p > 3$. Such fields will be called *admissible*. The zero-characteristic case is only relevant for examples and visualizations.

1.4.1 Definition

Definition 1.10 (Elliptic curve) Let K be an admissible field. An *elliptic curve*[6] E defined over K is a curve parameterized by a degree-three polynomial $f \in K[X]$ with three distinct roots in $\overline{K}$ with equation

$$E/K\colon y^2 = f(x).$$

◇

Example 1.11 The elliptic curve $E/\mathbb{R}\colon y^2 = x^3 - x + 1$ is depicted in Fig. 1.1.

An elliptic curve has exactly one point at infinity, which is denoted by $O \in C$.

Elliptic curves are said to be *isomorphic* if there is an isomorphism *defined over* $\overline{K}$, i. e. a morphism with an inverse morphism, between them. Elliptic curves that are isomorphic

6 This is a special case of the definition given by Silverman [Silo9, Section III.3]. The requirement that f has three distinct roots is equivalent to the requirement that E is smooth. Indeed, the singular points (x, y) of E are exactly those with $f(x) = f'(x) = 0$, hence, with x being a zero of f of multiplicity two or higher.

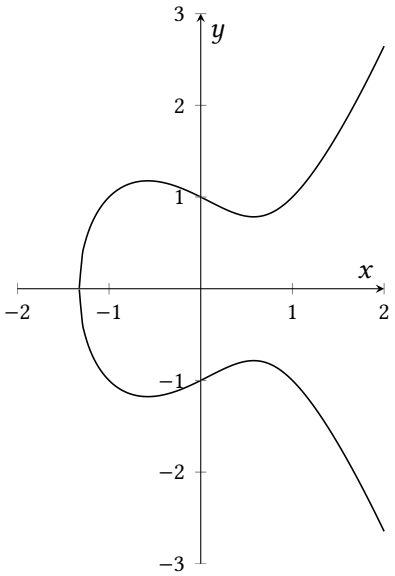


Figure 1.1: The elliptic curve $E/\mathbb{R}\colon y^2 = x^3 - x + 1$

are said to belong to the same isomorphism class. There is a quantity that characterizes isomorphism classes: According to Silverman [Silo9, Proposition III.1.4b], elliptic curves are isomorphic if and only if they have the same j-invariant.

Definition 1.12 (j-invariant) Let K be an admissible field and E the elliptic curve

$$E/K\colon y^2 = c\left(x^3 + a_2x^2 + a_4x + a_6\right).$$

The *j-invariant*[7] of E is

$$j(E) \coloneqq 16\left(-4a_4^3 - 27a_6^2 + 18a_2a_4a_6 - 4a_2^3a_6 + a_2^2a_4^2\right). \qquad \diamond$$

1.4.2 Group structure

What makes elliptic curves interesting for cryptography is that they admit a group structure with neutral element O. The group operation, as derived by Silverman [Silo9, Algorithm III.2.3], is deeply connected to the algebraic structure of the curve.

7 Compare to the definition of the j-invariant by Silverman [Silo9, Section III.1]. The formula is obtained after the linear coordinate change $\tilde{y} = \sqrt{c}^{-1}y$ to get rid of the c.

Theorem 1.13 (Group law) Let E be an elliptic curve given by

$$E\colon y^2 = c\left(x^3 + a_2x^2 + a_4x + a_6\right),$$

and suppose that $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ are points on E. Then:

(a) O is the neutral element of E.

(b) The inverse of P_1 is $-P_1 = (x_1, -y_1)$. In particular, if $x_1 = x_2$ and $y_1 = -y_2$, then $P_1 + P_2 = O$.

(c) To obtain an explicit formula for the sum $P + Q$ if P_1 is not inverse to P_2, we distinguish two cases.

(i) If $P = Q$, the tangent at P is $L\colon y = \lambda x + \nu$ with

$$\lambda = c\frac{3x_1^2 + 2a_2x_1 + a_4}{2y_1}, \quad \nu = c\frac{-x_1^3 + a_4x_1 + 2a_6}{2y_1}.$$

(ii) If $P \neq \pm Q$, the secant at P is $L\colon y = \lambda x + \nu$ with

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}, \quad \nu = \frac{y_1x_2 - y_2x_1}{x_2 - x_1}.$$

In both cases, $P + Q = (x_3, y_3)$ is the inverse of the third intersection (counted with multiplicities) of L and E and given by

$$x_3 = \left(c^{-1}\lambda^2 - a_2\right) - x_1 - x_2, \quad y_3 = -\left(\lambda x_3 + \nu\right).$$

These formulas make $(E, +)$ an abelian group. ◇

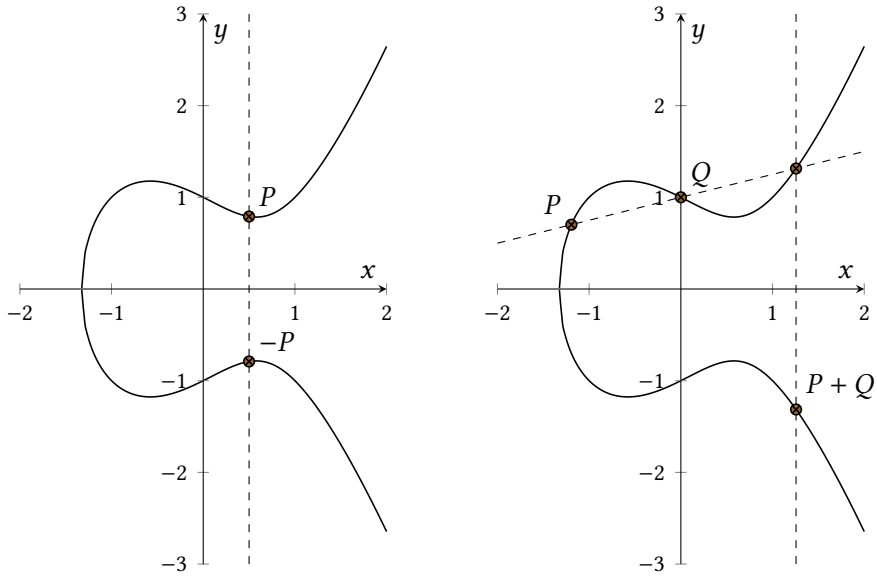
Proof Silverman [Silo9, Algorithm II.2.3] gives the proof for the case $c = 1$. Because isomorphisms of elliptic curves preserving O preserve the group law, the general case is recovered using the coordinate transformation $\tilde{y} = \sqrt{c}^{-1}y$ and the group law on the curve

$$\tilde{E}\colon \tilde{y}^2 = x^3 + a_2x^2 + a_4x + a_6.$$

■

The geometric interpretation of the group law is shown in Fig. 1.2.

Remark 1.14 The theorem above implies that if $K \subseteq L \subseteq \overline{K}$ is a tower of fields and E is an



(a) Construction of $-P$

(b) Construction of $P + Q$

Figure 1.2: The group law on the curve $C/\mathbb{R}\colon y^2 = x^3 - x + 1$ has a geometric interpretation. All points of intersection of a straight line with C always sum up to O. Note that points of intersection are counted with multiplicities.

elliptic curve defined over K, the set of L-rational points $E(L)$ is a subgroup of E. ◇

1.4.3 Special curves

This section presents special types of curves that are of particular interest for applied cryptography.

Short Weierstraß curves

Definition 1.15 (Short Weierstraß curve) A *short Weierstraß curve*[8] defined over an admissible field K is an elliptic curve $W_{a,b}$ that is given by the equation

$$W_{a,b}\colon y^2 = x^3 + ax + b. \tag{1.1}$$

For the sake of the right-hand side having distict roots, a and b must satisfy $4a^3 + 27b^2 \neq 0$. ◇

Fig. 1.1 is an example of a short Weierstraß curve.

Montgomery curves

Montgomery curves are special elliptic curves that have been introduced by Montgomery [Mon87] in 1987.

Definition 1.16 (Montgomery curve) A *Montgomery curve* defined over an admissible field K, with parameters A and B from K, is an elliptic curve $E_{A,B}$ that is given by the equation

$$M_{A,B}\colon By^2 = x^3 + Ax^2 + x. \tag{1.2}$$

The parameters are required satisfy $B(A-2)^2 \neq 0$. ◇

Twisted Edwards curves

Twisted Edwards curves have been introduced by Bernstein et al. [Ber+08] in 2008. They are *not* elliptic.

8 Compare to Silverman [Silo9, Remark III.1.3].

Definition 1.17 (Twisted Edwards curve) A *twisted Edwards curve* defined over a admissible field K with parameters a and d from K is a curve $E_{a,d}$ given by the equation

$$E_{a,d}\colon ax^2 + y^2 = 1 + dx^2y^2.$$

The parameters must fulfil $ad\,(a-d) \neq 0$. ◇

Relations between short Weierstraß, twisted Edwards and Montgomery curves

Every elliptic curve defined over K is isomorphic over K to a short Weierstraß curve defined over K. While twisted Edwards curves are not elliptic, they are close to an elliptic curve. Consider a twisted Edwards curve $E_{a,d}$. Then with

$$A \coloneqq 2\,(a+d)\,(a-d)\,, \quad B \coloneqq 4/(a-d)\,,$$

$M_{A,B}$ is a Montgomery curve. Bernstein et al. [Ber+08] found two surjective rational maps

$$\Phi\colon M_{A,B} \to E_{a,d}, \quad [u:v:w] \mapsto [u\,(u+w) : v\,(u-w) : v\,(u+w)]$$

and

$$\Psi\colon E_{a,d} \to M_{A,B}, \quad [x:y:z] \mapsto [x\,(z+y) : z\,(z+y) : x\,(z-y)]$$

that are inverses as far as they are defined. Such a pair of mutually inverse surjective rational maps is called a *birational equivalence* of curves. In fact, Φ is even a morphism.[9] The only points of nonregularity of Ψ are the points at infinity $[1:0:0]$ and $[0:1:0]$ so that Ψ is an injective map from the affine twisted Edwards curve into the projective Montgomery curve. Its image misses out not more than four points of order 2 and 4 of $M_{A,B}$.

1.4.4 Isogenies

Under which circumstances is a morphism of elliptic curves $\varphi\colon E_1 \to E_2$ a group homomorphism? A necessary condition is that φ preserves the point at infinity because it is the neutral element. This condition turns out to be sufficient (for a proof refer to Silverman [Silo9, Theorem III.4.8]) and it motivates the following definition.

9 The reason why Φ is a morphism is that $M_{A,B}$ is an elliptic curve and as such smooth.

Definition 1.18 (Isogeny) Let K be an admissible field and E_1 and E_2 elliptic curves over K. An *isogeny*[10] from E_1 to E_2 is a morphism

$$\varphi\colon E_1 \longrightarrow E_2 \quad \text{satisfying} \quad \varphi(O) = O. \qquad \diamond$$

Nonzero isogenies are always surjective (Silverman [Silo9, Section III.4]). Their kernels are always finite (Silverman [Silo9, Corollary III.4.9]).

1.4.5 Isogenies over positive characteristic

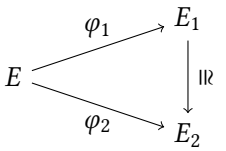
Remark and definition 1.19 (Frobenius map) Let K be an admissible field of characteristic $p > 3$ and $E/K\colon y^2 = c\left(x^3 + a_2x^2 + a_4x + a_6\right)$ an elliptic curve. Then $E^{(p)}/K\colon y^2 = c^p\left(x^3 + a_2^px^2 + a_4^px + a_6^p\right)$ is also an elliptic curve and the *Frobenius*[11] map

$$\tau\colon E \longrightarrow E^{(p)}, \quad [x : y : z] \mapsto \left[x^p : y^p : z^p\right]$$

is an isogeny. $\diamond$

The Frobenius map τ is a group isomorphism, so for every isogeny φ with kernel N, we obtain more isogenies with the same kernel, namely $\tau^e \circ \varphi$. Isogenies that cannot be obtained by composing τ with another isogeny are called *separable*. This is a characterization proven by Silverman [Silo9, Corollary II.2.12].

Given a finite subgroup N of an elliptic curve E, there is at least one separable isogeny $E \longrightarrow E_1$ with kernel N, and if there is another one $E \longrightarrow E_2$, then there must be an isomorphism $E_1 \cong E_2$ that makes the following diagram commute (Silverman [Silo9, Corollary III.4.12]).



The codomain (E_1 or E_2 above) is often denoted by E/N. The notation is motivated by the fact that the codomain is isomorphic to the quotient group E/N by virtue of the fundamental theorem on group homomorphisms.

10 Compare to Silverman [Silo9, Section III.4].
11 Compare to Silverman [Silo9, Exapmle III.4.6].

A separable isogeny is said to have *degree* $k \in \mathbb{N}$ if the order of its kernel is k.

We also need the following result.

Theorem 1.20 (Order of isogenous curves) Let E_1, E_2 be elliptic curves defined over $\mathbb{F}_{p^2}$, a field over characteristic $p > 3$. If there is an isogeny $E_1 \to E_2$ defined over $\mathbb{F}_{p^2}$, then E_1 and E_2 have the same number of $\mathbb{F}_{p^2}$-rational points. ◇

Proof The theorem is due to Tate [Tat66, Theorem 1]. ∎

1.4.6 Vélu's formulas

Let K be a finite field and E an elliptic curve over K of the form

$$E : y^2 = c\left(x^3 + a_2x^2 + a_4x + a_6\right),$$

for example, a short Weierstraß or Montgomery curve, and G a finite subgroup. The separable isogeny $\varphi\colon E \to E'$ with kernel G is given explicitly by Vélu's [Vél71] formulas.[12] For points $P \in E$ outside G, the image is given by

$$x(\varphi(P)) = x(P) + \sum_{Q \in G\setminus\{O\}} (x(P+Q) - x(Q))$$

and

$$y(\varphi(P)) = y(P) + \sum_{Q \in G\setminus\{O\}} (y(P+Q) - y(Q)).$$

The image curve E' is

$$E'\colon y^2 = c\left(x^3 + a_2x^2 + a_4'x + a_6'\right) \quad \text{for some } a_4' \text{ and } a_6' \text{ in } \overline{K}.$$

In particular, the coefficient a_2 is preserved. So if E is a short Weierstraß curve, then E' is also a short Weierstraß curve. However, E' need not be a Montgomery curve even if E is.

1.4.7 Torsion subgroups and supersingularity

Our interest in finite subgroups will lead us to the study of torsion points and torsion groups.

12 Vélu only proved the case $c = 1$ but the general case easily follows from pre- and postcomposing linear transformations that scale the y-coordinate.

Definition 1.21 (Torsion point, torsion group) Let E be an elliptic curve. A *torsion point* on E is a point P on E of finite order. More specifically, P is an *m-torsion point* if $[m]P = \mathcal{O}$.

The *m-torsion group* of E, denoted by $E[m]$, is the subgroup of E that consists of all m-torsion points. ◇

Remark 1.22 The m-torsion subgroup is the kernel of the multiplication-by-m isogeny $[m]$.◇

Let K be a field of prime characteristic $p > 3$ and E an elliptic curve over K. Silverman [Silo9, Theorem V.3.1] gives a description of E's torsion subgroups (up to isomorphism).

If $m \in \mathbb{N}$ is not divisible by p,

$$E[m] \cong \frac{\mathbb{Z}}{m\mathbb{Z}} \times \frac{\mathbb{Z}}{m\mathbb{Z}}. \tag{1.3}$$

However, there are two possibilities for powers of p. One of the following equations holds:

$$E[p^e] \cong 0, \quad \text{for } \textit{all } e \in \mathbb{N}, \tag{1.4}$$

or

$$E[p^e] \cong \frac{\mathbb{Z}}{p^e\mathbb{Z}}, \quad \text{for } \textit{all } e \in \mathbb{N}. \tag{1.5}$$

Although we do not need this fact, the structure of a torsion subgroup that does not match any of the above cases can be obtained using the fundamental theorem of finitely generated abelian groups.

Definition 1.23 (Ordinary and supersingular) Those elliptic curves that satisfy Eq. (1.5) are called *ordinary*. If they satisfy (1.4) instead, they are called *supersingular*. ◇

Supersingular curves are fundamental for the cryptography in the following chapters. They are always isomorphic to an elliptic curve defined over $\mathbb{F}_{p^2}$ (Silverman [Silo9, p. V.3.1]).

1.4.8 Special supersingular curves

The following result is important for the efficient representation of torsion points on the elliptic curves that are relevant to SIDH.

Theorem 1.24 (Torsion points are $\mathbb{F}_{p^2}$-rational) Let p be a prime $p > 3$, E a supersingular elliptic curve and $\left|E(\mathbb{F}_{p^2})\right| = (p \pm 1)^2$. Then

$$E[p \pm 1] = E(\mathbb{F}_{p^2}). \qquad \diamond$$

Proof Let $k := p \pm 1$ and $t := p^2 + 1 - \left|E(\mathbb{F}_{p^2})\right| = \mp 2p$. Then $t^2 = 4p^2$.

We already know from Remark 1.14 that $E(\mathbb{F}_{p^2})$ is a subgroup of E. According to Schoof [Sch87, Lemma 4.8], we obtain

$$E(\mathbb{F}_{p^2}) \cong \frac{\mathbb{Z}}{k\mathbb{Z}} \times \frac{\mathbb{Z}}{k\mathbb{Z}}. \tag{1.6}$$

This implies that $E(\mathbb{F}_{p^2})$ is a subset of $E[k]$. On the other hand, Eqs. (1.3) and (1.6) imply that both groups have exactly k^2 elements. It follows that they are equal. ∎

2 Supersingular isogeny cryptography

The prospect of large quantum computers in the near future motivated cryptographers to develop cryptographic primitives that are conjectured not to be efficiently breakable, even with quantum computers.

2.1 Supersingular Isogeny Diffie-Hellman

Supersingular Isogeny Diffie-Hellman (SIDH) key exchange can be considered as a modification of the well-known Diffie-Hellman key exchange. In the same way as the latter gives rise to ElGamal public-key encryption, SIDH can be used to construct a public-key encryption scheme whose security can be reduced to a well-defined, though less well-understood, mathematical hardness assumption analogous to the Decisional Diffie Hellman (DDH) and Computational Diffie Hellman (CDH) problems.

2.1.1 History of isogeny-based cryptography

Probably the first application of elliptic curves and their group structure in the context of public-key cryptography was their use in classical schemes such as El-Gamal or Diffie-Hellman. Their use was motivated by the observation that the discrete logarithm problem was hard to solve in some ordinary elliptic curves using a classical computing model. The same problem, however, is easy to solve in a quantum computing model.

A cryptographic scheme proposed by Couveignes [Cou06] and independently rediscovered by Rostovtsev and Stolbunov [RS06] seemed to depend on a harder problem: simply said, finding an isogeny between two given ordinary elliptic curves. But Childs, Jao, and Soukharev [CJS10] showed that the scheme of Couveignes, that relied fundamentally on the commutativity of the group action of the ideal class group of the endomorphism ring on the curve, turned out to be vulnerable to a subexponential quantum attack using a quantum algorithm for the so-called

abelian hidden shift problem.

It is not the case that this attack forbids the use of the Couveignes-Rostovtsev-Stolbunov scheme in practice; as Castryck et al. [Cas+18] have pointed out, RSA is still considered secure notwithstanding the fact that there is a subexponential attack on the factorization problem, and the more recent Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) scheme of Castryck et al. [Cas+18], that was designed for a better performance instead of mitigating the hidden shift attack, still relies on a commutative group action.

Another branch of research was opened up earlier by Feo, Jao, and Plût [FJP11] when they proposed SIDH, that is fundamentally different from the Couveignes-Rostovtsev-Stolbunov scheme and uses supersingular elliptic curves to avoid the commutativity of the ideal class group. This is the protocol that this thesis is concerned with.

2.1.2 SIDH protocol

Feo, Jao, and Plût [FJP11] proposed SIDH in 2011. Assume Alice and Bob want to exchange a key using SIDH. They proceed as follows(see Fig. 2.1):

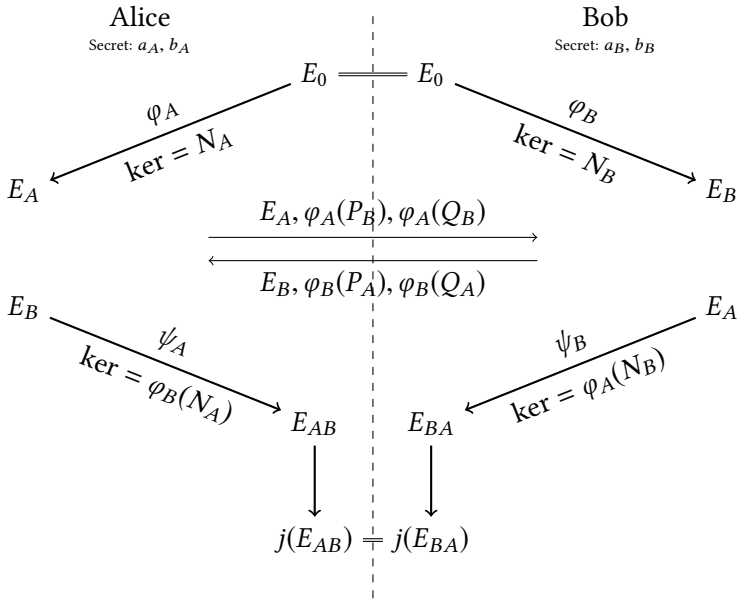


Figure 2.1: Visualization of an SIDH key exchange between Alice and Bob

1. Alice and Bob publicly agree on parameters: small distinct primes ℓ_A, ℓ_B, powers $n_A =$

$\ell_A{}^{e_A}$ and $n_B = \ell_B{}^{e_B}$, a prime $p = n_A n_B f \pm 1$ where f is coprime to ℓ_A and ℓ_B, a supersingular curve E_0 over $\mathbb{F}_{p^2}$ of cardinality $(\ell_A{}^{e_A}\ell_B{}^{e_B} f)^2$ and bases $(P_A, Q_A), (P_B, Q_B)$ of the free $\mathbb{Z}/n_A\mathbb{Z}$- or $\mathbb{Z}/n_B\mathbb{Z}$-modules $E_0[n_A]$ and $E_0[n_B]$. The parameters should satisfy $n_A \approx n_B$.

2. Using the base of $E_0[n_A]$, Alice chooses two random secret integer coefficients a_A, b_A at random such that $R_A = [a_A]P_A + [b_A]Q_A$ has order n_A and computes a separable isogeny defined over $\mathbb{F}_{p^2}$, namely $\varphi_A \colon E_0 \to E_A$, with kernel $N_A = \langle R_A \rangle$. Then she sends E_A and $P'_B = \varphi_A(P_B)$, $Q'_B = \varphi_A(Q_B)$ to Bob. Bob proceeds analogously.

3. Having received E_B, P'_A and Q'_A, Alice computes $R'_A = \varphi_B(R_A)$ using the received data and her secret a_A, b_A. Then she computes a separable isogeny defined over $\mathbb{F}_{p^2}$, namely $\psi_A \colon E_B \to E_{AB}$. The j-invariant of E_{AB} is the shared secret. Bob proceeds analogously and finally computes E_{BA} and its j-invariant.

Indeed, E_{BA} and E_{AB} are isomorphic since both are the codomains of a separable isogeny with kernel $\langle N_A, N_B \rangle$, so that Alice and Bob compute the same shared secret.

Remark 2.1 By construction and Theorem 1.20, all curves in the isogeny class of E_0 have order $|E_0|$ and fulfil the requirements of Theorem 1.24. Thus, all points that occur in the protocol are $\mathbb{F}_{p^2}$-rational and can be easily represented as tuples of $\mathbb{F}_{p^2}$-valued coordinates. ◇

2.1.3 Semantical security of SIDH and the derived public-key cryptosystem

The central security assumption regarding SIDH are the Supersingular Computational Diffie-Hellman (SSCDH) and the Supersingular Decision Diffie-Hellman (SSDDH) problem, introduced by Feo, Jao, and Plût [FJP11]. Before stating them, it is customary to start with a definition.

Definition 2.2 Let $X \in \{A, B\}$. Denote by $\mathcal{K}_X$ the distribution of (a_X, b_X), both chosen at random from $\mathbb{Z}/\ell_X{}^{e_X}\mathbb{Z}$ such that at least one is coprime to ℓ_X. Furthermore, denote by $\iota_X(a_X, b_X)$ a separable isogeny whose kernel is $\langle [a_X]P_X + [b_X]Q_X \rangle$, and by $j_{AB}(a_A, b_A, a_B, b_B)$ the j-invariant of

$$E_0/\langle [a_A]P_A + [b_A]Q_A, [a_B]P_B + [b_B]P_B \rangle\,.$$ ◇

Problem 2.3 (Supersingular Computational Diffie-Hellman (SSCDH)) For any choice of $X = A, B$, let $(a_X, b_X) \leftarrow \mathcal{K}_X$, and $\varphi_X \leftarrow (\iota_X(a_X, b_X) \colon E_0 \to E_X)$.

Given the curves E_A, E_B and the points $\varphi_A(P_B)$, $\varphi_A(Q_B)$, $\varphi_B(P_A)$ and $\varphi_B(P_B)$, compute the j-invariant $j_{AB}(a_A, b_B, a_B, b_B)$. ◇

Problem 2.4 (Supersingular Decision Diffie-Hellman (SSDDH)) For any $i = 1, 2$ and for $X = A, B$, let $(a_{X,i}, b_{X,i}) \leftarrow \mathcal{K}_X$, and let $\varphi_X \leftarrow (\iota_X(a_{X,1}, b_{X,1})\colon E_0 \to E_X)$.

Also let $j_{AB} \coloneqq j_{AB}(a_{A,i}, b_{A,i}, a_{B,i}, b_{B,i})$, where $i = 1, 2$ is chosen at random.

Now, given $(E_A, E_B, \varphi_A(P_B), \varphi_A(Q_B), \varphi_B(P_A), \varphi_B(Q_A), j_{AB})$, decide whether $i = 1$ or $i = 2$. ◇

According to Feo, Jao, and Plût [FJP11], SIDH fulfils semantical security properties provided that SSDDH is hard. For example, the SIDH key-exchange protocol itself is session-key secure in the authenticated-links adversarial model of Canetti and Krawczyk [CK01]. The derived public-key encryption protocol, that uses a hash function family $\mathcal{H}$, is IND-CPA secure if $\mathcal{H}$ is entropy smoothing. (For a definition of entropy smoothing, one may resort to Shoup [Sho04]).

Jao et al. [Jao+20] proved in the updated specification for their submission that an IND-CCA-secure public-key encryption protocol, SIKE, can be obtained by applying a post-quantum variant of the Fujisaki-Okamoto transformation from Hofheinz, Hövelmanns, and Kiltz [HHK17]. Here, the underlying security assumption is the weaker SSCDH hardness assumption.

No subexponential attack against the original protocol by Feo, Jao, and Plût [FJP11] is known.

2.2 Practical security

Public parameters Table 2.1 shows the primes p for four realistic public parameter sets that are believed to provide a certain security strength. These primes are the ones that are most likely to be used in practice because they have been submitted to the NIST standardization process by Jao et al. [Jao+20].

Side-channel security It is important to remember that even if SIDH turns out to be theoretically secure, implementations of the protocol might leak information through side channels. For example, if the execution time of a protocol step is a function of some secret value, an attacker might be able to gain information about the secret by analyzing response times during the protocol. Even if there are several other side channels (e. g., power consumption), time is the only one considered in this thesis. In the following chapters, algorithms with inherent time-based side-channel vulnerabilities are avoided, notwithstanding the fact that there are subtleties in the concrete implementation that are out of scope, for example, caching and the translation to assembler code.

Name	bit length of p	p	exp. classical sec.	exp. quantum sec.
SIKEp434	434	$2^{216} \cdot 3^{137} - 1$	108 bits	72 bits
SIKEp503	503	$2^{250} \cdot 3^{159} - 1$	125 bits	83 bits
SIKEp610	610	$2^{305} \cdot 3^{192} - 1$	152 bits	101 bits
SIKEp751	751	$2^{372} \cdot 3^{239} - 1$	186 bits	124 bits

Table 2.1: Choices for the prime p in NIST-submitted parameter sets given by Jao et al. [Jao+20]. The expected bits of classical and quantum security have been obtained following the approach of Costello, Longa, and Naehrig [CLN16, Section 4]: Classical security is assumed to be half the size of the shortest secret (here, $e_A/2$) and quantum security is a third of the shortest secret (here, $e_A/3$).

2.3 Overview over involved algorithms and optimizations

In the rest of the thesis, we discuss at a high level how it is possible to obtain a fast implementation of the SIDH protocol. Such an implementation breaks down into smaller problems for which we give optimized algorithms in the following chapters.

Fig. 2.2 is an overview over the involved problems (in boxes) and interface design choices (question marks) that influence the applicable problem-solving techniques (exclamation marks).

High-level perspective Achieving a fast implementation of the SIDH protocol is the focus of Chapter 4. The exchange of public data divides the protocol into two phases. In both phases, each party essentially computes an isogeny with kernel $\langle [a_X]P + [b_X]Q \rangle$, given points P, Q on an elliptic curve and the secret a_X, b_X.

The computation of this isogeny is done in two steps, the first being the computation of the kernel point $[a_X]P + [b_X]Q$ (see Section 4.3) and the second being the computation of the isogeny using the kernel point (see Section 4.4).

Besides the isogeny computation, some minor tasks are necessary, for example the secret key generation at the beginning of phase 1 and the computation of a j-invariant at the end of phase 2. Their efficient realization is not the focus of the thesis as their performance impact is small.

The algorithms for these high-level problems depend on elementary arithmetic in and between elliptic curves. The available arithmetic operations determine which algorithms are possible

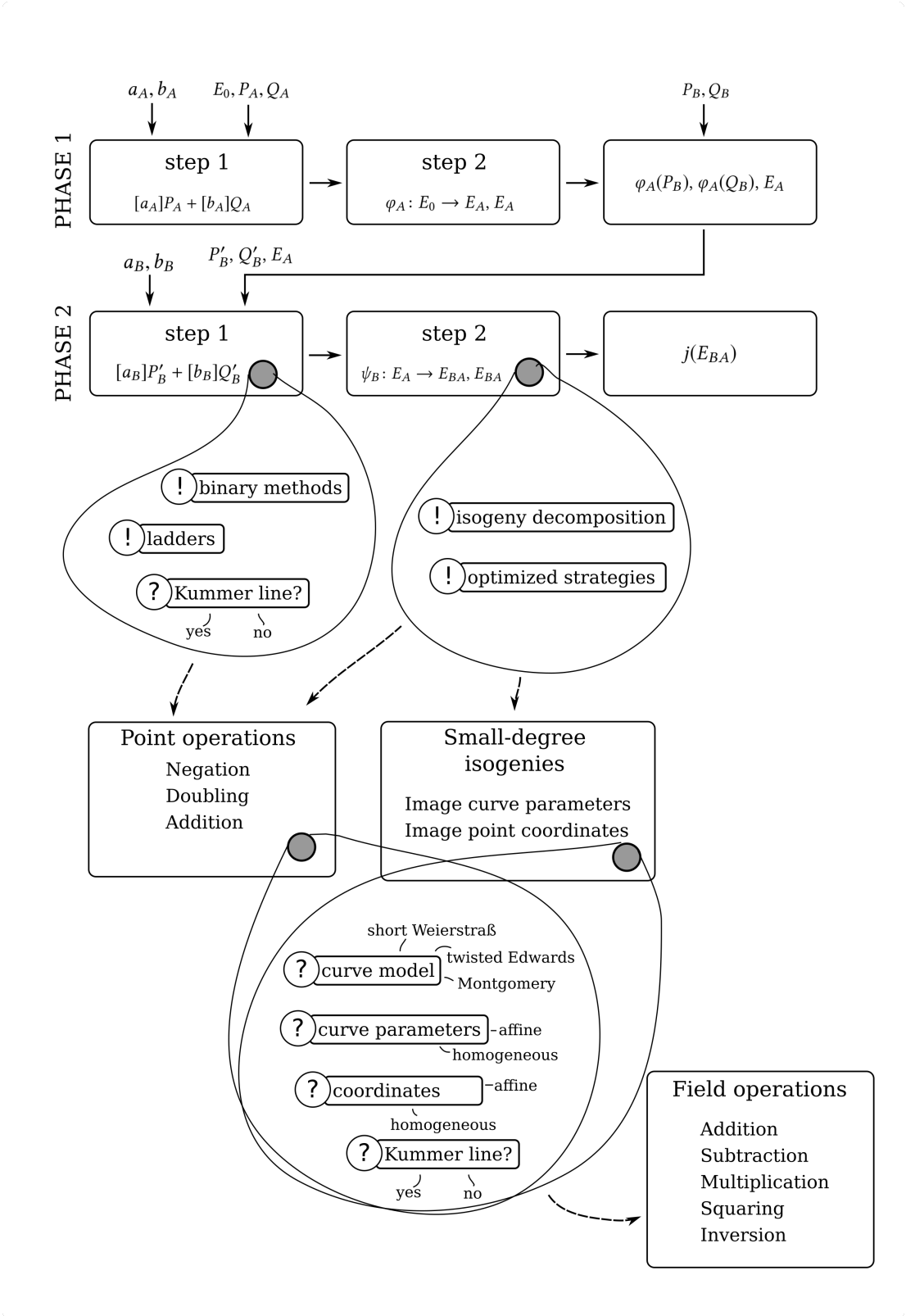


Figure 2.2: Overview of the discussed tasks, choices and optimization techniques involved in the computation of $j(E_{BA})$. Phase 1 happens on Alice's side and phase 2 on Bob's side. The computation of $j(E_{AB})$ is completely analogous.

to implement and the performance of the operations determines the metrics by which we compare the performance of the algorithms. A large imact has the decision to operate on the so-called Kummer line because it restricts the allowed point additions (see Section 3.2.4).

Curve-level perspective The toolbox of elementary curve arithmetic is the focus of Section 3.2. It encompasses the group operations on points, namely negation, doubling and addition, that are addressed in Section 3.2.3, as well as the computation of an isogeny from a small, fixed-size kernel using specialized versions Vélu's formulas which are the topic of Section 3.2.5.

The performance of these operations heavily depends on the representation of the involved curves, points and isogenies, as discussed in Section 3.2.2. A considerable performance boost is obtained from forgetting the signs of the involved points and working on a so-called Kummer line instead of an elliptic curve, as explained in Section 3.2.4.

The supersingular elliptic curves of interest are always defined over the field $\mathbb{F}_{p^2}$ (see Remark 2.1) and the mentioned operations rely on arithmetic in $\mathbb{F}_{p^2}$.

Field-level perspective The field operations of interest are addition, subtraction, multiplication and inversion (division is a combination of multiplication and inversion) in $\mathbb{F}_{p^2}$. Algorithms on the field level are not the focus of the thesis but are briefly addressed in Section 3.1.

3 Efficient field and curve arithmetic

3.1 Field arithmetic

3.1.1 Overview of components

Field arithmetic includes the basic arithmetic operations, namely addition, subtraction, multiplication and inversion, in the fields $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$. It is sometimes helpful to have an extra operation for the multiplication of an element by itself, namely the squaring operation, to allow more performance optimizations.

Algorithms that use field arithmetic can be analyzed using their *field operation count* (the field being $\mathbb{F}_{p^2}$ if not stated otherwise). The operation count summarizes how many of which operations are needed to execute the algorithm. One addition or subtraction is denoted by $\mathbf{a}$, a squaring by $\mathbf{S}$, a multiplication by $\mathbf{M}$ and an inversion by $\mathbf{I}$. For example, an algorithm with an operation count of $3\mathbf{M} + 4\mathbf{S} + 2\mathbf{a}$ requires the execution of three multiplications, four squarings and two additions.

3.1.2 Implementation techniques

We first look at modular arithmetic in $\mathbb{F}_p$ and then build on our results to address extension-field arithmetic in $\mathbb{F}_{p^2}$.

Modular arithmetic Elements of $\mathbb{F}_p \cong \mathbb{Z}/p\mathbb{Z}$ can be represented as integers $0, 1, 2, \ldots p-1$ that represent their congruence class modulo p. To evaluate a modular addition, subtraction or multiplication, the operation is performed in $\mathbb{Z}$. The result of the integer operation is then *reduced*, i. e. the representant of its congruence class in $0, 1, 2, \ldots p-1$ is computed.

Depending on $\log p$, various techniques for integer arithmetic and reduction can be used.

Among others, these are the multiplication of Karatsuba and Ofman [KO62] and Barrett [Bar86] or Montgomery [Mon85] reduction. The general techniques are sometimes tweaked for the exact parameters used, as Bernstein [Ber06b] has done for classical elliptic curve cryptography.

Modular inversion is usually implemented using the Extended Euclidean Algorithm (EEA) and variants or Fermat's Little Theorem (FLT). While EEA is considered to be theoretically faster, it is harder to implement than the FLT inversion algorithm that simply computes $a^{-1} = a^{p-2}$ using a square-and-multiply approach. Furthermore, Azarderakhsh et al. [Aza+19] suggest that FLT is quite competitive in real-world scenarios, at least if it is compared with constant-time implementations.

Extension-field arithmetic Elements of $\mathbb{F}_{p^2} \cong \mathbb{F}_p[X]/(X^2 - d)$ for some non-square $d \in \mathbb{F}_p$ can be represented as $a + bX$, where a, b are elements of $\mathbb{F}_p$. All arithmetic operations are computed using base-field arithmetic operations in a manner analogous to complex number arithmetic. For example, if $p \equiv 3 \pmod 4$, we can choose $d = -1$ and get

$$(a + bX)(c + eX) = (ac - be) + (ae + bc)X = (p - bc + ae) + (bc + ae)X,$$

where $p = (a + b)(c - e)$, and

$$(a + bX)^{-1} = \frac{a - bX}{(a + bX)(a - bX)} = \frac{a - bX}{a^2 + b^2}.$$

Squarings can be computed faster using the binomial formulas

$$(a + bX)^2 = a^2 - b^2 + 2abX = (a + b)(a - b) + 2abX.$$

3.1.3 Performance characteristics

Asymptotically as well as practically, the cost of additions and subtractions is almost negligible compared to squarings and multiplications, which in turn are faster than inversions and divisions. Squarings in $\mathbb{F}_{p^2}$ are faster than multiplications in $\mathbb{F}_{p^2}$.

For example, Feo, Jao, and Plût [FJP11] found that in their implementation, the performance of operations in $\mathbb{F}_{p^2}$ was determined roughly by $\mathbf{I} = 10\mathbf{M}$ and $\mathbf{S} = 0.8\mathbf{M}$. Note that their EEA-based inversion algorithm is not constant-time. However, Azarderakhsh et al. [Aza+19] explain how it

could be made side-channel secure using randomization. Costello, Longa, and Naehrig [CLN16] have used FLT because it is easier to implement and inversions were considered a negligible part of their implementation.

3.2 Curve arithmetic

3.2.1 Overview of components

Curve arithmetic encompasses elementary operations regarding elliptic curves, or more precisely,

- the evaluation of $-P$, $P + Q$ and $[2]P$ for distinct points P, Q on an elliptic curve, and
- the computation of a separable small-degree isogeny whose kernel is generated by a given kernel point.

3.2.2 Representations of points and curves

It is well known that efficiency is not only about algorithms; it is also about data structures. Most people know that this is true for lists: Using an array makes random access to its elements easy, but if you want to insert elements between others you are better off with a linked list. The same is true for matrices: If you want to compute the power of a matrix, you wish the matrix to be in Jordan normal form.

Curve representations Algebraic curves are similar. They can be represented and stored in different ways, in the sense that they are isomorphic or at least birationally equivalent to a variety of interesting curves, which are called *models*. Most models are relatively unhandy: They have lots of parameters and the group operations are relatively expensive. More efficient operations are available on special curves such as short Weierstraß, twisted Edwards or Montgomery curves. As with other data structures, all representations have advantages and disadvantages.

All elliptic curves defined over $\mathbb{F}_{p^2}$ are isomorphic over $\mathbb{F}_{p^2}$ to a short Weierstraß curve, and Feo, Jao, and Plût [FJP11, Section 4.3] showed that in the situation of SIDH, all occurring curves can be represented by a Montgomery curve and twisted Edwards curve.

Fixing a curve model, there is still some leeway in how to represent its parameters. Is, for example, the curve $W_{a,b}$ represented by affine parameters (a, b) or by homogeneous parameters $[A : B : C]$ such that $a = A/C$ and $b = B/C$? The computation of curve parameters, for example of the image curve of some isogeny, might require costly inversions if the resulting curve parameters must be affine.

Point representations Points on a given elliptic curve can also be represented either in affine or homogeneous coordinates, with similar tradeoffs. Bernstein and Lange [BL07] list even more representations in their Explicit Formulas Database.

Kummer line It turns out that not all information that describes a point and a curve is necessary in the SIDH protocol. Omitting this information helps achieving faster algorithms.

3.2.3 Optimized point operations

This section is a discussion of algorithms for point operations on elliptic curves and their performance. The operations of interest are inversions, additions and doublings. Addition in this context referst to the computation of $P + Q$ for two *distinct* points on an elliptic curve E, whereas doubling refers to the computation of $[2]P = P + P$ for a single point P on E. This distinction is necessary because of the different formulas the group law provides for the two special cases of addition. Depending on the curve model, there are sometimes additional restrictions regarding the situations in which an algorithm is applicable.

Because all implementations depend on the chosen curve model as well as the representation of the curve and its points, we restrict our attention to short Weierstraß, twisted Edwards and Montgomery curves. We assume in all algorithms that the curve parameters are affine, while points are given by homogeneous coordinates.

While point negations are trivial to compute in all models, fast addition and doubling operations are nontrivial. In this section, the point doubling algorithm on short Weierstraß curves is derived explicitly to demonstrate some of the tricks involved in the derivation of the algorithms. The remaining doubling and addition formulas found in the literature are presented without derivation.

Short Weierstraß curves

Consider the short Weierstraß curve

$$E\colon y^2 = x^3 + ax + b.$$

Doubling algorithm Let $P_1 = (x_1, y_1) = [X_1 : Y_1 : Z_1] \in E$ s. t. $[2]P \neq O$. We are interested in an explicit formula for $[2]P_1 = (x_+, y_+) = [X_+ : Y_+ : Z_+]$ and derive it from the doubling law given in Theorem 1.13.

Specializing the doubling law given in Theorem 1.13 to the case of the short Weierstraß curve E and bringing the formulas on the common denominator $(2y)^3$ yields

$$x = \lambda^2 - 2x = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1 = \frac{(2y_1)\left(\left(3x_1^2 + a\right)^2 - 2x_1\,(2y_1)^2\right)}{(2y_1)^3} \tag{3.1}$$

$$y = -\lambda x - \nu = \frac{-\left(3x_1^2 + a\right)^3 + 2x\,(2y_1)^2\left(3x_1^2 + a\right) - (2y_1)^2\left(-x_1^3 + ax_1 + b\right)}{(2y_1)^3} \tag{3.2}$$

Homogenization leads to

$$X_+ = \underline{\underline{2Y_1Z_1}}\left(\left(\underline{3X_1^2 + aZ_1^2}\right)^2 - \left(\underline{\underline{2Y_1Z_1}}\right)4X_1Y_1\right) \tag{3.3}$$

$$Y_+ = -\left(\underline{3X_1^2 + aZ_1^2}\right)^3 + \left(\underline{\underline{2Y_1Z_1}}\right)4X_1Y_1\left(\underline{3X_1^2 + aZ_1^2}\right) + \left(\underline{\underline{2Y_1Z_1}}\right)2Y_1\left(\underline{\underline{X_1^3 - aX_1Z_1^2 + bZ_1^3}}\right) \tag{3.4}$$

$$Z_+ = \left(\underline{\underline{2Y_1Z_1}}\right)^3. \tag{3.5}$$

A straightforward evaluation of these formulas without any reuse of intermediate values requires 23**M** + 12**S** + 24**a**.

The terms $S \coloneqq \underline{\underline{2Y_1Z_1}}$ and $W \coloneqq \underline{3X_1^2 + aZ_1^2}$ occur so often that we introduce shortcuts for them. After noticing that

$$\underline{\underline{X_1^3 - aX_1Z_1^2 + bZ_1^3}} = \left(3X_1^3 + aX_1Z_1^2\right) - 2\left(X_1^3 + aX_1Z_1^2 + bZ_1^3\right) = X_1W - 2Y^2Z = X_1W - Y_1S,$$

the formulas simplify to

$$X_+ = S\left(W^2 - 4X_1Y_1S\right), \tag{3.6}$$

$$Y_+ = W\left(6X_1Y_1S - W^2\right) - 2\left(Y_1S\right)^2, \tag{3.7}$$

$$Z_+ = S^3. \tag{3.8}$$

Eqs. (3.6) to (3.8) allow us to compute S and W once and reuse them instead of recalculating. Computing S (1**M** + 1**a**), W (1**M** + 2**S** + 3**a**), X_+ (3**M** + 1**S** + 3**a**), Y_+ (4**M** + 2**S** + 6**a**), and Z_+ (1**M** + 1**S**) in this order requires only 10**M** + 6**S** + 14**a**.

Sharing even more intermediate results, Bernstein and Lange [BL07, Formula dbl-2007-bl] found that computing X_1^2, Z_1^2 (2**S**), W (1**M** + 3**a**), S (1**M** + 1**a**), $R \coloneqq Y_1S, R^2$ (1**M** + 1**S**), $B \coloneqq 2X_1Y_1S$ (1**S** + 3**a**), $H \coloneqq W^2 - 2B$ (1**S** + 2**a**), and finally $X_+ = HS$ (1**M**), $Y_+ = W(B - H) - 2R^2$ (1**M** + 3**a**), and $Z_+ = S^3$ (1**M** + 1**S**) in this order requires in total only 6**M** + 6**S** + 12**a**. This algorithm is shown in Algorithm 3.1.

Algorithm 3.1 A point doubling algorithm using 6**M** + 6**S** + 12**a** on short Weierstraß curves

Input: A point $P = [X_1 : Y_1 : Z_1]$ on a short Weierstraß curve $W_{a,b}$
Precondition: $[2]P$ is affine
Output: $[2]P = [X_+ : Y_+ : Z_+]$

1: $X_s \leftarrow X_1^2$
2: $Z_s \leftarrow Z_1^2$
3: $W \leftarrow 3X_s + aZ_s$
4: $S \leftarrow 2Y_1Z_1$
5: $S_s \leftarrow S^2$
6: $S_t \leftarrow SS_s$
7: $R \leftarrow Y_1S$
8: $R_s \leftarrow R^2$
9: $B \leftarrow (X_1 + R)^2 - X_s - R_s$
10: $H \leftarrow W^2 - 2B$
11: $X_+ \leftarrow HS$
12: $Y_+ \leftarrow W(B - H) - 2R_s$
13: $Z_+ \leftarrow S_t$

Addition algorithm According to the Explicit Formulas Database of Bernstein and Lange [BL07, add-1998-cmo-2], one of the fastest algorithms for point addition on short Weierstraß curves was found by Cohen, Miyaji, and Ono [CMO98, p. 3]. This algorithm has an operation count of 12**M** + 2**S** + 7**a** and is presented in Algorithm 3.2.

Montgomery curves

Now consider a Montgomery curve $M_{A,B}$.

Algorithm 3.2 A point addition algorithm using 12**M** + 2**S** + 7**a** on short Weierstraß curves

Input: Points $P_i = [X_i : Y_i : Z_i]$ $(i = 1, 2)$ on $W_{a,b}$
Precondition: P_1 and P_2 are affine, $P_1 \neq \pm P_2$
Output: $P_1 + P_2 = [X_+ : Y_+ : Z_+]$

1: $A \leftarrow X_1 Z_2$
2: $B \leftarrow Y_1 Z_2$
3: $C \leftarrow Z_1 Z_2$
4: $U \leftarrow Y_2 Z_1 - B$
5: $U_s \leftarrow U^2$
6: $V \leftarrow X_2 Z_1 - A$
7: $V_s \leftarrow V^2$
8: $V_t \leftarrow V V^2$
9: $R \leftarrow V_s A$
10: $D \leftarrow U_s C - V_t - 2R$
11: $X_+ \leftarrow VD$
12: $Y_+ \leftarrow U(R - D) - V_t B$
13: $Z_+ \leftarrow V_t C$

Doubling algorithm The homogenized doubling law for $M_{A,B}$ tells us that if the point $P = [X_1 : Y_1 : Z_1]$ is not of order 1 or 2, then $[2]P = [X_+ : Y_+ : Z_+]$ is given by

$$
\begin{aligned}
X_+ &= B\left(\underline{2BY_1Z_1}\right)\left(\underline{3X_1^2 + 2AX_1Z_1 + Z_1^2}\right)^2 - 2BY_1\left(\underline{2BY_1Z_1}\right)^2 (AZ_1 + 2X_1) \\
Y_+ &= -\left(\underline{3X_1^2 + 2AX_1Z_1 + Z_1^2}\right)\left(B\left(\underline{3X_1^2 + 2AX_1Z_1 + Z_1^2}\right)^2 - 2BY_1\left(\underline{2BY_1Z_1}\right)(AZ_1 + 2X_1)\right) \\
&\quad - \left(-X_1^3 + X_1Z_1^2\right)\left(\underline{2BY_1Z_1}\right)^2 \\
Z_+ &= \left(\underline{2BY_1Z_1}\right)^3 .
\end{aligned}
$$

Setting, among others, $S \coloneqq \underline{2BY_1Z_1}$ and $U \coloneqq 3X_1^2 + 2AX_1Z_1 + Z_1^2$, we obtain Algorithm 3.3 and an operation count of 13**M** + 4**S** + 10**a**.

Algorithm 3.3 A point doubling algorithm on Montgomery curves using 13**M** + 4**S** + 10**a**

Input: A point $P = [X_1 : Y_1 : Z_1]$ $(i = 1, 2)$ on $M_{A,B}$
Precondition: $[2]P$ is affine
Output: $[2]P = [X_1 : Y_1 : Z_1]$

1: $X_s \leftarrow X_1^2$
2: $Z_s \leftarrow Z_1^2$
3: $S \leftarrow 2BY_1Z_1$
4: $S_s \leftarrow S^2$
5: $T \leftarrow AZ_1 + 2X_1$
6: $U \leftarrow 3X_s + 2AX_1Z_1 + Z_s$
7: $V \leftarrow B\left(U^2 - 2Y_1ST\right)$
8: $W \leftarrow X_1(X_s - Z_s)$
9: $X_+ \leftarrow SV$
10: $Y_+ \leftarrow WS_s - UV$
11: $Z_+ \leftarrow SS_s$

Addition algorithm If $P_1 = [X_1 : Y_1 : Z_1]$ and $P_2 = [X_2 : Y_2 : Z_2]$ such that $P_1 \neq \pm P_2$, their sum $P_1 + P_2 = [X_+ : Y_+ : Z_+]$ is given by

$$\begin{aligned} X_+ &= (X_2Z_1 - X_1Z_2)\left(B(Y_2Z_1 - Y_1Z_2)^2 - (AZ_1Z_2 + X_1Z_2 + X_2Z_1)(X_2Z_1 - X_1Z_2)^2\right) \\ Y_+ &= (Y_1Z_2 - Y_2Z_1)\left(B(Y_2Z_1 - Y_1Z_2)^2 - (AZ_1Z_2 + X_1Z_2 + X_2Z_1)(X_2Z_1 - X_1Z_2)^2\right) \\ &\quad - (Y_1X_2 - Y_2X_1)(X_2Z_1 - X_1Z_2)^2 \\ Z_+ &= (X_2Z_1 - X_1Z_2)^3 \end{aligned}$$

This leads to an algorithm using 14**M** + 2**S** + 7**a**, as shown in Algorithm 3.4.

Algorithm 3.4 A point addition algorithm on Montgomery curves using 14**M** + 2**S** + 7**a**

Input: Points $P_i = [X_i : Y_i : Z_i]$ $(i = 1, 2)$ on $M_{A,B}$
Precondition: P_1 and P_2 are affine, $P_1 \neq \pm P_2$
Output: $P_1 + P_2 = [X_+ : Y_+ : Z_+]$

1: $X_1' \leftarrow X_1Z_2$
2: $X_2' \leftarrow X_2Z_1$
3: $D \leftarrow X_2' - X_1'$
4: $D_s \leftarrow D^2$
5: $E \leftarrow Y_1Z_2 - Y_2Z_1$
6: $U \leftarrow AZ_1Z_2 + X_1' + X_2'$
7: $V \leftarrow BE^2 - UD_s$
8: $W \leftarrow Y_1X_2 - Y_2X_1$
9: $X_+ \leftarrow VD$
10: $Y_+ \leftarrow EV - WD_s$
11: $Z_+ \leftarrow DD_s$

Twisted Edwards curves

Let $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$ be affine points on a twisted Edwards curve $E_{a,d}$. Whenever the denominators are nonzero, according to Bernstein et al. [Ber+08], the sum of these points is given by

$$P_1 + P_2 = \left(\frac{x_1y_2 + y_1x_2}{1 + dx_1x_2y_1y_2}, \frac{y_1y_2 - ax_1x_2}{1 - dx_1x_2y_1y_2}\right). \tag{3.9}$$

This holds even when $P_1 = P_2$. However, the restriction that the denominators must be nonzero limits the use of the following algorithms. If d is not a square, it is always satisfied; otherwise, its satisfaction is hard to predict.

Doubling algorithm In the same paper where they introduced twisted Edwards curves, Bernstein et al. [Ber+08, Section 6] gave a doubling algorithm that is still state of the art, according to Bernstein and Lange [BL07, dbl-2008-bbjlp]; see Algorithm 3.5. However, there are in general major restrictions to when the algorithm is correct.

Algorithm 3.5 A point doubling algorithm on twisted Edwards curves using 4**M** + 4**S** + 7**a**

Input: A point $P = [X_1 : Y_1 : Z_1]$ on $E_{a,d}$
Precondition: $Z_1^4 \neq \pm dX_1^2X_2^2$, $Z_1 \neq 0$
Output: $[2]P$

1: $B \leftarrow (X_1 + Y_1)^2$
2: $C \leftarrow X_1^2$
3: $D \leftarrow Y_1^2$
4: $E \leftarrow aC$
5: $F \leftarrow E + D$
6: $H \leftarrow Z_1^2$
7: $J \leftarrow F - 2H$
8: $X_+ \leftarrow (B - C - D)\,J$
9: $Y_+ \leftarrow F\,(E - D)$
10: $Z_+ \leftarrow FJ$

Addition algorithm Bernstein et al. [Ber+08, Section 6] also gave an addition algorithm that is still state of the art according to Bernstein and Lange [BL07, add-2008-bbjlp]; see Algorithm 3.6. Again, the algorithm does not work on all points.

Algorithm 3.6 A point addition algorithm on twisted Edwards curves using 12**M** + 1**S** + 7

Input: Points $P_i = [X_i : Y_i : Z_i]$ $(i = 1, 2)$ on $E_{a,d}$
Precondition: $Z_1^2Z_2^2 \neq \pm dX_1X_2Y_1Y_2$; $Z_1, Z_2 \neq 0$
Output: $P_1 + P_2$

1: $A \leftarrow Z_1Z_2$
2: $B \leftarrow A^2$
3: $C \leftarrow X_1X_2$
4: $D \leftarrow Y_1Y_2$
5: $E \leftarrow dCD$
6: $F \leftarrow B - E$
7: $G \leftarrow B + E$
8: $H \leftarrow (X_1 + Y_2)\,(X_2 + Y_2)$
9: $X_+ \leftarrow AF\,(H - C - D)$
10: $Y_+ \leftarrow AG\,(D - aC)$
11: $Z_+ \leftarrow FG$

3.2.4 The Kummer line

On each of the three special curves discussed, there is an affine coordinate that contains information about the sign of a point P ("Is it P or $-P$?") but nothing more. This is the y-coordinate on short Weierstraß and Montgomery curves (see Theorem 1.13) and the x-coordinate on twisted Edwards curves (according to Bernstein et al. [Ber+08]).

Montgomery [Mon87] was the first to observe that omitting one coordinate in a computation has various advantages. First of all, it is not necessary to compute and store the coordinate, opening the possibility for faster and more space-efficient algorithms. Furthermore, in the case of Montgomery curves, not all curve parameters are relevant for point operations on the x-coordinate only, and this leads to a more compact curve representation.

The problem is, of course, whether it is possible to perform computations with constrained

knowledge about the involved points.

Formalization The idea of forgetting the sign of the points of an elliptic curve E corresponds to the notion of a (topological or set-theoretical) quotient. The resulting space of points, called the *Kummer line* of E by Feo, Jao, and Plût [FJP11], consists of equivalence classes $\{P, -P\}$: A point P is considered equivalent only to P and $-P$. On short Weierstraß and Montgomery curves, the equivalence classes are classes of points with equal x-coordinate. On twisted Edwards curves, they are classes of points with equal y-coordinate.

Pseudo-operations Some point operations on an elliptic curve E induce well-defined operations on the corresponding Kummer line. This is certainly true for the doubling operation because the equivalence class of the result does not depend on the sign of the input point $P \in E$, as the following calculation shows.

$$[2]P \sim -[2]P = [2](-P).$$

On short Weierstraß and Montgomery curves, this means that $x([2]P)$ is uniquely determined by $x(P)$, while the same is true on twisted Edwards curves for the y- instead of the x-coordinate.

So there is an induced doubling operation on the Kummer line of E. This operation is called the *pseudo-doubling* operation.

Unfortunately, there is no induced point addition operation on the Kummer line of E. The reason is that the equivalence class of $P + Q$ might not be the same as that of $P - Q$. On Montgomery curves, for example, $x(P + Q)$ is not uniquely determined by $x(P)$ and $x(Q)$.

It is, however, possible to recover the eqivalence class of $P + Q$ from the equivalence classes of P, Q and $P - Q$. So we can define a *differential addition* or *pseudo-addition* operation on the Kummer line that takes P, Q and $P - Q$ and outputs $P + Q$.

Computations on the Kummer line

Short Weierstraß curves According to the Explicit Formulas Database of Bernstein and Lange [BL07, dbl-2002-bj-3, dadd-2002-it-3], the state-of-the art algorithms are (after applying standard optimizations) the pseudo-doubling algorithm of Brier and Joye [BJ02, Formula (10)] as shown in Algorithm 3.7 and the pseudo-addition algorithm of Izu and Takagi [IT02, Formula

1] as shown in Algorithm 3.8.

Algorithm 3.7 A pseudo-doubling algorithm on short Weierstraß curves using 5**M** + 5**S** + 10**a**

Input: A point $P = [X_1 : \_ : Z_1]$ on $W_{a,b}$
Precondition: $[2]P$ is affine
Output: $[2]P = [X_+ : \_ : Z_+]$

1: $X_s \leftarrow X_1^2$
2: $Z_s \leftarrow Z_1^2$
3: $A \leftarrow 2\left((X_1 + Z_1)^2 - X_s - Z_s\right)$
4: $U \leftarrow aZ_s$
5: $B_d \leftarrow 2b$
6: $X_+ \leftarrow (X_s - U)^2 - B_d A Z_s$
7: $Z_+ \leftarrow A(X_s + U) + 2B_d Z_s^2$

Algorithm 3.8 A pseudo-addition algorithm on short Weierstraß curves using 9**M** + 2**S** + 6**a**

Input: Points $P_i = [X_i : \_ : Z_i]$ $(i = 1, 2, 3)$ on $W_{a,b}$
Precondition: P_1 and P_2 are affine, $P_1 \neq \pm P_2$, $P_3 = P_1 - P_2$, $X_3 \neq 0$
Output: $P_1 + P_2 = [X_+ : \_ : Z_+]$

1: $T_2 \leftarrow Z_1 Z_2$
2: $T_3 \leftarrow X_1 Z_2$
3: $T_4 \leftarrow Z_1 X_2$
4: $T_6 \leftarrow X_1 X_2 - aT_2$
5: $T_{11} \leftarrow 4bT_2(T_3 + T_4)$
6: $X_+ \leftarrow Z_3\left(T_6^2 - T_{11}\right)$
7: $Z_+ \leftarrow X_3(T_3 - T_4)^2$

Montgomery curves According to Bernstein and Lange [BL07, dbl-1987-m-3, dadd-1987-m-3], the pseudo-operations given by Montgomery [Mon87] are still state-of-the-art. See Algorithm 3.9 and Algorithm 3.10. Bernstein [Ber06a] proved that the given pseudo-addition algorithm even works if one or two of the summands have order two, while Montgomery had explicitly excluded this case.

Algorithm 3.9 A pseudo-doubling algorithm on Montgomery curves using 3**M** + 2**S** + 9**a**

Input: A point $P = [X_1 : \_ : Z_1]$ on $M_{A,B}$
Precondition: $[2]P$ is affine
Output: $[2]P = [X_+ : \_ : Z_+]$

1: $A_2 \leftarrow A + 2$
2: $U \leftarrow X_1 + Z_1$
3: $U_s \leftarrow U^2$
4: $V \leftarrow X_1 - Z_1$
5: $V_s \leftarrow V^2$
6: $W \leftarrow U_s - V_s$
7: $X_+ \leftarrow 4U_s V_s$
8: $Z_+ \leftarrow W(4V_s + A_2 W)$

Twisted Edwards curves Castryck, Galbraith, and Farashahi [CGF08] have found a pseudo-doubling algorithm on generalized Edwards curves, i. e., twisted Edwards curves $E_{a,d}$ with

Algorithm 3.10 A pseudo-addition algorithm on Montgomery curves using 4**M** + 2**S** + 6**a**

Input: Points $P_i = [X_i : \_ : Z_i] \neq O$ on $M_{A,B}$
Precondition: P_1 and P_2 are affine, $P_1 \neq \pm P_2$, $P_3 = P_1 - P_2$, $[2]P_3$ is affine
Output: $P_1 + P_2 = [X_+ : \_ Z_+]$

1: $R \leftarrow X_2 + Z_2$
2: $S \leftarrow X_2 - Z_2$
3: $T \leftarrow X_3 + Z_3$
4: $U \leftarrow X_3 - Z_3$
5: $V \leftarrow RU$
6: $W \leftarrow ST$
7: $X_+ \leftarrow Z_1 (V + W)^2$
8: $Z_+ \leftarrow X_1 (V - W)^2$

$a = 1$. An algorithm for *twisted* Edwards curves that was derived analogously is shown in Algorithm 3.11. The value $a^{-1}d$ must be precomputed in order to achieve a few multiplications per point operation as possible. Otherwise, more multiplications are needed.

Algorithm 3.11 A pseudo-doubling algorithm on twisted Edwards curves using 2**M** + 5**S** + 6**a** if $a^{-1}d$ is precomputed

Input: A point $P = [\_ : Y_1 : Z_1]$ on $E_{a,d}$
Precondition: $Z_1^4 + a^{-1}dY_1^4 \neq 2a^{-1}dY_1^2Z_1^2$, $Z_1 \neq 0$
Output: $[2]P = [\_ : Y_+ : Z_+]$

1: $Y_s \leftarrow Y_1^2$
2: $Z_s \leftarrow Z_1^2$
3: $Y_q \leftarrow Y^4$
4: $Z_q \leftarrow Z^4$
5: $S \leftarrow (Y_s + Z_s)^2 - Y_q - Z_q$
6: $T \leftarrow Z_q + a^{-1}dY_q$
7: $Y_+ \leftarrow S - T$
8: $Z_+ \leftarrow T - a^{-1}dS$

A pseudo-addition algorithm on Edwards curves was proposed by Justus and Loebenberger [JL09]. Although the inventors only claimed that their formula works for nonsquare d, this requirement only serves to tame the nasty requirement that the denominators in the addition law Eq. (3.9) must be nonzero. If used carefully, it can be also applied if d is a square. An analogous algorithm that makes use of the precomputed constant $a^{-1}d$ for twisted Edwards curves is shown in Algorithm 3.12.

Remark 3.1 It is helpful to observe that the pseudo-operations on a twisted Edwards curve $E_{a,d}$ only depend on $a^{-1}d$. Indeed, scaling a and d by the same factor e^2, the x-coordinate of a point is scaled by $1/e$ if the y-coordinate is fixed. Note that this is analogous to the situation on a Montgomery curve $M_{A,B}$, whose parameter B does not influence the group law. ◇

Algorithm 3.12 A pseudo-addition algorithm on twisted Edwards curves using 7**M** + 4**S** + 7**a** if $a^{-1}d$ is precomputed

Input: Points $P_i = [\_ : Y_i : Z_i]$ $(i = 1, 2)$ on $E_{a,d}$
Precondition: $P_1 - P_2 = P_3$; $Z_1, Z_2, Z_3 \neq 0$; $X_3 \neq 0$
Precondition: $Z_1^2 Z_2^2 \left(aZ_1^2 - dY_1^2\right)\left(aZ_2^2 - dY_2^2\right) \neq dY_1^2 Y_2^2 \left(Z_1^2 - Y_1^2\right)\left(Z_2^2 - Y_2^2\right)$
Output: $P_1 + P_2 = [\_ : Y_+ : Z_+]$

1: $Y_{1,s} \leftarrow Y_1^2$
2: $Z_{1,s} \leftarrow Z_1^2$
3: $Y_{2,s} \leftarrow Y_2^2$
4: $Z_{2,s} \leftarrow Z_2^2$
5: $U \leftarrow Y_{2,s} - Z_{2,s}$
6: $V \leftarrow Z_{2,s} - a^{-1}dY_{2,s}$
7: $W \leftarrow \left(Y_{1,s} + Z_{1,s}\right)(U + V)$
8: $S \leftarrow Y_{1,s}U$
9: $T \leftarrow Z_{1,s}V$
10: $Y_+ \leftarrow Z_3\left(W - S - T\right)$
11: $Z_+ \leftarrow Y_3\left(a^{-1}dS + T\right)$

Curve operation	$\mathbb{F}_{p^2}$ operation count		
	Short Weierstraß	Twisted Edwards	Montgomery
Addition	12**M** + 2**S** + 7**a**	12**M** + 1**S** + 7**a**	14**M** + 2**S** + 7**a***
Doubling	6**M** + 6**S** + 12**a**	4**M** + 4**S** + 7**a**	13**M** + 4**S** + 10**a***
Pseudo-addition	9**M** + 2**S** + 6**a**	7**M** + 4**S** + 7**a**	4**M** + 2**S** + 6**a**
Pseudo-doubling	5**M** + 5**S** + 10**a**	2**M** + 5**S** + 6**a**	3**M** + 2**S** + 9**a**

Figure 3.1: Operation counts (see Section 3.1.1) required by the presented point operation algorithms on different curves. Entries annotated with * are the result of an ad-hoc derivation and might leave room for optimizations.

Summary

Fig. 3.1 summarizes the operation counts for point operations on different curves. Care has to be taken for the individual restrictions the presented algorithms impose on the input points and on precomputations.

After all, twisted Edwards curves excel when additions and doublings are used, while Montgomery curves provide the fastest pseudo-operations. Operations on the Kummer line are considerably faster than operations on the elliptic curve itself, regardless of the curve model.

All presented algorithms need a constant number of multiplications, squarings, additions and subtractions, a fact that helps protecting against side-channel attacks because it makes it easier to come up with a constant-time implementation.

However, the requirements of operations on twisted Edwards curves are so chaotic that it is sometimes necessary to switch to a corresponding Montgomery curve. The unpredictability of the switch makes twisted Edwards curves less suited for constant-time implementations.

3.2.5 Optimized small-degree isogeny computation on Montgomery curves

Motivation We show in Section 4.4.2 that every separable isogeny with a cyclic kernel of order ℓ^e, where ℓ and e are natural numbers, is a composition of cyclic isogenies of degree ℓ. For example, if $\ell_A = 2$ and $\ell_B = 3$, every cyclic separable isogeny of degree ${\ell_A}^{e_A}$ (respectively, ${\ell_B}^{e_B}$) is composed of cyclic separable isogenies of degree 2 (respectively, 3). If e_A is even, we can alternatively rely on 4-isogenies instead of 2-isogenies.

This section is about the computation of small-degree isogenies. If e_A is even, and this is the case for all NIST-submitted parameter sets (see Table 2.1), the implementation of Costello, Longa, and Naehrig [CLN16] and the NIST submission rely on 3- and 4-isogenies.

Assumptions We focus on the computation of cyclic 3- and 4-isogenies. Being the most frequent choice in the context of SIDH, we only address the case of Montgomery curves and work on their Kummer line. The curve parameters are represented by homogeneous coordinates and we write $M_{[A:B:C]}$ for the Montgomery curve with first parameter A/C and second parameter B/C.

If we would use an affine representation of the parameters, our algorithms would need to compute field inversions, which are expensive. Note that our choice of homogeneous parameters is incompatible with the choice of affine parameters in the previous section when we gave algorithms for point operations. For a real implementation, the addition and doubling algorithms need to be adopted to homogeneous parameter representations.

Goal Given a point Q of order 3 or 4 on a Montgomery curve $E = M_{[A:B:C]}$ over $\mathbb{F}_{p^2}$, the goal is to find a separable isogeny $\varphi\colon E \to E' = M_{[A':B':C']}$ with kernel $\langle Q \rangle$. More precisely, we want to compute the parameters of the image curve and evaluate φ at a point P of E.

Kummer line considerations Because φ is a group homomorphism, the x-coordinate of $\varphi(P)$ is determined by the x-coordinate of P. Furthermore, the isogeny φ itself is determined

by the x-coordinate of its kernel point Q. Hence, our algorithms can ignore the y-coordinate and work with x-only coordinates.

The curve parameter B of Montgomery curves $M_{[A:B:C]}$ is also irrelevant, as it only affects the scaling of the y-coordinate of the involved points. Because vertical scaling is an isomorphism, the j-invariant of the curves is also independent from B. These omissions lead to faster algorithms.

Isogenies of degree 3 on Kummer lines

Let $M_{[A:\_:C]}$ be a Montgomery curve and $Q = [X_3 : \_ : Z_3]$ a point of order 3. Feo, Jao, and Plût [FJP11] derived affine formulas for the separable isogeny with kernel point Q. Later, Costello, Longa, and Naehrig [CLN16] gave projective formulas. The image curve $M_{[A':\_:C']}$ is determined by

$$[A' : \_ : C'] = \left[Z_3^4 + 18X_3^2Z_3^2 - 27X_3^4 : \_ : 4X_3Z_3^3\right],$$

and for $P = [X : \_ : Z] \in M_{[A:\_:C]}$,

$$\varphi(P) = \left[X\left(X_3X - Z_3Z\right)^2 : \_ : Z\left(Z_3X - X_3Z\right)^2\right].$$

Costello and Hisil [CH17] observed that

$$A' - 2C' = Z_3^4 - 8X_3Z_3^3 + 18X_3^2Z_3^2 - 27X_3^4 = (X_3 + Z_3)\left(Z_3 - 3X_3\right)^3,$$

an observation that leads to algorithm Algorithm 3.13 for the computation of the image curve parameters. The variable V_1 first gets the value $-(A' - 2C')$. Considering V_1 and V_2 as functions of X_3 and Z_3, $V_2(X_3, Z_3)$ equals $V_1(X_3, -Z_3)$. Then the algorithm takes advantage of the fact that the even part of V_1 is $\frac{1}{2}(V_1 + V_2) = -A'$ and that the odd part is $\frac{1}{2}(V_1 - V_2) = 2C'$.

Algorithm 3.14 for the evaluation of a 3-isogeny can be understood from a similar perspective. The variable T_1 gets

$$(X + Z)(X_3 - Z_3) = X_3X - Z_3Z + X_3Z - Z_3X,$$

and T_2 is obtained from T_1 by negating (Z, Z_3).

Algorithm 3.13 A 3-isogeny image curve algorithm on Kummer lines using 2**M** + 3**S** + 14**a**

Input: A point $Q = [X_3 : \_ : Z_3]$ on $M_{[A:\_:C]}$ of degree 3
Output: The image curve $M_{[A':\_:C']}$ of the isogeny with kernel $\langle Q \rangle$

1: $C_1 \leftarrow X_3 - Z_3$
2: $C_2 \leftarrow X_3 + Z_3$
3: $R_1 \leftarrow C_1^2$
4: $R_2 \leftarrow C_2^2$
5: $S \leftarrow (C_1 + C_2)^2$
6: $T_1 \leftarrow S - R_2$
7: $T_2 \leftarrow S - R_1$
8: $U_1 \leftarrow 2(R_1 + T_1) + R_2$
9: $U_2 \leftarrow 2(R_2 + T_2) + R_1$
10: $V_1 \leftarrow U_1 T_2$
11: $V_2 \leftarrow U_2 T_1$
12: $A' \leftarrow V_1 + V_2$
13: $C' \leftarrow 2(V_2 - V_1)$

Algorithm 3.14 A 3-isogeny evaluation algorithm on Kummer lines using 4**M** + 2**S** + 4**a**

Input: Points $P = [X : \_ : Z], Q = [X_3 : \_ : Z_3]$ on $M_{[A:\_:C]}$, Q of degree 3
Input: $C_1 = X_3 - Z_3, C_2 = X_3 + Z_3$
Output: The image $[X' : \_ : Z']$ of P under the isogeny with kernel $\langle Q \rangle$

1: $T_1 \leftarrow (X + Z) C_1$
2: $T_2 \leftarrow (X - Z) C_2$
3: $X' \leftarrow X (T_1 + T_2)^2$
4: $Z' \leftarrow Z (T_1 - T_2)^2$

Isogenies of degree 4 on Kummer lines

Finding 4-isogenies is complicated because we must distinguish two cases. Feo, Jao, and Plût [FJP11, Equations (19)-(21)] gave explicit formulas for the computation of 4-isogenies in the special case that the kernel point Q fulfils $x_Q = 1$. It is trivial to construct an analogous 4-isogeny when $x_Q = -1$. Together, these formulas cover the case that $[2]Q = (0, 0)$.

Feo, Jao, and Plût [FJP11, Equation (15)] also gave an isomorphism of Montgomery curves that maps a point Q of order 4 to a point with x-coordinate 1 in order to apply their 4-isogeny when the kernel point does not lie above $(0, 0)$.

From these building blocks, Costello, Longa, and Naehrig [CLN16] derived simple formulas for 4-isogenies, one assuming that x_Q is 1 (the modifications for $x_Q = -1$ are obvious) and one assuming that x_Q is neither 1 nor -1.

Isogenies with kernel points above $(0, 0)$ Let $E = M_{[A:\_:C]}$ be a Montgomery curve over $\mathbb{F}_{p^2}$ and $Q = (\pm 1, \ldots) \in E$ a point above $(0, 0)$. Then the separable isogeny $\varphi\colon E \to E'$ with kernel $\langle Q \rangle$ has the image curve $E' = M_{[A':\_:C']}$, where

$$[A' : C'] = [2(\pm A + 6C) : \pm A - 2C],$$

and it can be evaluated at $P = [X : \_ : Z] \in E \setminus \{O\}$ using the formula

$$\varphi(P) = \left[\pm(\pm X + Z)^2 \left(AXZ + CX^2 + CZ^2\right) : \_ : (2C \mp A) XZ (\pm X - Z)^2\right].$$

The corresponding algorithms for the image curve and the evaluation are shown in Algorithm 3.15 and Algorithm 3.16.

Algorithm 3.15 A 4-isogeny image curve algorithm, first case, using 6**a**

Input: A point $P = (\pm 1, \ldots)$ on $M_{[A:\_:C]}$
Output: The image curve $M_{[A':\_:C']}$ of the isogeny with kernel $\langle Q \rangle$

1: $A \leftarrow \pm A$
2: $C_d \leftarrow 2C$
3: $A' \leftarrow 2(A + 3C_d)$
4: $C' \leftarrow A - C_d$

Algorithm 3.16 A 4-isogeny evaluation algorithm, first case, using 5**M** + 2**S** + 7**a**

Input: Points $P = [X : \_ : Z], Q = (\pm 1, \ldots)$ on $M_{[A:\_:C]}$
Output: The image $[X' : \_ : Z']$ of P under the isogeny with kernel $\langle Q \rangle$

1: $A \leftarrow \pm A$
2: $X \leftarrow \pm X$
3: $C_d \leftarrow 2C$
4: $R_1 \leftarrow (X - Z)^2$
5: $R_2 \leftarrow (X + Z)^2$
6: $S \leftarrow R_2 - R_1$
7: $T \leftarrow R_2 + R_1$
8: $X' \leftarrow R_2 (AS + C_d T)$
9: $Z' \leftarrow (C_d - A) SR_1$

Isogenies with kernel points not above $(0, 0)$ Let E, A and C be as above but now let $Q = [X_4 : \_ : Z_4]$ be a point on E with $[2]Q \neq (0, 0)$. Then the separable isogeny $\varphi\colon E \to E'$ with kernel $\langle Q \rangle$ has the image curve $E' = M_{[A':\_:C']}$, where

$$[A' : C'] = \left[2\left(2X_4^4 - Z_4^4\right) : Z_4^4\right],$$

and it can be evaluated at $P = [X : \_ : Z] \in E \setminus \{O\}$ using the formula

$$\begin{aligned}\varphi(P) = [&X\left(2X_Q Z_Q Z - X\left(X_Q^2 + Z_Q^2\right)\right)\left(X_Q X - Z_Q Z\right)^2 : \_ : \\ &Z\left(2X_Q Z_Q X - Z\left(X_Q^2 + Z_Q^2\right)\right)\left(Z_Q X - X_Q Z\right)^2].\end{aligned}$$

The algorithms given by Costello, Longa, and Naehrig [CLN16] are suboptimal. The idea for the state-of-the-art algorithms for 4-isogenies, as it is part of the NIST-submitted optimized

implementation, is due to Costello and Hisil [CH17], although there appears to be a mistake in their algorithm such that it does not match the claimed operation count. Curve parameter computation is done in Algorithm 3.17, while isogeny evaluation is presented in Algorithm 3.18.

While the algorithm of Costello and Hisil [CH17] computes $[A + 2 : 4C]$ instead of $[A : C]$ for performance reasons, the performance gain from that optimization is restricted to a few additions and we ignore this optimization for the sake of simplicity. Algorithm 3.17 precomputes three values C_1, C_2, C_3 that are reused in Algorithm 3.18.

Algorithm 3.17 A 4-isogeny image curve algorithm, second case, using 4**S** + 7**a**

Input: A point $Q = [X : \_ : Z]$ on $M_{[A:\_:C]}$
Precondition: Q has degree 4, $[2]Q \neq (0, 0)$
Output: The image curve $M_{[A':\_:C']}$ of the isogeny with kernel $\langle Q \rangle$
Output: $C_1 = 4Z^2, C_2 = X - Z, C_3 = X + Z$

1: $Z_s \leftarrow Z^2$
2: $C_1 \leftarrow 4Z_s$
3: $C_2 \leftarrow X - Z$
4: $C_3 \leftarrow X + Z$
5: $C' \leftarrow Z_s^2$
6: $A' \leftarrow 2\left(2X^4 - C'\right)$

Algorithm 3.18 A 4-isogeny evaluation algorithm, second case, using 6**M** + 2**S** + 6**a**

Input: Points $P = [X_P : \_ : Z_P], Q = \left[X_Q : \_ : Z_Q\right]$ on $M_{[A:\_:C]}$
Input: $C_1 = 4Z_Q^2, C_2 = X_Q - Z_Q, C_3 = X_Q + Z_Q$
Precondition: Q has degree 4, $[2]Q \neq (0, 0)$
Output: The image $[X' : \_ : Z']$ of P under the isogeny with kernel $\langle Q \rangle$

1: $D_2 \leftarrow X - Z$
2: $D_3 \leftarrow X + Z$
3: $R \leftarrow C_2 D_3$
4: $S \leftarrow C_3 D_2$
5: $T \leftarrow C_1 D_2 D_3$
6: $U \leftarrow (R + S)^2$
7: $V \leftarrow (R - S)^2$
8: $X' \leftarrow (U + T)\, U$
9: $Z' \leftarrow (V - T)\, V$

Conclusion

The previous section showed that the performance of 3- and 4-isogeny computations on Montgomery Kummer lines is comparable to the performance of pseudo-operations. The presented algorithms require a constant $\mathbb{F}_{p^2}$ operation count which makes them suitable for constant-time implementations of SIDH. The resulting operation counts are summarized in Table 3.1.

However, 4-isogenies are troubling. Depending on whether their kernel point Q lies above $(0, 0)$, different algorithms need to be used. This situation seems to lead to needlessly complex

Operation	3-isogenies	4-isogenies, $(0,0) \mapsto O$	4-isogenies, otherwise
image curve	2**M** + 3**S** + 14**a**	0**M** + 0**S** + 6**a**	0**M** + 4**S** + 7**a**
evaluation	4**M** + 2**S** + 4**a**	5**M** + 2**S** + 7**a**	6**M** + 2**S** + 6**a**

Table 3.1: Operation counts for 3- and 4-isogeny operations

implementations and a loss of the constant-time property if not handled wisely.

4 Efficient algorithms for SIDH

4.1 Overview of components

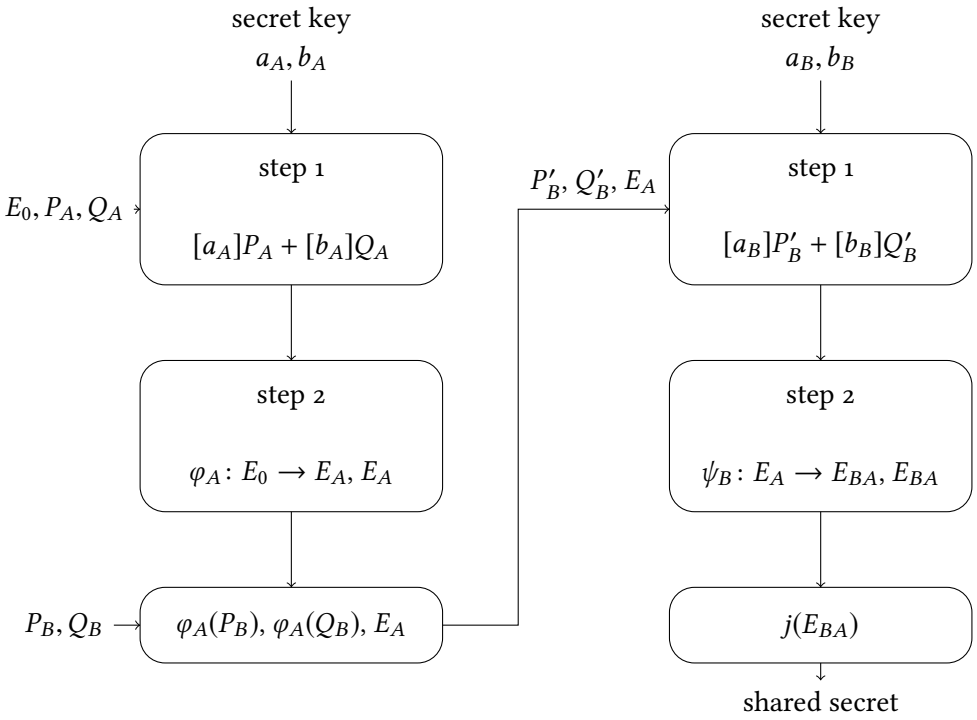


Figure 4.1: Components of SIDH involved in the computation of $j(E_{BA})$. Phase 1 happens on Alice's side and phase 2 on Bob's side. The computation of $j(E_{AB})$ is completely analogous.

This section is about high-level algorithms needed for a realization of the SIDH protocol. The protocol is symmetric: Both parties, Alice and Bob, follow the same procedure. This procedure is divided into smaller components, as shown in Fig. 4.1.

First of all, the protocol is divided into two phases by the exchange of public keys: Alice and

Bob compute their public key in the first phase and the shared secret in the second phase.

Although the first and second phase are not identical, both contain the computationally expensive task of computing a separable isogeny with kernel $[a_X]P + [b_X]Q$, given an elliptic curve E, numbers a_X and b_X and points P and Q that generate $E[n_X]$. This task consists of two steps:

1. Compute $R = [a_X]P + [b_X]Q$.
2. Find an isogeny whose kernel is generated by R.

There is a bit more work to be done: In the first phase, the secret key has to be generated and φ must be evaluated at two points. In the second phase, the j-invariant of the image curve, as it is the shared secret, has to be computed. The real performance bottlenecks are, however, steps 1 and 2.

4.2 Secret key generation

Arbitrary secret key generation It is easy to generate an arbitrary secret key. First, generate a tuple of numbers a, b from $\{0, \ldots, n_X - 1\}$. Repeat this procedure until either a or b are invertible modulo n_X, i. e., not divisible by ℓ_X.

Restricting the key space Every secret key (a, b) contains superfluous information that has no impact on the resulting public and shared key. The keys (a, b) and (a', b') lead to the same results whenever $\langle [a]P_X + [b]Q_X \rangle$ equals $\langle [a']P_X + [b]Q_X \rangle$. We call the generated subgroup the effective secret key. Because $E[n_X]$ is isomorphic to $(\mathbb{Z}/n_X\mathbb{Z})^2$, the two keys represent the same effective key if $a' \equiv \lambda a$ and $b' \equiv \lambda b$ for some integer λ that is invertible modulo n_X.

Recall that either a or b is required to be invertible modulo n_X. Assume without loss of generality that a is invertible. Setting $\lambda = a^{-1} \pmod{n}$, the key $\left(1, a^{-1}b\right)$ represents the same effective secret key as (a, b).

Hence, at least half of all effective secret keys are represented by a secret key of the form $(1, b)$ (the remaining ones having a representative of the form $(a, 1)$). This is why most real SIDH implementations, for example the one of Costello, Longa, and Naehrig [CLN16], restrict the key space to the case $(1, b)$ in order to simplify key generation and step 1, although it costs about one bit of security. We assume this simplification in the following section.

4.3 Step 1: Computation of the kernel point

4.3.1 Situation and goals

Situation In step 1 we are given generators P and Q of the n-torsion $E[n]$ of an elliptic curve E, where $n = \ell^e$ is a prime power. Furthermore, we are given coefficients $a = 1$ and b from $\mathbb{Z}/n\mathbb{Z}$.

Computational goal The given inputs define a point $R := [1]P + [b]Q$. The goal is to compute R.

Security goal Classical algorithms for the computational goal are not side-channel secure because their execution time and power consumption depend on the secret numbers a and b, as we observe below. The goal is to find algorithms that are not vulnerable through these side-channels. This leads to a trade-off between efficiency and security.

Outline We approach the problem of computing $P + [b]Q$ in two subsections. The first subsection is about classical scalar multiplication algorithms that work in any abelian group. Those form the foundation of the more sophisticated algorithms over Montgomery curves that are presented in the second subsection. They are less vulnerable to side-channel attacks and more efficient than the classical ones, as we will see.

4.3.2 Double and add

Suppose we need to compute $P + [13]Q$. We could simply start from P and add Q thirteen times. But that strategy does not scale up in a reasonably efficient way if we are in the setting of, for example, `SIKEp434` where we might need to compute $P + [2^{200}]Q$. The scalar multiplication is a performance bottleneck.

As Knuth [Knu81] has documented, much thought has been dedicated to the problem of scalar multiplication in such additive groups (or originally, exponentiation in multiplicative groups). We follow Knuth in stating algorithms that are thousands of years old and take a look at their computational hardness depending on the scalar factor.

The classical algorithms of interest are known as *binary methods*. Their name suggests that they consume the binary representation of the scalar coefficient b bit by bit, finally returning $[b]Q$. A *left-to-right* binary method first consumes the highest bit down to the lowest one. Consuming increasingly more bits, the algorithm computes $[b']Q$ for increasingly large prefixes of b; a *right-to-left* binary method starts with the lowest bit, working with suffixes instead of prefixes. Note that there is more than a single algorithm qualifying as a binary method.

Remark 4.1 The terminology of "left" and "right" can be confusing. When we look at the binary representation of $n \in \mathbb{N}_0$, it is usually written with the most significant bit on the left, `1101` representing 13. However, following the conventions for polynomial coefficients, the digits are usually indexed according to their significance. From this perspective, it seems intuitive to write $13 = 1 + 0 \cdot 2 + 1 \cdot 4 + 1 \cdot 8$, or more generally, $n = \sum_{i=0}^{l-1} n_i \cdot 2^i$, so that the *least* significant bit is on the left. Following Knuth [Knu81], "left-to-right" is used in the sense of "top-down" or "significance-decreasing", the most significant bit being considered as the leftmost one. ◇

Example 4.2 (A left-to-right binary method) The binary representation of 13 is `1101`. The prefixes correspond to the numbers 1, $3 = 1 + 2 \cdot 1$, $6 = 2 \cdot 3$, and $13 = 1 + 2 \cdot 6$. Clearly, $[1]Q = Q$. Multiplying the equations by Q, we can compute

$$A \leftarrow [1]Q, \quad B \leftarrow [3]Q = Q + [2]A, \quad C \leftarrow [6]Q = [2]B, \quad [13]Q = Q + [2]C.$$ ◇

See Fig. 4.2a.

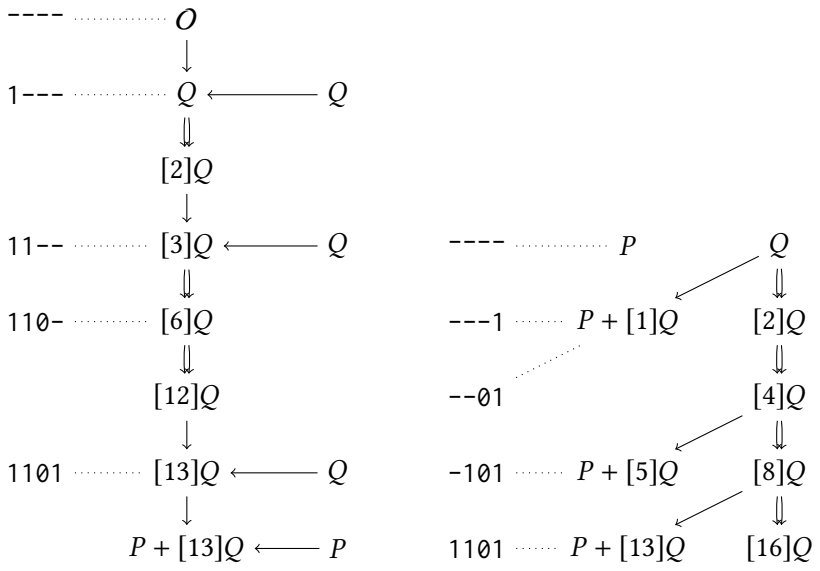
Example 4.3 (A right-to-left binary method) The suffixes of `1101` correspond to the numbers 1, $1 = 0 \cdot 1 + 2$, $5 = 1 + 4$, and $13 = 1 + 4 + 8$. Again multiplying the equations by Q, we can compute from the left to the right

$$[13]Q = Q + [4]Q + [8]Q.$$

Note that $[8]Q = [2][4]Q$. See Fig. 4.2b. ◇

A binary method that is implemented using doublings and additions in the straight-forward way as in the examples is called a *double-and-add* algorithm. The left-to-right and right-to-left variants are shown in Algorithm 4.1 and Algorithm 4.2.

Both methods require the same number of group additions and doublings: a doubling for every digit and an addition for every digit that is one.



(a) left-to-right (b) right-to-left

Figure 4.2: Visualization of the double-and-add algorithms for $P + [13]Q$

Algorithm 4.1 Double and add (left-to-right).

Input: An elliptic curve E; $P, Q \in E$; $n = \sum_{i=0}^{l-1} n_i 2^i \in \{0, \ldots, 2^l - 1\}$
Output: $P + [n]Q$
Notation: $n_{[a:b]} = \sum_{i=a}^{b} n_i 2^{i-a}$

1: $R \leftarrow O$
2: **for** i **from** $l - 1$ **down to** 0 **do**
3: $\quad R \leftarrow [2]R$
4: $\quad$ **if** $n_l = 1$ **then**
5: $\quad\quad R \leftarrow R + Q$
6: $\quad$ **end if** ▷ $R = [n_{[i:l-1]}]Q$
7: **end for**
8: **return** $P + R$

Algorithm 4.2 Double and add (right-to-left)

Input: An elliptic curve E; $P, Q \in E$; $n = \sum_{i=0}^{l-1} n_i 2^i \in \{0, \ldots, 2^l - 1\}$
Output: $P + [n]Q$
Notation: $n_{[a:b]} = \sum_{i=a}^{b} n_i 2^{i-a}$

1: $R \leftarrow \mathcal{O}$
2: $Q_{\text{pow}} \leftarrow Q$
3: **for** i **from** 0 **to** $l - 1$ **do**
4: **if** $n_i = 1$ **then**
5: $R \leftarrow R + Q_{\text{pow}}$
6: **end if** ▷ $R = [n_{[0:i]}]Q$
7: $Q_{\text{pow}} \leftarrow [2]Q_{\text{pow}}$ ▷ skip in the last step
8: **end for**
9: **return** $P + R$

We can compute R relatively quickly using a double-and-add algorithm. The downside is that execution time and power consumption depend on the number and (more critically) the pattern of 1's in the binary representation of the secret scalar b. Thus, double-and-add algorithms are vulnerable to side-channel attacks.

4.3.3 A constant-time algorithm: Montgomery ladders

A more secure alternative to double-and-add is a Montgomery ladder as shown in Algorithm 4.3 because the loop body consists of an addition and a doubling, in this order, regardless of the value of the scalar factor.

Algorithm 4.3 Montgomery ladder for scalar multiplication

Input: an elliptic curve E, $P, Q \in E$, $n = \sum_{i=0}^{l-1} n_i 2^i \in \{0, \ldots, 2^l - 1\}$
Output: $P + [n]Q$
Notation: $n_{[a:b]} = \sum_{i=a}^{b} n_i 2^{i-a}$

1: $R \leftarrow \mathcal{O}$
2: $R_{+Q} \leftarrow Q$
3: **for** i **from** $l - 1$ **down to** 0 **do**
4: **if** $n_i = 0$ **then**
5: $(R, R_{+Q}) \leftarrow ([2]R, R + R_{+Q})$
6: **else**
7: $(R, R_{+Q}) \leftarrow (R + R_{+Q}, [2]R_{+Q})$
8: **end if** ▷ $R = [n_{[i:l-1]}]Q$
9: **end for**
10: **return** $P + R$

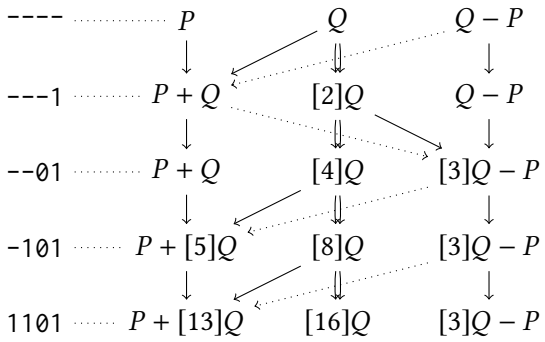


Figure 4.3: Visualization of the three-point ladder of Faz-Hernández et al. [Faz+18] computing $P + [13]Q$ using differential addition

Counted in group operations, this algorithm is slower because it is less adaptive than double-and-add. It requires an addition for *every* digit, not just for every digit that is one. But this comparison assumes that the listings are implemented using ordinary group operations on the curve instead of pseudo-operations on the Kummer line. The summands in lines 5 and 7 of Algorithm 4.3 both differ by Q, so that the Montgomery ladder can, if we ignore line 10, leverage faster pseudo-operations. This is a performance advantage of the Montgomery ladder compared to double-and-add.

But assuming that lines 1 to 9 are implemented on a Montgomery curve, it is not clear how to perform the addition $P + [b]Q$ in line 10 using pseudo-operations. A differential addition seems impossible because $P - [b]Q$ is unknown. The first solution to this problem — a left-to-right three-point ladder that is slower than double-and-add — was given in 2011 by de Feo and Jao in their introductory SIDH paper by Feo, Jao, and Plût [FJP11]. Later, Faz-Hernández et al. [Faz+18] proposed two alternative approaches.

The first approach requires the y-coordinate of P and Q to be known. Using the formula of Okeya and Sakurai [OS01] to reconstruct the y-coordinate of the result in a Montgomery model, $y(R)$ can be recovered from $x(R)$, $x(R_{+Q})$, $x(Q)$ and $y(Q)$ before line 10 of Algorithm 4.3. After that, $P + R$ can be computed using ordinary group operations. Many modern implementations, however, do not follow this approach, so that we focus on the second approach.

The second approach is a right-to-left three-point ladder. In the i-indexed step, the algorithm computes $P + [n^{(i)}]Q$, $[2^{i+1}]Q$ and $-P + [2^{i+1} - n^{(i)}]Q$ for the i-length suffix $n^{(i)}$ of n. The resulting algorithm roughly recovers the performance of the original Montgomery ladder in Algorithm 4.3. See Algorithm 4.4 and Fig. 4.3.

Algorithm 4.4 Three-point ladder of Faz-Hernández et al. [Faz+18]

Input: an elliptic curve $E, P, Q, Q - P \in E$, $n = \sum_{i=0}^{l-1} n_i 2^i \in \{0, \ldots, 2^l - 1\}$
Output: $P + [n]Q$
Notation: $n_{[a:b]} = \sum_{i=a}^{b} n_i 2^{i-a}$

1: $R_1 \leftarrow P$
2: $R_2 \leftarrow Q - P$
3: $S \leftarrow Q$
4: **for** i **from** $l - 1$ **down to** 0 **do**
5: **if** $n_i = 1$ **then**
6: $R_1 \leftarrow R_1 + S$
7: **else**
8: $R_2 \leftarrow R_2 + S$
9: **end if**
10: $S \leftarrow [2]S$ ▷ skip in the last step
11: **end for**
12: **return** R_1

4.3.4 Implementation subtleties

Ladders achieve their performance by leveraging fast pseudo-operations. It is known from Section 3.2 that the pseudo-addition and pseudo-doubling formulas are not applicable to every combination of points. We infer from the preconditions of Algorithms 3.9 and 3.10 that on Montgomery curves, for the given formulas to yield the mathematically correct result it is sufficient that all input points and results of the pseudo-doubling and pseudo-addition operations are nonzero and the difference of differentially added points is outside the 2-torsion.

Although this condition is not automatically true for the three-point ladder of Algorithm 4.4, it is certainly true in the situation of SIDH, since P and Q form a basis of $E[n_X]$.

4.4 Step 2: Computation of the isogeny

Feo, Jao, and Plût [FJP11] proposed the central optimizations in step 2, such as isogeny decomposition and the precomputation of optimal strategies. We use a different notation for the sake of consistency with the rest of the work and for mathematical clarity.

4.4.1 Situation and goals

In step 2 we are given a point R on a supersingular elliptic curve E defined over $\mathbb{F}_{p^2}$. The point R is the result of step 1 and its order is ${\ell_X}^{e_X}$.

The goal is to obtain a representation of the separable isogeny $\varphi\colon E \to E'$ with kernel $\langle R\rangle$ and a representation of the codomain curve E'.

4.4.2 Decomposing isogenies

Motivation While one might be tempted to simply apply Vélu's formulas on $\langle R\rangle$, this approach has a very bad performance. The computational cost of Vélu's formulas is linear in the size of $\langle R\rangle$ and $\#\langle R\rangle = {\ell_X}^{e_X}$ is exponential in $\log p$. For example, in `SIKEp434`, Alice is confronted with a kernel point of order 2^{216}.

Note, however, that R is special. The order of R is smooth, which means in this case that R is a power of a small number ℓ_X that is independent of the number of bits of security. Observe that by the fundamental theorem on group homomorphisms, if $\varphi\colon E \to E'$ is the separable isogeny with kernel R and $\varphi_1\colon E \to E''$ is a separable isogeny whose kernel is a subgroup of $\langle R\rangle$, there is a group homomorphism $\varphi_2\colon E'' \to E'$ such that $\varphi = \varphi_2 \circ \varphi_1$.

Decomposition theorem This important observation allows us to decompose the requested isogeny into smaller ones.

Theorem 4.4 (Decomposition of smooth isogenies) Let E_0 be an elliptic curve and R a point of order $\ell^e > 1$ on E_0. Define inductively for $i = 1, \ldots, e$,

$$R_i := [\ell]^{e-i}(\varphi_{i-1} \circ \ldots \circ \varphi_1(R)),$$

and choose an elliptic curve E_i and a separable isogeny $\varphi_i := E_{i-1} \to E_i$ such that $\ker \varphi_i = \langle R_i\rangle$.

Then $\varphi := \varphi_e \circ \ldots \circ \varphi_1\colon E_0 \to E_e$ is a separable isogeny with $\ker \varphi = \langle R\rangle$ that is composed of ℓ-degree isogenies. ◇

Proof The composition of separable isogenies is a separable isogeny. The proof uses induction according to $e \in \mathbb{N}$.

Let $e = 1$. Then having $R_1 = R$ and $\varphi = \varphi_1$, the claim is obvious.

If the claim holds for $e \in \mathbb{N}$, it also holds for $e+1$. To see this, note that $R' \coloneqq \varphi_1(R)$ is a point of order ℓ^e:

$$\varphi_1([\ell^e]R) = \varphi_1([\ell^e](R)) = 0, \quad \varphi_1([\ell^{e-1}]R) = \varphi_1([\ell^{e-1}](R)) \neq 0,$$

as follows from the requirements to φ_1. Set for $i = 1, \ldots, e$,

$$R'_i \coloneqq R_{i+1} = [\ell^{(e+1)-(i+1)}](\varphi_i \circ \ldots \circ \varphi_1(R)) = [\ell^{e-i}](\varphi'_{i-1} \circ \ldots \circ \varphi'_1(R')), \quad \varphi'_i \coloneqq \varphi_{i+1}.$$

We have $\ker \varphi'_i = \langle R_i \rangle$ for all such i. By the induction hypothesis, $\varphi' \coloneqq \varphi'_e \circ \ldots \circ \varphi'_1 \coloneqq E_2 \to E_{e+1}$ is a separable isogeny with kernel $\langle R' \rangle$.

Now, $\varphi = \varphi' \circ \varphi_1$ satisfies

$$\ker \varphi = \ker(\varphi' \circ \varphi_1) = \varphi_1^{-1}((\varphi')^{-1}(O)) = \varphi_1^{-1}(\langle \varphi_1(R) \rangle) = \langle R \rangle + \langle [\ell^e]R \rangle = \langle R \rangle .$$

This concludes the proof. ∎

Relation to small-degree isogenies If ${\ell_A}^{e_A}$ is a power of 4 and ${\ell_B}^{e_B}$ is a power of 3, Theorem 4.4 helps us to represent an isogeny with a cyclic kernel of order ${\ell_A}^{e_A}$ or ${\ell_B}^{e_B}$ as a composition of cyclic 3- or 4-isogenes, so that we can apply the algorithms that have been developed in Section 3.2.5. See Section 4.4.7 for an explanation how to tackle the caveats regarding 4-isogenies.

Convention For reasons of simplicity, we write φ_* instead of $\varphi_1, \ldots, \varphi_e$. It should be clear from the context which domain the isogeny is supposed to have, which determines which isogeny is meant. For example, this notation enables us to simply write

$$R_i = [\ell]^{e-i} \circ \varphi_*^{i-1}(R) \quad \text{instead of} \quad R_i = [\ell]^{e-i}(\varphi_{i-1} \circ \ldots \circ \varphi_1(R)).$$

4.4.3 Modeling isogeny computation strategies

SIDH never requires an explicit representation of φ. Its decomposition $(\varphi_1, \ldots, \varphi_e)$ is sufficient to evaluate φ at a point $P \in E_0$. The isogeny φ_i can be computed from R_i using Vélu's formulas

or the optimized formulas from Section 3.2. R_i, in turn, is given by

$$R_i = [\ell]^{e-1} \circ \varphi_*^{i-1}(R).$$

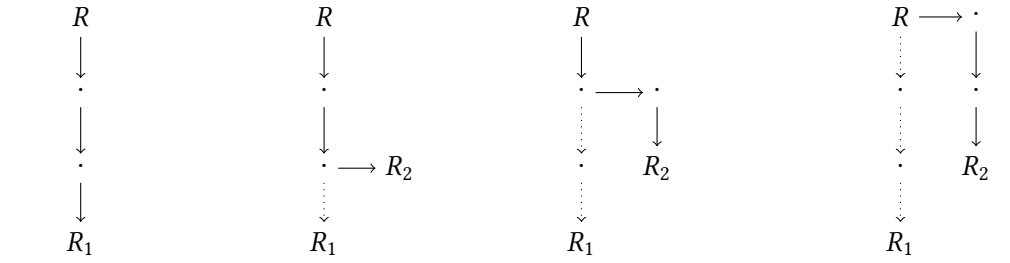
So $\varphi_1, \ldots, \varphi_{i-1}$ must have been computed before it is possible to compute R_i and φ_i using $e - i$ applications of $[\ell]$ and $i - 1$ applications of φ_*.

The order of these isogeny applications is irrelevant here because $[\ell]$ commutes with $\varphi_1, \ldots, \varphi_e$.

Example 4.5 In the case $e = 4$, there is only one order of application that yields R_1 but there are three ways to compute

$$R_2 = [\ell]^2 \circ \varphi_1(R) = [\ell] \circ \varphi_1 \circ [\ell](R) = \varphi_1 \circ [\ell]^2(R).$$

Fig. 4.4 visualizes them as paths from R to R_1 and R_2. Note that different paths to R_2 lead to four,



(a) $R_1 = [\ell]^3(R)$ (b) $R_2 = \varphi_* \circ [\ell]^2(R)$ (c) $R_2 = [\ell] \circ \varphi_* \circ [\ell](R)$ (d) $R_2 = [\ell]^2 \circ \varphi_*(R)$

Figure 4.4: Possible orders of application of $[\ell]$ (vertical arrows) and φ_* (horizontal arrows) in the case $e = 4$ for the computation of R_1 and R_2. Dotted arrows are the applications required in order to compute φ_1.

five, or six required isogeny evaluations due to different opportunities for sharing common intermediate results. ◇

Choosing different orders of application, that is, different paths from R to $R_1, \ldots, R_e$, this leads to the definition of a strategy.

Definition 4.6 (Strategy) Let $e \in \mathbb{N}$. The directed acyclic graph $\mathbf{G}_e$ consists of the vertices

$$\mathbf{V}_e = \{(i, j) \in \mathbb{Z}^2 \mid i, j \geq 0 \wedge i + j \leq e - 1\}$$

and edges

$$\mathbf{E}_e = \{\, ((i,j),(i+1,j)),\ ((i,j),(i,j+1)) \mid \text{source and target are vertices in } \mathbf{V}_e \,\}.$$

The initial element is $v_0 \coloneqq (0,0)$ and the terminal elements are $v_1 \coloneqq (e-1,0), v_2 \coloneqq (e-2,1), \ldots, v_e \coloneqq (0,e-1)$.

An *e-step strategy* is a subgraph σ of G_e that consists of paths from v_0 to each of $v_1, \ldots, v_e$. We write $|\sigma| = e$. Its *cost* is the number of edges of σ and is denoted by (σ). The strategy σ is called *Pareto-optimal* if there is no proper subgraph of σ that is also a strategy. It is called *optimal* if there is no e-step strategy with a smaller cost. ◇

The idea behind this definition is that nodes represent intermediate results in a computation of $\varphi_1, \ldots, \varphi_e$. The node $(0,0)$ corresponds to the point R, v_i corresponds to R_i and more generally, (i,j) corresponds to the point $[\ell]^i \circ \varphi_*^j(R)$. The edges represent isogeny applications.

Example 4.7 The paths in Example 4.5 can be reinterpreted as paths from v_0 to v_1 and v_2 in the case $e = 4$. However, none of those subgraphs is a strategy because paths to v_3 and v_4 are missing. ◇

Obviously, any optimal strategy is Pareto-optimal. Pareto-optimal strategies have a useful characterization.

Lemma 4.8 (Characterization of Pareto-optimal strategies) Let $e \in \mathbb{N}$. Pareto-optimal e-step strategies are characterized as the subtrees of G_e whose leaves are exactly $v_1, \ldots, v_e$.◇

Proof Let σ be a Pareto-optimal e-step strategy. Then σ has a spanning tree σ'. Combining the unique paths from v_0 to $v_1, \ldots, v_e$ inside σ', we obtain a strategy σ'' that is a subtree of σ. Because σ is Pareto-optimal, $\sigma = \sigma''$ is a tree whose leaves are exactly $v_1, \ldots, v_e$.

Conversely, assume that σ is a subtree of G_e whose leaves are exactly $v_1, \ldots, v_e$. Because v_0 is the only common ancestor of the leaves, v_0 must be the root of σ. There is a unique path from v_0 to each leaf. Every node w contained in σ is contained in such a path: There is a path from the root to w and a path from w to a leaf. So σ is an e-step strategy. Because the paths from the root to the leaves are unique, there is no proper substrategy and σ must be Pareto-optimal. ∎

4.4.4 Computing isogenies using strategies

The following lemma formally shows that strategies can be used to compute isogenies of degree ℓ^e.

Lemma 4.9 (Strategies induce isogeny computation algorithms) Any e-step strategy σ gives rise to an algorithm that computes the decomposition of an ℓ^e-degree isogeny requiring exactly (σ) isogeny applications and e image curve parameter computations from Section 3.2.5. The applied isogenies are $[\ell]$ and $\varphi_1, \ldots, \varphi_e$. ◇

Proof The algorithm proceeds in e steps, the result of the i-th step being an explicit formula for φ_i.

The nodes of σ are interpreted as intermediate results, the node (i, j) corresponding to the point $[\ell]^i \circ \varphi_*^j(R)$. An edge is interpreted as an isogeny that maps its source to its target. Vertical edges $(i, j) \longrightarrow (i+1, j)$ correspond to $[\ell]$, while horizontal edges $(i, j) \longrightarrow (i, j+1)$ correspond to φ_{j+1}. By saying that we evaluate an edge, we mean that we compute its target from its source. This requires that the source has been computed before.

In step 1, we evaluate all vertical edges on the unique path from v_0 to v_1. Since v_1 corresponds to R_1, we have now computed R_1. The point R_1 generates the kernel of φ_1, and we compute the parameters of the image curve. No we are able to evaluate φ_1 on the domain curve and $[\ell]$ on the image curve.

In step $k \geq 2$, we assume that $\varphi_1, \ldots, \varphi_{k-1}$ and all intermediary results corresponding to nodes (i, j) of σ with $i < k-1$ are available from earlier steps. Then we evaluate all horizontal edges corresponding to φ_{k-1} and then all following vertical edges in top-down order (these edges belong to the column indexed by $k-1$). Since there must be a path from v_0 to v_k in the strategy, we now know R_k and can compute the image curve parameters of φ_k and evaluate φ_k when necessary.

After the e-th step, we have evaluated every edge exactly once in a left-to-right, top-down manner. Hence, we applied exactly (σ) isogenies to intermediate points. Every step ended with an image curve parameter computation, in total e times. ∎

4.4.5 Fast and simple strategies

We will analyze optimal strategies later. However, it is easier to find potentially suboptimal, though reasonably fast, ones using a divide-and-conquer approach.

The central observation is that two strategies can be composed to a larger strategy.

Lemma 4.10 (Composition of strategies) Let σ_l, σ_r be strategies with $|\sigma_l| = e_l, |\sigma_r| = e_r$. Then they give rise to a strategy $\sigma = \sigma_l \boxplus \sigma_r$ with $|\sigma| = e_l + e_r$ and $e = \ell(\sigma) = e_l + e_r + \ell(\sigma_l) + \ell(\sigma_r)$.$\diamond$

Proof We define σ as the subgraph of G_e that consists of the following parts:

- the unique path from $(0,0)$ to $(e_r, 0)$ together with a copy of σ_l along the graph inclusion

$$G_{e_l} \to G_e, \quad (i,j) \mapsto (i + e_r, j),$$

- and the unique path from $(0,0)$ to $(0, e_l)$ together with a copy of σ_r along the graph inclusion

$$G_{e_r} \to G_e, \quad (i,j) \mapsto (i, j + e_l).$$

There is a path from v_0 to the roots of the copies of σ_l and σ_r. Furthermore, the leaves of σ_l correspond to the leaves $v_1, \ldots, v_{e_l}$ of σ, while the leaves of σ_r correspond to the leaves $v_{e_l+1}, \ldots, v_{e_l+e_r}$ of σ. Thus, σ is a strategy. One easily counts that the quantitative statements are true. ∎

Theorem 4.11 (Divide-and-conquer strategies) Let $e \in \mathbb{N}$. There is a e-step strategy with $\ell(e) \leq 2(e \log e + e) \in O(e \log e)$. $\diamond$

Proof The strategy is constructed inductively using a divide-and-conquer technique.

By induction according to $e \in \mathbb{N}$, define σ_1 to be the unique 1-step strategy (consisting of a vertex without edges) and for $e \geq 2$, set $\sigma_e := \sigma_{\lfloor e/2 \rfloor} \boxplus \sigma_{\lceil e/2 \rceil}$.

One easily checks using induction that for all $e \in \mathbb{N}$, we have $|S_e| = e$ and $\ell(S_e) \leq \ell(S_{e+1})$.

Next we show inductively that for all $k \in \mathbb{N}_0$, $\ell(S_{2^k}) = k \cdot 2^k$. Clearly, $\ell(S_1) = 0$. Now assume that the claim is true for $k \in \mathbb{N}_0$. Then: $\ell(S_{2^{k+1}}) = 2 \cdot 2^k + 2\ell(S_{2^k}) = 2^{k+1} + 2 \cdot k \cdot 2^k = (k+1) \cdot 2^{k+1}$.

Now consider arbitrary $e \in \mathbb{N}$. Choose a $k \in \mathbb{N}_0$ such that $e \leq 2^k \leq 2e$, and it follows $\ell(S_e) \leq \ell(S_{2^k}) = k \cdot 2^k \leq \log(2e) \cdot (2e) = 2(e \log e + e)$. ∎

4.4.6 Finding optimal strategies

Lemma 4.12 (Decomposability of optimal strategies) Let $e \geq 2$. Every optimal e-step strategy σ is composed of two smaller optimal strategies, i. e., $\sigma = \sigma_l \boxplus \sigma_r$. ◇

Proof Imagine removing v_0 from σ. Then the tree σ breaks down into two smaller trees with roots $v_l := (1, 0)$ and $v_r := (0, 1)$. The leaves of σ that lay below v_l are $v_1, \ldots, v_{e_l}$ and the leaves that lay below v_r are $v_{e_l+1}, \ldots, v_{e_l+e_r}$, where $1 \leq e_l, e_r < e$ and $e_l + e_r = e$.

Let $b_l := \max\{i \mid (i, 0) \text{ is a common ancestor of all leaves below } v_l\}$. We claim that $b_l = e_r$.

The leaves below v_l are $v_1 = (e_r + e_l - 1, 0), v_2 = (e_r + e_l - 2, 1), \ldots, v_{e_l} = (e_r, e_l - 1)$ and they have the common ancestor $(e_r, 0)$.

Thus, the left branch of σ consists of the unique path from v_0 to $(e_r, 0)$ and the subtree below $(e_r, 0)$. The latter is a copy of a subtree σ_l of G_{e_l} along the graph inclusion

$$G_{e_l} \to G_e, \quad (i, j) \mapsto (i + e_r, j).$$

The leaves of σ_l in G_{e_l} are exactly $v_1, \ldots, v_{e_l}$, so σ_l is a Pareto-optimal e_l-step strategy.

Proceeding analogously for the right branch of σ, we conclude that $\sigma = \sigma_l \boxplus \sigma_r$. ∎

Theorem 4.13 (Computation of optimal strategies) Optimal e-step strategies can be computed in quadratic time according to e. ◇

Proof According to Lemma 4.12, we can compose optimal strategies of smaller optimal strategies. Feo, Jao, and Plût [FJP11, Equation (5)] give a dynamic-programming algorithm that computes the optimal e-step strategy in time $O(n^2)$. Refer to their paper for a proof of validity. ∎

4.4.7 Choosing 4-isogeny algorithms

It was noted in Section 3.2.5 that the existence of two different 4-isogeny computation algorithms that are not always applicable is problematic and might lead to overly complex or even side-channel-insecure implementations.

In practice, it is not that bad, at least with carefully chosen public parameters, as the following result shows.

Theorem 4.14 (Applicability of 4-isogeny algorithms) Let E be a Montgomery curve and $P_A, Q_A \in E$ according to the SIDH protocol. Assume that ${\ell_A}^{e_A}$ a power of 4 and ${\ell_B}^{e_B}$ a power of 3 and that $[2^{e_A-1}]Q_A = (0,0)$. Then we distinguish two cases, depending on the secret (a_A, b_A) of Alice.

- If $(a_A, b_A) \equiv (0,1) \pmod 2$, then the decomposed isogeny can be computed by using Algorithms 3.15 and 3.16 for the first isogeny φ_1 and using Algorithms 3.17 and 3.18 for the remaining isogenies.
- If $a_A \equiv 1 \pmod 2$, then the decomposed isogeny can be computed by using always Algorithms 3.17 and 3.18. ◇

Proof We are concerned with the kernel point $R = [a]P_A + [b]Q_A$. It has, by the validity of the secret key, order $n_A = 2^{e_A}$.

If R does not lie above $(0,0)$ then $R_1 = [2^{e_A-2}]R$ does not either. Hence, Algorithms 3.17 and 3.18 can be applied for the first isogeny. In this case, φ_1 maps $(0,0)$ to $(0,0)$.

If R had order 4, we are done. Otherwise, assume for contradiction that $\varphi_1(R)$ lies above $(0,0)$. Then

$$O = [2^{e_A-3}] \circ \varphi_1(R) - \varphi_1((0,0)) = \varphi_1([2^{e_A-3}]R - (0,0)).$$

But $[2^{e_A-3}]R - (0,0)$ is a point of order 8, contradicting the fact that the kernel of φ_1 has order 4. So $\varphi_1(R)$ does not lie over $(0,0)$ either and we see by induction that φ is found by applying the same algorithms again and again.

If R does lie above $(0,0)$ then R_1 also does. Hence, Algorithms 3.15 and 3.16 can be applied for the first isogeny.

Again, if R had order 4, we are done. Otherwise, $\varphi_1(R)$ does not lie above $(0,0)$: Assume for contradiction that it does. Observing that $\varphi_1((-1,\ldots)) = (0,0)$, we obtain

$$O = [2^{e_A-3}] \circ \varphi_1(R) - \varphi_1((-1,\ldots)) = \varphi_1([2^{e_A-3}R] - (-1,\ldots)).$$

But $[2^{e_A-3}R] - (-1,\ldots)$ is a point of order 8, contradicting the fact that the kernel of φ_1 has order 4. So R does not lie over $(0,0)$. As we showed in the first case, we can now repeatedly use Algorithms 3.17 and 3.18 to obtain φ.

To conclude the proof, note if $(a_A, b_A) \equiv (0,1) \pmod 2$, then $[2^{e_A-1}a_A]P_A$ is zero. We obtain

that R lies over $(0,0)$ because

$$[2^{e_A-1}]R = \mathcal{O} + [b_A]\,(0,0) = (0,0)\,.$$

If, on the other hand, $a_A \equiv 1 \pmod 2$, observe that $[2^{e_A-1}]P_A$ is a point of order two but not in the group $\langle(0,0)\rangle$, while $[2^{e_A-1}]Q_A$ is. Hence,

$$[2^{e_A-1}]R = [2^{e_A-1}]P_A + [2^{e_A-1}b_A]Q_A$$

cannot be $(0,0)$, and R does not lie above $(0,0)$. ∎

Hence, as Renes [Ren18] remarked, by reducing the key space by $1/3$ or $2/3$ and hardcoding one of the two cases, the choice of algorithms becomes predictable. The NIST-submitted specification guarantees that we are in the latter case, while Costello, Longa, and Naehrig [CLN16] had previously relied on the former case.

Remark 4.15 Note that this theorem also applies in the second phase of the protocol if it did in the first. The reason is that the isogeny φ_B maps $(0,0)$ to $(0,0)$ and is injective on the 2^{e_A}-torsion so that $\varphi_B(Q_A)$ fulfils the requirements of the theorem if Q_A does. ◇

4.5 Conclusion

We have analyzed the most performance-critical parts of the SIDH protocol. Asymptotically, all of them have been addressed by asymptotically efficient algorithms. Although it is always hard in a theoretical work to talk about the constant factors, these factors have been shrinked significantly, especially because of the efficiency of curve arithmetic on the Kummer line. The practical results of Azarderakhsh et al. [Aza+19] and Seo, Jalali, and Azarderakhsh [SJA19] suggest that an optimized implementation of SIDH is fast enough to be used for everyday encryption.

Another topic I find very interesting, however out of scope, is public key compression. One of the advantages of SIDH and SIKE are their small public keys. There has been a line of work on how to compress these keys even more without trading too much of the performance. If you are interested in this topic, some relevant papers are those of Azarderakhsh et al. [Aza+16], Costello et al. [Cos+17], and Naehrig and Renes [NR19].

Bibliography

[Aza+16] Reza Azarderakhsh et al. “Key Compression for Isogeny-Based Cryptosystems.” In: May 2016, pp. 1–10.

[Aza+19] Reza Azarderakhsh et al. “Supersingular Isogeny Diffie-Hellman Key Exchange on 64-Bit ARM.” In: *IEEE Trans. Dependable Sec. Comput.* 16.5 (2019), pp. 902–912. URL: https://doi.org/10.1109/TDSC.2017.2723891.

[Bar86] Paul Barrett. “Implementing the Rivest Shamir and Adleman Public Key Encryption Algorithm on a Standard Digital Signal Processor.” In: *Advances in Cryptology - CRYPTO ’86, Santa Barbara, California, USA, 1986, Proceedings*. Ed. by Andrew M. Odlyzko. Vol. 263. Lecture Notes in Computer Science. Springer, 1986, pp. 311–323. URL: https://doi.org/10.1007/3-540-47721-7%5C_24.

[Ber+08] Daniel Bernstein et al. “Twisted Edwards Curves.” In: vol. 5023. June 2008, pp. 389–405.

[Ber06a] Daniel J. Bernstein. *Can we avoid tests for zero in fast elliptic-curve arithmetic?* 2006. URL: https://cr.yp.to/ecdh/curvezero-20060721.pdf.

[Ber06b] Daniel J. Bernstein. “Curve25519: New Diffie-Hellman Speed Records.” In: *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, April 24-26, 2006, Proceedings*. Ed. by Moti Yung et al. Vol. 3958. Lecture Notes in Computer Science. Springer, 2006, pp. 207–228. URL: https://doi.org/10.1007/11745853%5C_14.

[BJ02] Eric Brier and Marc Joye. “Weierstraß Elliptic Curves and Side-Channel Attacks.” In: *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*. Ed. by David Naccache and Pascal Paillier. Vol. 2274. Lecture Notes in Computer Science. Springer, 2002, pp. 335–345. URL: https://doi.org/10.1007/3-540-45664-3%5C_24.

[BL07] Daniel Bernstein and Tanja Lange. 2007. URL: https://www.hyperelliptic.org/EFD.

[Cas+18] Wouter Castryck et al. "CSIDH: An Efficient Post-Quantum Commutative Group Action." In: *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*. Ed. by Thomas Peyrin and Steven D. Galbraith. Vol. 11274. Lecture Notes in Computer Science. Springer, 2018, pp. 395–427. URL: https://doi.org/10.1007/978-3-030-03332-3%5C_15.

[CGF08] Wouter Castryck, Steven D. Galbraith, and Reza Rezaeian Farashahi. "Efficient arithmetic on elliptic curves using a mixed Edwards-Montgomery representation." In: *IACR Cryptol. ePrint Arch.* 2008 (2008), p. 218. URL: http://eprint.iacr.org/2008/218.

[CH17] Craig Costello and Hüseyin Hisil. "A simple and compact algorithm for SIDH with arbitrary degree isogenies." In: *IACR Cryptol. ePrint Arch.* 2017 (2017), p. 504. URL: http://eprint.iacr.org/2017/504.

[CJS10] Andrew M. Childs, David Jao, and Vladimir Soukharev. "Constructing elliptic curve isogenies in quantum subexponential time." In: *CoRR* abs/1012.4019 (2010). arXiv: 1012.4019. URL: http://arxiv.org/abs/1012.4019.

[CK01] Ran Canetti and Hugo Krawczyk. "Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels." In: *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*. Ed. by Birgit Pfitzmann. Vol. 2045. Lecture Notes in Computer Science. Springer, 2001, pp. 453–474. URL: https://doi.org/10.1007/3-540-44987-6%5C_28.

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig. "Efficient Algorithms for Supersingular Isogeny Diffie-Hellman." In: Aug. 2016.

[CMO98] Henri Cohen, Atsuko Miyaji, and Takatoshi Ono. "Efficient Elliptic Curve Exponentiation Using Mixed Coordinates." In: *Advances in Cryptology - ASIACRYPT '98, International Conference on the Theory and Applications of Cryptology and Information Security, Beijing, China, October 18-22, 1998, Proceedings*. Ed. by Kazuo Ohta and Dingyi Pei. Vol. 1514. Lecture Notes in Computer Science. Springer, 1998, pp. 51–65. URL: https://doi.org/10.1007/3-540-49649-1%5C_6.

[Cos+17] Craig Costello et al. "Efficient Compression of SIDH Public Keys." In: *Advances in Cryptology – EUROCRYPT 2017*. Ed. by Jean-Sébastien Coron and Jesper Buus Nielsen. Cham: Springer International Publishing, 2017, pp. 679–706.

[Cou06] Jean Marc Couveignes. "Hard Homogeneous Spaces." In: *IACR Cryptol. ePrint Arch.* 2006 (2006), p. 291. URL: http://eprint.iacr.org/2006/291.

[Faz+18] Armando Faz-Hernández et al. "A Faster Software Implementation of the Supersingular Isogeny Diffie-Hellman Key Exchange Protocol." In: *IEEE Trans. Computers* 67.11 (2018), pp. 1622–1636. URL: https://doi.org/10.1109/TC.2017.2771535.

[FJP11] Luca De Feo, David Jao, and Jérôme Plût. *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies*. Cryptology ePrint Archive, Report 2011/506. https://eprint.iacr.org/2011/506. 2011.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. "A Modular Analysis of the Fujisaki-Okamoto Transformation." In: *IACR Cryptol. ePrint Arch.* 2017 (2017), p. 604. URL: http://eprint.iacr.org/2017/604.

[IT02] Tetsuya Izu and Tsuyoshi Takagi. "A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks." In: *Public Key Cryptography, 5th International Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2002, Paris, France, February 12-14, 2002, Proceedings*. Ed. by David Naccache and Pascal Paillier. Vol. 2274. Lecture Notes in Computer Science. Springer, 2002, pp. 280–296. URL: https://doi.org/10.1007/3-540-45664-3%5C_20.

[Jao+20] David Jao et al. *Supersingular Isogeny Key Encapsulation*. Specification of NIST submission. Apr. 2020. URL: https://sike.org/files/SIDH-spec.pdf (visited on 06/23/2020).

[JL09] Benjamin Justus and Daniel Loebenberger. "Differential Addition in generalized Edwards Coordinates." In: *IACR Cryptol. ePrint Arch.* 2009 (2009), p. 523. URL: http://eprint.iacr.org/2009/523.

[Knu81] Donald E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms, 2nd Edition*. Addison-Wesley, 1981.

[KO62] Anatolii Karatsuba and Y. Ofman. "Multiplication of Many-Digital Numbers by Automatic Computers." In: 145 (Jan. 1962).

[Mon85] Peter L Montgomery. "Modular multiplication without trial division." In: *Mathematics of computation* 44.170 (1985), pp. 519–521.

[Mon87] Peter L. Montgomery. "Speeding the Pollard and elliptic curve methods of factorization." In: 1987.

[NR19] Michael Naehrig and Joost Renes. *Dual Isogenies and Their Application to Public-key Compression for Isogeny-based Cryptography*. Cryptology ePrint Archive, Report 2019/499. https://eprint.iacr.org/2019/499. 2019.

[OS01] Katsuyuki Okeya and Kouichi Sakurai. "Efficient Elliptic Curve Cryptosystems from a Scalar Multiplication Algorithm with Recovery of the y-Coordinate on a Montgomery-Form Elliptic Curve." In: vol. 2162. May 2001, pp. 126–141.

[Ren18] Joost Renes. "Computing Isogenies Between Montgomery Curves Using the Action of (0, 0)." In: *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*. Ed. by Tanja Lange and Rainer Steinwandt. Vol. 10786. Lecture Notes in Computer Science. Springer, 2018, pp. 229–247. URL: https://doi.org/10.1007/978-3-319-79063-3%5C_11.

[RS06] Alexander Rostovtsev and Anton Stolbunov. "Public-Key Cryptosystem Based on Isogenies." In: *IACR Cryptol. ePrint Arch.* 2006 (2006), p. 145. URL: http://eprint.iacr.org/2006/145.

[Sch87] René Schoof. "Nonsingular plane cubic curves over finite fields." In: *J. Comb. Theory, Ser. A* 46.2 (1987), pp. 183–211. URL: https://doi.org/10.1016/0097-3165(87)90003-3.

[Sho04] Victor Shoup. "Sequences of games: a tool for taming complexity in security proofs." In: *IACR Cryptol. ePrint Arch.* 2004 (2004), p. 332. URL: http://eprint.iacr.org/2004/332.

[Sil09] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. New York: Springer-Verlag, 2009.

[SJA19] Hwajeong Seo, Amir Jalali, and Reza Azarderakhsh. *Optimized SIKE Round 2 on 64-bit ARM*. Cryptology ePrint Archive, Report 2019/721. https://eprint.iacr.org/2019/721. 2019.

[Tat66] John Tate. "Endomorphisms of abelian varieties over finite fields." In: Inventiones mathematicae 2 (Apr. 1966), pp. 134–144.

[Vél71] Jaques Vélu. "Isogenies entre courbes elliptiques." In: *C. R. Acad. Sci. Paris Sér. A-B* 273 (Jan. 1971).

List of Tables

List of Figures

List of Definitions and Theorems

Definitions

Theorems