A Faster Software Implementation of SQISign

Kaizhan Lin[1], Weize Wang[1], Zheng Xu✉[2,3], and Chang-An Zhao[1,4]

[1] School of Mathematics, Sun Yat-sen University, Guangzhou, China
linkzh5@mail2.sysu.edu.cn
wangwz@mail2.sysu.edu.cn
zhaochan3@mail.sysu.edu.cn

[2] Ding Lab, Yanqi Lake Beijing Institute of Mathematical Sciences and Applications, Beijing, China
xuzheng@bimsa.cn

[3] Yau Mathematical Sciences Center, Tsinghua University, Beijing, China

[4] Guangdong Key Laboratory of Information Security, Guangzhou, China

Abstract. Isogeny-based cryptography is famous for its short key size. As one of the most compact digital signatures, SQISign (Short Quaternion and Isogeny Signature) is attractive among post-quantum cryptography, but it is inefficient compared to other post-quantum competitors because of complicated procedures in ideal to isogeny translation, which is the efficiency bottleneck of the signing phase.

In this paper, we recall the current implementation of SQISign and mainly discuss how to improve the execution of ideal to isogeny translation in SQISign. To be precise, we modify the SigningKLPT algorithm to accelerate the performance of generating the ideal I_σ. In addition, we explore how to save one of the two elliptic curve discrete logarithms and compute the remainder with the help of the reduced Tate pairing correctly and efficiently. We speed up other procedures in ideal to isogeny translation with various techniques as well. It should be noted that our improvements also benefit the performances of key generation and verification in SQISign. In particular, in the instantiation with p_{3923} the improvements lead to a speedup of 8.82%, 8.50% and 18.94% for key generation, signature and verification, respectively.

Keywords: Isogeny-based Cryptography · SQISign · Pairings · Discrete Logarithms

1 Introduction

Among post-quantum cryptography, isogeny-based cryptography is famous for its short key size. In the last two decades, various isogeny-based key exchange schemes were proposed, such as SIDH [24], CSIDH [8] and OSIDH [10,31]. These protocols also motivate cryptographers to construct digital signatures. Until now, there are mainly three kinds of isogeny-based signatures: SIDH-based [14,9], CSIDH-based [15,6,1], and quaternion-based [22,16,17,13].

Authors are listed in alphabetical order.

SQISign (Short Quaternion and Isogeny Signature) was first proposed by De Feo, Kohel, Leroux, Petit and Wesolowski [16]. Compared with other isogeny-based signatures, the bitlength of the prime field characteristic used in SQISign is relatively small. Besides, the public key of SQISign does not reveal torsion point information (and thus it is not vulnerable to the Castryck-Decru-Maino-Martindale-Robert attacks [7,29,35]). Furthermore, the signer needs to respond for each challenge bit respectively in most of isogeny-based signatures, but there is no need for such a procedure in SQISign. Therefore, as one of the most compact signatures, SQISign is competitive in post-quantum cryptography.

SQISign is obtained by applying the Fiat–Shamir transform [19] to an identification protocol. The signing phase of SQISign mainly involves two procedures: ideal generation with the SigningKLPT algorithm and ideal to isogeny translation.

The SigningKLPT algorithm is based on the KLPT algorithm, which was first proposed by Kohel et al. [25] in 2014. Given a left ideal I, the SigningKLPT algorithm outputs another left ideal J of a smooth power reduced norm which is equivalent to I. After obtaining J, the signer needs to translate it into the corresponding isogeny φ_J and compress it to be a part of the signature. The translation from the ideal J to the isogeny φ_J is the efficiency bottleneck of SQISign, since it involves expensive procedures such as large degree isogeny computations. Recently, De Feo et al. [17] proposed a novel approach to speed up the performance of ideal to isogeny translation. Besides isogeny computations, the current implementation contains torsion point generation, discrete logarithm computations, etc.

In this paper, we explore the current SQISign implementation and further accelerate it by utilizing several techniques, especially the performance of the ideal to isogeny translation procedure, as we summarize in the following:

1. In the SigningKLPT algorithm, the output I_σ is required to be an ideal of a fixed reduced norm, which corresponds to a cyclic isogeny σ. Besides, the composition $\sigma \circ \varphi_{I_2}$ should also be cyclic, where φ_{I_2} is the secret isogeny from E_0 to E_A of degree $2^\bullet$. To achieve this goal, one may repeat the SigningKLPT algorithm with high probability. We improve this procedure by giving a modified SigningKLPT algorithm, which generates I_σ such that $\sigma \circ \varphi_{I_2}$ is always cyclic. Heuristically, we save about one third of the computational cost of generating the required I_σ.
2. In [17], each step of the new algorithm for ideal to isogeny translation requires computing two elliptic curve discrete logarithms, i.e.,

$$\theta(P) = [x_1]P + [x_2]Q,$$
$$\theta(Q) = [x_3]P + [x_4]Q,$$

   where $P, Q \in E[2^a]$, and the endomorphism θ is of reduced norm coprime to 2. For efficiency, the previous work used an x-only arithmetic to obtain the absolute values of x_1, x_2, x_3 and x_4, then employed the reduced trace of the endomorphism to confirm the signs of them. We claim that one can

avoid the second elliptic curve discrete logarithm computation by making full advantage of the properties of θ. We also provide a much more efficient approach to solve the other elliptic curve discrete logarithm by utilizing pairing computations and discrete logarithm computations over the finite field $\mathbb{F}_{p^2}$. The experimental results show that our method leads to 4.8× faster execution time in the implementation with p_{3923}. It should be noted that the improvement benefits not only signature but also key generation.
3. We propose new techniques to optimize other procedures in ideal to isogeny translation. In particular, our algorithm offers a speedup of about 1.5× to torsion point generation. Besides, we improve the performance of isogeny computations in SQISign, leading to a considerable improvement. We also show that one can accelerate the first step of ideal to isogeny translation in the signing phase via precomputation in the key generation phase. It may enlarge the cost of key generation, but reduces the signing cost. This would be preferred when the signer wants to sign a number of messages with the same secret key.
4. Based on the code presented in [17], we complied and benchmarked our code. The experimental results show that our techniques yield a significant acceleration of all the above procedures. Besides, we not only improve the signing phase but also key generation and verification of SQISign: the instantiation with p_{3923} of key generation, signature and verification with our techniques are 8.82%, 8.50% and 18.94% faster than those of the state-of-the-art, respectively. In particular, when the precomputation technique in the key generation phase is adapted, the performance of the signing phase can be up to 11.93% faster than that of the previous work.

The remainder of this paper is organized as follows. In Section 2 we explain some mathematical concepts and review SQISign, especially ideal to isogeny translation. In Section 3 we propose the modified SigningKLPT algorithm to accelerate the implementation of SQISign. Section 4 presents an efficient approach to compute discrete logarithms in ideal to isogeny translation. In Section 5 we give other improvements to speed up the performance. Finally, we report experimental results and give a performance comparison between ours and the previous work in Section 6 and conclude in Section 7.

2 Notations and Preliminaries

In this section, we provide the required background that will be used throughout the paper. We also recap SQISign and the implementation of ideal to isogeny translation.

2.1 Mathematical background

In this subsection we recall supersingular elliptic curves, isogenies and ideals in quaternion algebras, for more in-deep details see [39,37].

Elliptic curves and isogenies Elliptic curves are nonsingular projective curves with genus 1. We denote the infinity point of an elliptic curve E as ∞_E. An isogeny $\varphi : E_1 \to E_2$ is a non-constant morphism, which sends ∞_{E_1} to ∞_{E_2}. If the degree of isogeny φ is equal to the size of $\ker(\varphi)$, we call φ a separable isogeny. We abbreviate a separable isogeny of degree ℓ as ℓ-isogeny. For any subgroup G of an elliptic curve E, we can compute an isogeny with kernel G by Vélu's formula [38,4]. For any isogeny φ from E_1 to E_2, there exists a unique isogeny $\hat{\varphi}$ from E_2 to E_1 such that $\hat{\varphi} \circ \varphi = \varphi \circ \hat{\varphi} = [\deg(\varphi)]$. We call $\hat{\varphi}$ the dual isogeny of φ.

An endomorphism is an isogeny from E to itself. The set of endomorphisms forms a ring under addition and composition. We call the ring endomorphism ring, denoted by $\mathrm{End}(E)$. Since the scalar multiplication $[n]$ is an isogeny, we have $\mathbb{Z} \subseteq \mathrm{End}(E)$. Moreover, if $\mathrm{End}(E) \neq \mathbb{Z}$, we say that E has complex multiplication.

Each of elliptic curves over finite fields has complex multiplication, and they can be divided into two types by endomorphism rings. The curve E is said to be ordinary if $\mathrm{End}(E)$ is isomorphic to an order in a quadratic imaginary field. Otherwise, the elliptic curve E is said to be supersingular if $\mathrm{End}(E)$ is isomorphic to a maximal order in a quaternion algebra.

Orders and ideals in quaternion algebra A quaternion algebra over $\mathbb{Q}$ ramified only at p and ∞ is of the form $B_{p,\infty} = \mathbb{Q} + \mathbb{Q}i + \mathbb{Q}j + \mathbb{Q}k$, where $i^2 = -1$, $j^2 = -p$ and $k = ij = -ji$. For any $\alpha = a_1 + a_2 i + a_3 j + a_4 k \in B_{p,\infty}$, the canonical involution is the map sending α to $\bar{\alpha} = a_1 - a_2 i - a_3 j - a_4 k$. The reduced trace and the reduced norm of α are respectively defined by

$$\begin{aligned}\mathrm{Trd}(\alpha) &= \alpha + \bar{\alpha} = 2a_1,\\ \mathrm{Nrd}(\alpha) &= \alpha\bar{\alpha} = {a_1}^2 + {a_2}^2 + p{a_3}^2 + p{a_4}^2.\end{aligned}$$

An order in $B_{p,\infty}$ is a full-rank lattice and it is also a subring. A maximal order is an order which is not contained in any other order. The endomorphism rings of supersingular elliptic curves over $\overline{\mathbb{F}_p}$ are isomorphic to maximal orders in $B_{p,\infty}$. Let $\mathcal{O}$ be a maximal order. A full-rank lattice $I \subseteq \mathcal{O}$ is a left $\mathcal{O}$-ideal if $\mathcal{O}I \subseteq I$, and it is a right $\mathcal{O}$-ideal if $I\mathcal{O} \subseteq I$. For any left ideal I of a maximal order $\mathcal{O}$ in $B_{p,\infty}$, define the left order and right order of I as

$$\mathcal{O}_L(I) = \{x \in B_{p,\infty} \mid xI \subseteq I\}, \quad \mathcal{O}_R(I) = \{x \in B_{p,\infty} \mid Ix \subseteq I\}.$$

Note that $\mathcal{O}_L(I)$ and $\mathcal{O}_R(I)$ are also maximal orders. We say that I connects $\mathcal{O}_L(I)$ and $\mathcal{O}_R(I)$, and the corresponding Eichler order of I is defined as $\mathfrak{O} = \mathcal{O}_L(I) \cap \mathcal{O}_R(I)$. The reduced norm of I can be defined by $\mathrm{Nrd}(I) = \gcd(\{\mathrm{Nrd}(\alpha) \mid \alpha \in I\})$. The conjugate of I, denoted by $\bar{I}$, is the set of conjugates of elements of I satisfying $I\bar{I} = \mathrm{Nrd}(I)\mathcal{O}_L(I)$ and $\bar{I}I = \mathrm{Nrd}(I)\mathcal{O}_R(I)$. Two left ideals I and J in $\mathcal{O}$ are equivalent if there exists $\alpha \in B_{p,\infty}^{\times}$ such that $J = I\alpha$, and we denote the set of such classes by $\mathrm{cl}(\mathcal{O})$.

Isogeny graphs The ℓ-isogeny graph is denoted by $\mathcal{G}_\ell(\overline{\mathbb{F}_p})$. A vertex in this graph is an $\overline{\mathbb{F}_p}$-isomorphism class $[E]$ of supersingular elliptic curves defined over $\overline{\mathbb{F}_p}$, and all the elliptic curves in the $\overline{\mathbb{F}_p}$-isomorphism class have the same j-invariant. Let φ_1 and φ_2 be two isogenies from E_1 to E_2 with degree ℓ. We say that φ_1 and φ_2 are equivalent if $\ker(\varphi_1) = \ker(\varphi_2)$. Then an edge in this graph is an equivalent class of ℓ-isogenies. From [32], the ℓ-isogeny graph $\mathcal{G}_\ell(\overline{\mathbb{F}_p})$ is a Ramanujan graph.

Deuring Correspondence Suppose that E is a supersingular elliptic curve over $\mathbb{F}_{p^2}$, and its endomorphism ring $\mathrm{End}(E)$ is isomorphic to a maximal order of $B_{p,\infty}$, denoted by $\mathcal{O}$.

For a left integral ideal I of $\mathcal{O}$, let $E[I]=\{P \in E \mid \alpha(P) = \infty_E$ for any $\alpha \in I\}$, then the isogeny

$$\varphi_I : E \to E_I = E/E[I]$$

has $\ker(\varphi_I) = E[I]$ and $\deg(\varphi_I) = \mathrm{Nrd}(I)$. On the other hand, if $\varphi : E \to E'$ is an isogeny of degree n, then the cardinality of $\ker(\varphi)$ is n and $I_\varphi = \{\alpha \in \mathcal{O} \mid \alpha(P) = \infty_E \text{ for any } P \in \ker(\varphi)\}$ is a left $\mathcal{O}$-ideal of reduced norm n.

The Deuring Correspondence Theorem gives the connection between isogenies and ideals:

There is a one-to-one correspondence between left $\mathcal{O}$-ideals I of reduced norm n and equivalent classes of isogenies $\varphi : E \to E'$ of degree n given by $I \mapsto [\varphi_I]$ and $[\varphi] \mapsto I_\varphi$. If $\varphi : E \to E'$ and I are corresponding to each other, then $\mathrm{End}(E')$ is isomorphic to the right order of I in $B_{p,\infty}$. Particularly, $\varphi \in \mathrm{End}(E)$ if and only if $I = \mathcal{O}\varphi$ is a principal ideal. Furthermore, suppose that $\varphi_1 : E \to E_1$ and $\varphi_2 : E \to E_2$ are two isogenies corresponding to the left ideals $I_1, I_2 \subseteq \mathcal{O}$, respectively. Then E_1 and E_2 are in the same isomorphism class if and only if I_1 and I_2 are equivalent.

Here we illustrate the endomorphism ring of $E_0 : y^3 = x^3 + x$, which is the starting curve of the SQISign implementation.

Example of endomorphism ring Let $p \equiv 3 \pmod 4$ and $E_0 : y^2 = x^3 + x$ be a supersingular elliptic curve with j-invariant 1728. The endomorphism ring of E is isomorphic to the maximal order $\mathcal{O}_0 = \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}\frac{i+j}{2} + \mathbb{Z}\frac{1+k}{2}$, where $i^2 = -1$, $j^2 = -p$ and $ij = -ji = k$. Indeed, the Frobenius map $\pi : (x, y) \to (x^p, y^p)$ corresponds to j, while the distortion map $\omega : (x, y) \to (-x, iy)$ corresponds to i. By abuse of notation, we sometimes use i and j to represent ω and π, respectively when there is no ambiguity in the context.

2.2 SQISign

SQISign (Short Quaternion and Isogeny Signature) was first introduced by De Feo et al. [16] in 2020 and it is known as a compact post-quantum signature. This signature is based on an identification protocol with Fiat-Shamir transform [19]. The main procedures of the identification protocol are as follows:

- **Setup**: Generate a prime $p \equiv 3 \pmod 4$ of 2λ bits, where λ is the security parameter. Define a supersingular elliptic curve $E_0 : y^2 = x^3 + x$ over $\mathbb{F}_p$ with $j(E) = 1728$, and $\mathrm{End}(E_0) = \mathcal{O}_0$. Pick an odd smooth number D_c of λ bits and $D = 2^e$, where e is larger than the diameter of $\mathcal{G}_2(\overline{\mathbb{F}_p})$.

- **Key Generation**: Choose a prime $N_\tau \sim p^{\frac{1}{4}}$ and randomly select a N_τ-isogeny $\tau : E_0 \to E_A$. The secret key is the isogeny τ (note that the degree of τ is also private), and the public key is the image curve E_A.
- **Commitment**: The prover generates a random isogeny $\psi_1 : E_0 \to E_1$, and sends E_1 to the verifier.
- **Challenge**: The verifier sends a cyclic isogeny $\psi_2 : E_1 \to E_2$ of degree D_c to the prover.
- **Response**: From the knowledge of the isogeny $\psi_2 \circ \psi_1 \circ \hat{\tau} : E_A \to E_2$, the prover constructs an isogeny $\sigma : E_A \to E_2$ of degree D such that $\hat{\psi}_2 \circ \sigma$ is cyclic. After that, the prover sends σ to the verifier.
- **Verification**: The verifier accepts if the isogeny $\sigma : E_A \to E_2$ has degree D and $\hat{\psi}_2 \circ \sigma$ is cyclic. It rejects otherwise.

$$\begin{tikzcd}
& E_0 \arrow[r, "\psi_1"] \arrow[dl, dashed, "\tau"'] & E_1 \arrow[d, "\psi_2"] \\
E_A \arrow[rr, "\sigma"] & & E_2
\end{tikzcd}$$

Fig. 1: Sketch of the identification protocol.

Since the reduced norm of I_τ is a large prime, it is expensive to compute the corresponding isogeny τ directly by Vélu's formula. To compute the coefficient of E_A efficiently, one can use the KLPT algorithm to translate I_τ to another equivalent ideal I_2 of reduced norm 2^{e_τ}, which corresponds to an isogeny from E_0 to E_A of degree 2^{e_τ}. An alternative approach is to generate I_τ and I_2 simultaneously by finding $\gamma' \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k$ with reduced norm $N_\tau 2^{e_\tau}$, then set $I_\tau = \langle \gamma', N_\tau \rangle$ and $I_2 = \langle \overline{\gamma'}, 2^{e_\tau} \rangle$. Compared to the former one, the latter method is more efficient, and thus it is applied to the current implementation.

The response phase is the most complicated procedure. To avoid revealing the secret, one should first construct a new ideal I_σ using the SigningKLPT algorithm from the knowledge of $\psi_2 \circ \psi_1 \circ \hat{\tau}$, and then translate I_σ to the corresponding isogeny σ of degree D. In the following, we review the SigningKLPT algorithm and ideal to isogeny translation.

2.3 SigningKLPT algorithm

The KLPT algorithm was first proposed by [25]. To compute the equivalent ideal I_σ in $\mathrm{End}(E_A) \cong \mathcal{O}_A$, the authors in [16] generalized the KLPT algorithm to propose the SigningKLPT algorithm (Algorithm 1). We summarize the main procedures as follows:

1. **EquivlantRandomEichlerIdeal**(I,N_τ): Given a left $\mathcal{O}_A$-ideal I, outputs an ideal K of reduced norm coprime to N_τ which is equivalent to I.

2. **EquivlantPrimeIdeal**(I): Given a left $\mathcal{O}_0$-ideal I, outputs the smallest equivalent left $\mathcal{O}_0$-ideal of prime reduced norm.
3. **RepresentInteger**$_{\mathcal{O}_0}(M)$: Given an integer $M > p$, outputs $\gamma \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k \subseteq \mathcal{O}_0$ of reduced norm M.
4. **IdealModConstraint**(I, γ): Given a left-$\mathcal{O}_0$ ideal I of reduced norm N and $\gamma \in \mathcal{O}_0$, outputs $(C_0 : D_0) \in \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$ such that $\gamma\mu_0 \in I$, where $\mu_0 = j(C_0 + iD_0)$.
5. **EichlerModConstraint**(I, γ, δ): Given a left $\mathcal{O}_0$-ideal I of reduced norm N, $\gamma, \delta \in \mathcal{O}_0$ of reduced norms coprime to N, outputs $(C_1 : D_1) \in \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$ such that $\gamma\mu_1\delta \in \mathbb{Z} + I$, where $\mu_1 = j(C_1 + iD_1)$.
6. **StrongApproximation**$_{\ell^e}(N, C, D)$: Given $N, C, D \in \mathbb{Z}$, outputs $\mu = \lambda\mu_0 + N\mu_1$ of reduced norm ℓ^e, where $\mu_0 = j(C + iD)$ and $\mu_1 \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k$.

Algorithm 1 SigningKLPT(I_τ, I)

Require: An $(\mathcal{O}_0, \mathcal{O}_A)$-ideal I_τ of reduced norm N_τ, a left $\mathcal{O}_A$-ideal I.
Ensure: A left $\mathcal{O}$-ideal J of reduced norm ℓ^e such that $I \sim J$.
1: $K \leftarrow$**EquivlantRandomEichlerIdeal**(I,N_τ);
2: $K' \leftarrow [I_\tau]^*(K)$;
3: $L \leftarrow$**EquivlantPrimeIdeal**(K'), $N \leftarrow \mathrm{Nrd}(L)$;
4: Select $\delta \in K'$ such that $L = K'\frac{\bar{\delta}}{\mathrm{Nrd}(K')}$;
5: $e_0 \leftarrow e_0(N)$ and $e_1 \leftarrow e - e_0$;
6: $\gamma \leftarrow$**RepresentInteger**$_{\mathcal{O}_0}(N\ell^{e_0})$;
7: $(C_0 : D_0) \leftarrow$**IdealModConstraint**(L, γ);
8: $(C_1, D_1) \leftarrow$ **EichlerModConstraint**(I_τ, γ, δ);
9: $C \leftarrow$ **CRT**$_{N,N_\tau}(C_0, C_1)$, $D \leftarrow$ **CRT**$_{N,N_\tau}(D_0, D_1)$. If $\ell^e p(C^2 + D^2)$ is not a quadratic residue, go back to Step 6;
10: $\mu \leftarrow$**StrongApproximation**$_{\ell^{e_1}}(NN_\tau, C, D)$;
11: $\beta \leftarrow \gamma\mu$;
12: $J \leftarrow [I_\tau]_*\left(L\frac{\bar{\beta}}{\mathrm{Nrd}(L)}\right)$;
13: **return** J.

However, De Feo et al. [17] found that Algorithm 1 results in the invalid security proof of SQISign. To overcome this problem, they replaced **RepresentInteger** by **FullRepresentInteger** (defined below) to compute γ in the SigningKLPT algorithm. Heuristically, this modification leads to the uniform distribution of outputs.
FullRepresentInteger$_{\mathcal{O}_0}(M)$: Given an integer $M > p$, outputs $\gamma \in \mathcal{O}_0 \backslash 2\mathcal{O}_0$ of reduced norm dividing M.

2.4 Ideal to isogeny translation

The efficiency bottleneck of SQISign is the translation from the ideal I_σ to the corresponding isogeny σ. In the current implementation of SQISign, the signer needs to decompose the isogeny σ of degree 2^e into the isogenies φ_i, $i = 1, 2, \cdots, n$ of degree 2^a such that

$$\sigma = \varphi_n \circ \cdots \circ \varphi_2 \circ \varphi_1,$$

where a is the integer such that $2^a \| p + 1$. Here we review ideal to isogeny translation in detail. For simplicity, we only analyze how to generate φ_1, while the procedures to generate φ_i, $i = 2, \cdots, n$ are similar.

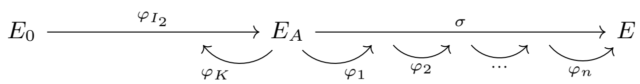


Fig. 2: Sketch of ideal to isogeny translation.

The core of ideal to isogeny translation is, given an isogeny φ_K of degree 2^a with kernel $\langle P \rangle$, one can find the corresponding isogeny of $I = \langle \alpha, 2^a \rangle$ by computing the kernel $\langle [C]P + [D]\theta(P) \rangle$, where $\theta \in \mathcal{O}_A \backslash (\mathbb{Z} + K + 2\mathcal{O}_A)$ has smooth reduced norm and satisfies that $\alpha(C + D\theta) \in K$ [17, Lemma 2]. To do this, the following two algorithms are required:
SpecialEichlerNorm$_T(\mathcal{O}, K)$: Given a maximal order $\mathcal{O}$ and a left $\mathcal{O}$-ideal K of reduced norm ℓ, outputs $\beta \in \mathcal{O} \backslash (\mathbb{Z} + K)$ of reduced norm dividing T^2, where T is a parameter such that $\gcd(T, \ell) = 1$ and $T | p^2 - 1$.
IdealToIsogeny(I): Given an ideal $I \subseteq \mathcal{O}_0$ of reduced norm dividing T, outputs the corresponding isogeny φ_I.

Algorithm 2 describes how to translate each φ_i. In the first execution to compute φ_1, the signer takes $\mathcal{O} = \mathcal{O}_A$, $I = I_\sigma + 2^a\mathcal{O}_A$, $J = I_2$, $\varphi_J = \varphi_{I_2}$ and the generator P of $E_A[2^a] \cap \ker(\hat{\varphi}_J)$ as the input.

Algorithm 2 IdealToIsogenyEichler$_{2^a}(\mathcal{O},\ I,\ J,\ \varphi_J,\ P)$

Require: A left $\mathcal{O}$-ideal I of reduced norm 2^a, an $(\mathcal{O}_0, \mathcal{O})$-ideal J of reduced norm $2^\bullet$ and $\varphi_J \colon E_0 \to E$ the corresponding isogeny, a generator P of $E[2^a] \cap \ker(\hat{\varphi}_J)$.
Ensure: φ_I of degree 2^a.
$K \leftarrow \overline{J} + 2^a\mathcal{O}$;
$\theta \leftarrow$ **SpecialEichlerNorm**$_T(\mathcal{O},\ K + 2\mathcal{O})$;
Select $\alpha \in I$ such that $I = \mathcal{O}\langle \alpha, 2^a \rangle$;
Compute C, D such that $\alpha(C + D\theta) \in K$ and $\gcd(C, D, 2) = 1$;
Take any $n_1 \mid T$ and $n_2 \mid T$ such that $n_1 n_2 = \mathrm{Nrd}(\theta)$. Compute $H_1 = \mathcal{O}\langle \theta, n_1 \rangle$ and $H_2 = \mathcal{O}\langle \overline{\theta}, n_2 \rangle$;
$L_i \leftarrow [J]^* H_i$, $\phi_i \leftarrow [\varphi_J]_*$**IdealToIsogeny**$(L_i)$ for $i \in \{1, 2\}$;
Compute $Q \leftarrow \hat{\phi}_2 \circ \phi_1(P)$;
Compute φ_I of kernel $\langle [C]P + [D]Q \rangle$;
return φ_I.

Remark 1. It should be noted that the implementation of ideal to isogeny translation also requires that the isogeny corresponding to the ideal $I_2 \cdot I_\sigma$ is cyclic. If not, in the second execution to compute φ_2, the input $J \subseteq 2\mathcal{O}_0$, which implies that $K = \overline{J} + 2^a\mathcal{O} \subseteq 2\mathcal{O}$. Hence, the isogeny φ_K is not cyclic. Therefore, one may repeat executing the SigningKLPT algorithm to generate I_σ until $\sigma \circ \varphi_{I_2}$ is cyclic.

The most expensive step of Algorithm 2 is to compute $Q = \theta(P) = \hat{\phi}_2 \circ \phi_1(P)$. To reduce the computational cost, one can utilize Algorithm 3 to obtain the x-coordinate of $[C]P + [D]Q$ from the knowledge of $\mathrm{Trd}(\theta)$. Compared to directly compute $Q = \theta(P)$, one isogeny construction could be saved.

Algorithm 3 EndomorphismEvaluation(ϕ_1, ϕ_2, C, D, t, P)

Require: Two isogenies ϕ_1, ϕ_2 from E to E', scalars C and D, the reduced trace $\mathrm{Trd}(\theta) = \mathrm{Trd}(\hat{\phi}_2 \circ \phi_1)$ and a point $P \in E[2^a]$.

Ensure: The x-coordinate of $[C]P + [D]\theta(P)$.

1: Compute Q such that $\langle P, Q\rangle = E[2^a]$ and compute $P + Q$;

2: Compute $x_{\phi_1(P)}$, $x_{\phi_1(Q)}$, $x_{\phi_2(P)}$, $x_{\phi_2(Q)}$, $x_{\phi_2(P+Q)}$;

3: Compute s_1, s_2 such that $x_{\phi_1(P)}$ is equal to the x-coordinate of $[s_1]\phi_2(P) + [s_2]\phi_2(Q)$;

4: Compute s_3, s_4 such that $x_{\phi_1(Q)}$ is equal to the x-coordinate of $[s_3]\phi_2(P) + [s_4]\phi_2(Q)$;

5: Change the signs of (s_1, s_2), (s_3, s_4) until $(s_1 + s_4)\deg(\phi_2) \equiv \mathrm{Trd}(\theta) \bmod 2^a$;

6: Compute the x-coordinate of $[C + s_1 D \deg(\phi_2)]P + [s_2 D \deg(\phi_2)]Q$ and set it as x_R;

7: **return** x_R.

2.5 Reduced Tate pairing

Let E be an elliptic curve over $\overline{\mathbb{F}_p}$, the reduced Tate pairing is a map :

$$e_n : E(\mathbb{F}_q)[n] \times E(\mathbb{F}_q)/nE(\mathbb{F}_q) \to \mu_n,$$

where q is the power of p and μ_n is the n-roots of unity in $\overline{\mathbb{F}_p}$. There are some properties of the reduced Tate pairing [21, Theorems IX.7, IX.9]:

1. Assume $P_1, P_2 \in E(\mathbb{F}_q)[n]$, $P_3, P_4 \in E(\mathbb{F}_q)/nE(\mathbb{F}_q)$. Then
$$e_n(P_1 + P_2, P_3) = e_n(P_1, P_3)e_n(P_2, P_3),$$
$$e_n(P_1, P_3 + P_4) = e_n(P_1, P_3)e_n(P_1, P_4).$$
2. Let $P \in E(\mathbb{F}_q)[n]$. If $e_n(P, Q) = 1$ for any $Q \in E(\mathbb{F}_q)/nE(\mathbb{F}_q)$, then $P = \infty_E$.
3. Let $Q \in E(\mathbb{F}_q)/nE(\mathbb{F}_q)$. If $e_n(P, Q) = 1$ for any $P \in E(\mathbb{F}_q)[n]$, then $Q \in nE(\mathbb{F}_q)$.

4. Let $\varphi : E \to E'$ be an isogeny, $P \in E(\mathbb{F}_q)[n]$, $Q' \in E'(\mathbb{F}_q)/nE'(\mathbb{F}_q)$, then
$$e_n(\varphi(P), Q') = e_n(P, \hat{\varphi}(Q')).$$
5. Let $P \in E(\mathbb{F}_q)[N]$ and $Q \in E(\mathbb{F}_q)$, where $N = nn'$. Then
$$e_n([n']P, Q) = e_N(P, Q)^{n'}.$$

3 Faster Generation of I_σ

The aim of this section is to speed up the performance of generating the required I_σ in the signing phase. As mentioned in Remark 1, the isogeny $\sigma \circ \varphi_{I_2}$ should be cyclic. Indeed, there exist three edges from the vertex $[E_A]$ in $\mathcal{G}_2(\overline{\mathbb{F}_p})$. Heuristically, there is a 33.3% probability that the isogeny is not cyclic. Therefore, one may try several times to obtain I_σ by applying the SigningKLPT algorithm to satisfy the above condition. It enlarges the computational cost of SQISign. For the rest of this section, we propose Propositions 1, 2 and 3 to present a modified SigningKLPT algorithm to overcome this issue, i.e., avoiding repeated calls to the SigningKLPT algorithm.

In the key generation phase we have $I_\tau = \langle \gamma', N_\tau \rangle$ and $I_2 = \langle \overline{\gamma'}, 2^{e_\tau} \rangle$, where $\gamma' \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k$ of reduced norm $N_\tau 2^{e_\tau}$. In the following, we give Proposition 1 to show that γ' is always contained in $\mathcal{O}_0(1+i)$, where $\mathcal{O}_0 = \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}\frac{i+j}{2} + \mathbb{Z}\frac{1+k}{2}$.

Proposition 1. *Assume that* $\alpha \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k \subseteq \mathcal{O}_0$. *If* $\mathrm{Nrd}(\alpha)$ *is even, then* $\alpha \in \mathcal{O}_0(1+i)$.

Proof. Since $\alpha \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k$, we can assume $\alpha = a + bi + cj + dk$ with $a, b, c, d \in \mathbb{Z}$, then α can be written as
$$\alpha = (a - d) + (b - c)i + 2c\frac{i+j}{2} + 2d\frac{1+k}{2}.$$
Denote $S = \{a, b, c, d\}$. Since the reduced norm of α is even, we have $a^2 + b^2 + pc^2 + pd^2$ is even. Now we prove that $a - d$ and $b - c$ are both odd or even. On the contrary, if one of $a - d$, $b - c$ is odd and the other is even, then one of the following statements holds:

- The set S contains only one odd element.
- The set S contains only one even element.

In both cases, we can deduce that $a^2 + b^2 + pc^2 + pd^2$ is odd. It is a contradiction.

Hence, it implies that $(a - d) - (b - c)$ is even, and
$$\begin{aligned}\alpha &= [(a-d)-(b-c)] + (b-c)(1+i) + 2c\frac{i+j}{2} + 2d\frac{1+k}{2}\\ &= \left[\frac{(a-d)-(b-c)}{2}(1-i) + (b-c) + c\frac{i+j}{2}(1-i) + d\frac{1+k}{2}(1-i)\right](1+i).\end{aligned}$$
Therefore, $\alpha \in \mathcal{O}_0(1+i)$. $\square$

Now we propose Proposition 2, which can be used to check whether the composition of two cyclic isogenies is still cyclic or not.

Proposition 2. *Assume* $\varphi_1 : E_1 \to E_2$*,* $\varphi_2 : E_2 \to E_3$ *are two cyclic isogenies. Then* $\varphi_2 \circ \varphi_1$ *is cyclic if and only if* $\ker(\hat{\varphi}_1) \cap \ker(\varphi_2) = \{\infty_{E_2}\}$.

Proof. If $\ker(\hat{\varphi}_1) \cap \ker(\varphi_2) \neq \{\infty_{E_2}\}$, there exists a point P of prime order ℓ such that $P \in \ker(\hat{\varphi}_1) \cap \ker(\varphi_2)$. Since the isogeny φ_1 is surjective, we can assume $\varphi_1(P') = P$. Obviously, $\langle P' \rangle \subseteq \ker(\varphi_2 \circ \varphi_1)$. Besides, it is clear that $\ker(\varphi_1) \subseteq \ker(\varphi_2 \circ \varphi_1)$ and $\langle P' \rangle \cap \ker(\varphi_1) = \{\infty_{E_1}\}$. Therefore, $E_1[\ell] \subseteq \ker(\varphi_2 \circ \varphi_1)$, which implies $\varphi_2 \circ \varphi_1$ is not cyclic.

On the other hand, if $\varphi_2 \circ \varphi_1$ is not a cyclic isogeny, then there exists a prime ℓ such that $\ell | \deg(\varphi_1)$ and $E_1[\ell] \subseteq \ker(\varphi_2 \circ \varphi_1)$. Let $\langle P \rangle \subseteq \ker(\varphi_1)$ be a subgroup of order ℓ. Suppose that $\langle P, Q \rangle = E_0[\ell]$, then $\varphi_1(Q) \in \ker(\hat{\varphi}_1)$ and $\varphi_1(Q) \in \ker(\varphi_2)$. This deduces that $\varphi_1(Q) \in \ker(\hat{\varphi}_1) \cap \ker(\varphi_2)$, which is a contradiction. □

Remark 2. The proof of Proposition 2 also implies that the isogeny $\varphi_2 \circ \varphi_1$ is cyclic if and only if $(\ker(\hat{\varphi}_1) \cap E_2[\ell]) \cap (\ker(\varphi_2) \cap E_2[\ell]) = \{\infty_{E_2}\}$ for any prime ℓ dividing $\gcd(\deg(\varphi_1), \deg(\varphi_2))$, which is equivalent to the first steps (ℓ-isogeny) of $\hat{\varphi}_1$ and φ_2 are different.

$$\begin{tikzcd} E_0 \arrow[r, bend right, "\tau"'] \arrow[rr, bend left, "\gamma'"] & E_A \arrow[r, bend right, "\hat{\varphi}_{I_2}"'] & E_0 \end{tikzcd}$$

Fig. 3: Decomposition of γ'.

Since $I_\tau \overline{I_2} = \langle \gamma', N_\tau \rangle \langle \gamma', 2^{e_\tau} \rangle = \langle \gamma' \rangle$, we can deduce that the endomorphism $\gamma' = \hat{\varphi}_{I_2} \circ \tau$ (illustrated in Figure 3). Therefore,

$$\gamma'(\ker(1+i)) = \hat{\varphi}_{I_2} \circ \tau(\ker(1+i)).$$

Note that $\gamma' \in \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}k$ and its reduced norm is even. Thanks to Proposition 1, $\gamma'(\ker(1+i)) = \{\infty_{E_2}\}$ and thus $\tau(\ker(1+i)) \subseteq \ker(\hat{\varphi}_{I_2})$, i.e., $\tau(\ker(1+i)) = \ker(\hat{\varphi}_{I_2}) \cap E_A[2]$.

According to Proposition 2, the isogeny $\sigma \circ \varphi_{I_2}$ is cyclic if and only if $\ker(\hat{\varphi}_{I_2}) \cap \ker(\sigma) = \ker(\hat{\varphi}_{I_2}) \cap \tau(\ker(\varphi_{I'})) = \{\infty_{E_A}\}$, where $I' = L\frac{\overline{\gamma\mu}}{\mathrm{Nrd}(L)}$ and γ, μ, L are obtained in the SigningKLPT algorithm.

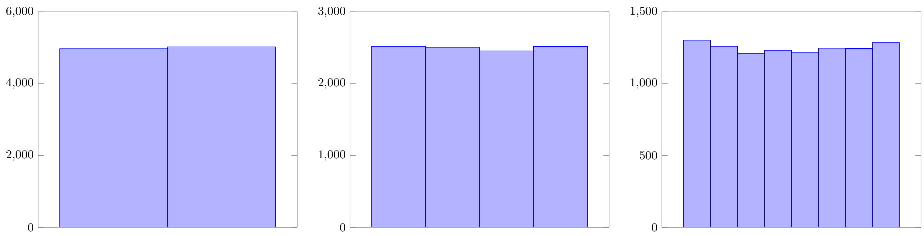
Since the endomorphism $\bar{\mu}\bar{\gamma}$ is the composition $\varphi_{\bar{L}} \circ \varphi_{I'}$ (illustrated in Figure 4), the first step (2-isogeny) of $\varphi_{I'}$ is that of $\bar{\gamma}$, which has kernel $\ker(\bar{\gamma}) \cap E_0[2]$. It implies that the kernel of the first step of σ is $\tau(\ker(\bar{\gamma}) \cap E_0[2])$. Note that the kernel of the first step of $\hat{\varphi}_{I_2}$ is $\ker(\hat{\varphi}_{I_2}) \cap E_A[2] = \tau(\ker(1+i))$. Therefore, we have the following proposition:

$$\begin{tikzcd}
E_0 \arrow[r, bend left, "\varphi_{I'}"] \arrow[rr, bend right, "\bar{\mu}\bar{\gamma}"'] & E' \arrow[r, bend left, "\varphi_{\bar{L}}"] & E_0
\end{tikzcd}$$

Fig. 4: Decomposition of φ_I.

Proposition 3. *Using the same notation as before, the isogeny $\sigma \circ \varphi_{I_2}$ is cyclic if and only if the first step of $\bar{\gamma}$ is not the loop $1+i$, i.e., $\gamma \notin (1+i)\mathcal{O}_0$.*

We propose the modified SigningKLPT algorithm in Algorithm 4. The main difference between ours and the previous work is that we ensure $\gamma \notin (1+i)\mathcal{O}_0$ in Steps 5-7. Analogous to the previous work, we first use Algorithm **FullRepresentInteger** to generate γ. If $\gamma \in (1+i)\mathcal{O}_0$, we set $\gamma = \frac{(1+i)}{2}\gamma$. With this minor modification, we can ensure the isogeny σ corresponding to the output of Algorithm 4 always satisfies that $\sigma \circ \varphi_{I_2}$ is cyclic. Heuristically, this modification does not lead to biased distributions of the first several steps of the response σ, as we report in Figure 5.



(a) First step (b) Second step (c) Third step

Fig. 5: Distribution of first three steps of the response σ for 10 secret keys and random ideal in input over 1000 attempts.

Even though the generator γ' is not in $\mathcal{O}_0(1+i)$, one can modify the SigningKLPT algorithm to achieve the goal with Lemma 1.

Lemma 1. *Let $p \equiv 3 \pmod 4$, $p > 8$, E_0 be a supersingular elliptic curve with j-invariant 1728. Then in $\mathcal{G}_2(\overline{\mathbb{F}_p})$, the vertex $[E_0]$ has one loop which corresponds to the ideal $\mathcal{O}_0(1+i)$, and connects to the vertex $[E_6]$ ($E_6 : y^2 = x^3 + 6x^2 + x$) by 2 edges which correspond to the non-principal ideals I_0 and $I_0 i$, respectively.*

Proof. The proof follows from [26, Section 5]. □

Algorithm 4 ModifiedSigningKLPT(I_τ, I)

Require: An $(\mathcal{O}_0, \mathcal{O}_A)$-ideal I_τ of reduced norm N_τ, a left $\mathcal{O}_A$-ideal I.
Ensure: Left integral $\mathcal{O}$-ideal J of reduced norm ℓ^e such that $I \sim J$.
1: $K \leftarrow$**EquivlantRandomEichlerIdeal**(I,N_τ);
2: $K' \leftarrow [I_\tau]^*(K)$;
3: $L \leftarrow$**EquivlantPrimeIdeal**(K'), $N \leftarrow \mathrm{Nrd}(L)$;
4: Select $\delta \in K'$ such that $L = K' \frac{\bar{\delta}}{\mathrm{Nrd}(K')}$;
5: $e_0 \leftarrow e_0(N)$;
6: $\gamma \leftarrow$**FullRepresentInteger**$_{\mathcal{O}_0}(N\ell^{e_0})$;
7: **if** $(1+i)\gamma \in 2\mathcal{O}_0$ **then**
8: $\quad \gamma \leftarrow \frac{(1+i)\gamma}{2}$;
9: **end if**
10: $e_0' \leftarrow \log_2(\mathrm{Nrd}(\gamma)/N)$, $e_1 \leftarrow e - e_0'$.
11: $(C_0 : D_0) \leftarrow$**IdealModConstraint**(L, γ);
12: $(C_1, D_1) \leftarrow$ **EichlerModConstraint**(I_τ, γ, δ);
13: $C \leftarrow$ **CRT**$_{N,N_\tau}(C_0, C_1)$, $D \leftarrow$ **CRT**$_{N,N_\tau}(D_0, D_1)$. If $\ell^e p(C^2 + D^2)$ is not a quadratic residue, go back to Step 6;
14: $\mu \leftarrow$**StrongApproximation**$_{\ell^{e_1}}(NN_\tau, C, D)$;
15: $\beta \leftarrow \gamma\mu$;
16: $J \leftarrow [I_\tau]_* \left(L \frac{\bar{\beta}}{\mathrm{Nrd}(L)}\right)$;
17: **return** J.

Assume that the first step (2-isogeny) of γ' corresponds to the non-principal ideal I_0, while the other case is similar. As above, the isogeny $\sigma \circ \varphi_{I_2}$ is cyclic if and only if the first steps of $\bar{\gamma}$ and γ' have distinct kernels. If the first step of $\bar{\gamma}$ is not φ_{I_0}, then the isogeny $\sigma \circ \varphi_{I_2}$ is cyclic. Otherwise, we can modify γ by $i\gamma$ or $(1+i)\gamma$ to ensure the first step of $\bar{\gamma}$ is $i(\ker(\varphi_{I_0}))$ or $(1+i)(\ker(\varphi_{I_0}))$, which is not equal to $\ker(\varphi_{I_0})$, i.e., the isogeny $\sigma \circ \varphi_{I_2}$ is cyclic.

4 Efficient Elliptic Curve Discrete Logarithm Computations

In this section, we focus on how to solve the two elliptic curve discrete logarithms in Algorithm 3 and propose a more efficient approach to obtain s_1 and s_2.

In the current implementation, each ideal to isogeny translation requires computing two elliptic curve discrete logarithms. To be precise,

$$\begin{aligned} \phi_1(P) &= [s_1]\phi_2(P) + [s_2]\phi_2(Q), \\ \phi_1(Q) &= [s_3]\phi_2(P) + [s_4]\phi_2(Q). \end{aligned} \tag{1}$$

where ϕ_1, ϕ_2 are two isogenies of odd degree. For simplicity, we denote $P_i = \phi_i(P)$ and $Q_i = \phi_i(Q)$, $i = 1, 2$.

The authors in [17] used the Pohlig-Hellman algorithm [33] with a balanced strategy to simplify the above two elliptic curve discrete logarithms in the group $E_A[2^a]$ into multiple elliptic curve discrete logarithms in the group $E_A[2]$. For efficiency, they suggested using the x-only arithmetic to recover the absolute values of s_1, s_2, s_3 and s_4 by computing two elliptic curve discrete logarithms in Equation (1), and then determine the signs of them with the help of $\mathrm{Trd}(\theta)$. However, the cost is still relatively large. This method needs to compute the x-coordinates of $P_i + Q_j$ and $P_1 + P_2 + Q_i$ $(i, j = 1, 2)$ in advance and store all of them into a stack. During the computation, all the elements in the stack need to be updated frequently in order to entirely utilize the x-only arithmetic. On the other hand, as we can see in Algorithm 3, the goal of computing the absolute values of s_3 and s_4 in the second elliptic curve discrete logarithm is merely to confirm the signs of s_1 and s_2. It is natural to ask whether one could compute only one elliptic curve discrete logarithm to obtain the exact values of s_1 and s_2.

In the following, we propose a more efficient method to obtain the exact values of s_1 and s_2. Firstly, we show how to avoid the second elliptic curve discrete logarithm computation in Equation (1) with the knowledge of θ. Next, inspired by previous works, we take full advantage of pairing computations to translate the first elliptic curve discrete logarithm into two discrete logarithms in the finite field $\mathbb{F}_{p^2}$. Finally, we show how to compute the two discrete logarithms in $\mathbb{F}_{p^2}$ efficiently.

4.1 Saving one elliptic curve discrete logarithm computation

Now we propose Theorem 1, a key observation leading to the saving of the second elliptic curve discrete logarithm computation in Equation (1):

Theorem 1. *Assume that φ_J is a cyclic $2^\bullet$-isogeny from E_0 to E, and J is the corresponding right $\mathcal{O}$-ideal. Suppose that $K = \overline{J} + 2\mathcal{O}$. If the endomorphism $\theta \in \mathcal{O} \setminus (\mathbb{Z} + K)$ and P is a point of order 2^a such that $\langle P \rangle = E[2^a] \cap \ker(\hat{\varphi}_J)$, then $\theta([2^{a-1}]P) \neq [2^{a-1}]P$.*

Proof. Clearly, the ideal corresponding to the isogeny $\hat{\varphi}_J$ is $\overline{J}$. Hence, for any $\delta \in K = \overline{J} + 2\mathcal{O}$, we have

$$\delta([2^{a-1}]P) = \infty_E.$$

Suppose that $\theta([2^{a-1}]P) = [2^{a-1}]P$. Since $\theta([2^{a-1}]P) - [2^{a-1}]P = \infty_E$, we have $\theta - 1 \in K$ from the Deuring Correspondence Theorem. It implies that $\theta \in \mathbb{Z} + K$. This contradicts the fact that $\theta \in \mathcal{O} \setminus (\mathbb{Z} + K)$. □

Theorem 1 implies that in each ideal to isogeny translation, the endomorphism θ we handle always maps $[2^{a-1}]P$ to a point which is not $[2^{a-1}]P$. Since the reduced norm of θ divides T^2 and T is odd, $\theta([2^{a-1}]P)$ is not the point at infinity. This implies that the endomorphism θ maps $[2^{a-1}]P$ to another point of order 2.

In the following, we show that s_2 in Equation (1) is always odd. It confirms that $s_2^{-1} \bmod 2^a$ exists, which can be employed to accelerate the performance.

Corollary 1. *At Step 3 of Algorithm 3, we have $s_2 \equiv 1 \bmod 2$.*

Proof. From $P_1 = [s_1]P_2 + [s_2]Q_2$, we have

$$[2^{a-1}]P_1 = [s_1]([2^{a-1}]P_2) + [s_2]([2^{a-1}]Q_2).$$

Suppose for contradiction that s_2 is even. Since the order of $[2^{a-1}]Q_2$ is 2, $[s_2]([2^{a-1}]Q_2)$ is the point at infinity. Therefore,

$$[2^{a-1}]P_1 = [s_1]([2^{a-1}]P_2).$$

Applying $\hat{\phi}_2$ to the above equation yields:

$$\theta([2^{a-1}]P) = [s_1 \deg(\phi_2)]([2^{a-1}]P).$$

From the deduction above, we know that $\theta([2^{a-1}]P)$ is of order 2. It implies that $\theta([2^{a-1}]P) = [2^{a-1}]P$, which is a contradiction with Theorem 1. Hence, we have $s_2 \equiv 1 \bmod 2$. □

With the investigation above, we can directly compute the absolute values of s_3 and s_4 with the help of $\mathrm{Trd}(\theta)$ and $\mathrm{Nrd}(\theta)$ instead of computing the second elliptic curve discrete logarithm in Equation (1). To be precise, after recovering the absolute values of s_1 and s_2 in the first elliptic curve discrete logarithm computation, one can suppose

$$\begin{aligned} s_4 =& \mathrm{Trd}(\theta) - s_1 \bmod 2^a, \\ s_3 =& \frac{s_1 s_4 - \mathrm{Nrd}(\theta)}{s_2} \bmod 2^a. \end{aligned} \tag{2}$$

Then, compute the x-coordinate of $[s_3]P_2+[s_4]Q_2$. If the x-coordinate of $[s_3]P_2+[s_4]Q_2$ is equal to that of Q_1, then the signs of s_1 and s_2 are correct. Otherwise, we need to change the signs of them. The main procedure is summarized in Algorithm 5:

Algorithm 5 EndomorphismEvaluation(φ_1, φ_2, C, D, t, n, P)

Require: Two isogenies ϕ_1, ϕ_2 from E to E', scalars C and D, the reduced trace $\mathrm{Trd}(\theta) = \mathrm{Trd}(\hat{\phi}_2 \circ \phi_1)$, the reduced norm $\mathrm{Nrd}(\theta) = \mathrm{Nrd}(\hat{\phi}_2 \circ \phi_1)$ and a point $P \in E[2^a]$.
Ensure: The x-coordinate of $[C]P + [D]\theta(P)$.
Compute Q such that $\langle P, Q\rangle = E[2^a]$ and compute $P + Q$;
Compute $x_{\phi_1(P)}$, $x_{\phi_1(Q)}$, $x_{\phi_2(P)}$, $x_{\phi_2(Q)}$, $x_{\phi_2(P+Q)}$;
Compute s_1, s_2 such that $x_{\phi_1(P)}$ is equal to the x-coordinate of $[s_1]\phi_2(P) + [s_2]\phi_2(Q)$;
Let $s_4 = \mathrm{Trd}(\theta) - s_1 \bmod 2^a$ and $s_3 = (s_1 s_4 - \mathrm{Nrd}(\theta))/s_2 \bmod 2^a$;
Compute the x-coordinate of $[s_3]\phi_2(P) + [s_4]\phi_2(Q)$ and set it as x_t;
if $x_t \neq x_{\phi_1(Q)}$ **then**
    $s_1 \leftarrow -s_1$, $s_2 \leftarrow -s_2$;
end if
Compute the x-coordinate of $[C + s_1 D \deg(\phi_2)]P + [s_2 D \deg(\phi_2)]Q$ and set it as x_R;
return x_R.

At the beginning of this section, we reviewed the current implementation of computing discrete logarithms on elliptic curves in SQISign. Even though the authors in [17] utilized the x-only arithmetic, it is still an expensive procedure. A question raised here is how to compute the first elliptic curve discrete logarithm in Equation (1) more efficiently.

Our optimization is reminiscent of public-key compression in SIDH [3]. That is, applying pairings (note that the pairing we use should satisfy $e_{2^a}(R, R) = 1$ for any $R \in E(\mathbb{F}_{p^2})[2^a]$) to translate the elliptic curve discrete logarithm into two discrete logarithms in the cyclic group $\mu_{2^a} = \{h^{2^a} = 1 | h \in \mathbb{F}_{p^2}\}$:

$$\begin{aligned} h_0 &= e_{2^a}(P_2, Q_2), \\ h_1 &= e_{2^a}(P_2, P_1) = e_{2^a}(P_2, [s_1]P_2 + [s_2]Q_2) = e_{2^a}(P_2, [s_2]Q_2) = h_0^{s_2}, \\ h_2 &= e_{2^a}(Q_2, P_1) = e_{2^a}(P_2, [s_1]P_2 + [s_2]Q_2) = e_{2^a}(Q_2, [s_1]P_2) = h_0^{-s_1}. \end{aligned} \tag{3}$$

In Sections 4.2 and 4.3, we show how to efficiently compute the pairings in Equation (3) and the two discrete logarithms in μ_{2^a} to recover s_1 and s_2, respectively.

4.2 Pairing computations

In this subsection, we show why we can adapt the reduced Tate pairing in Equation (3) and explore how to compute h_0, h_1 and h_2 efficiently. Besides, we analyze the situation when using the Weil pairing. For simplicity, we write $e_{T,n}(\cdot,\cdot)$ and $e_{W,n}(\cdot,\cdot)$ to denote the reduced Tate pairing and the Weil pairing, respectively.

Since the embedding degree is equal to 1, one may doubt whether $e_{T,2^a}(R, R)$ is equal to 1 for any $R \in E(\mathbb{F}_{p^2})[2^a]$ in this case. Hence, the deduction of Equation (3) when applying the reduced Tate pairing may not be convinced. Indeed, the fact that $e_{T,2^a}(R, R) = 1$ has been applied into public-key compression in SIDH [11]. It seems that the correctness of the above fact has been well known to the experts. However, we do not find a relevant proof in the literature. Therefore, we propose Theorem 2 for illustrating the special feature of the reduced Tate pairing in our case.

Theorem 2. *Suppose that E is a supersingular elliptic curve over $\mathbb{F}_{p^2}$, where $2^a \| p + 1$ and $E[2^a] \subseteq E(\mathbb{F}_{p^2})$ with $a > 2$. Then $e_{T,2^a}(R, R) = 1$ for any $R \in E(\mathbb{F}_{p^2})[2^a]$.*

Proof. Since isogeny graphs for supersingular elliptic curves have the Ramanujan property [32], there exists an isogeny $\psi : E_0 \to E$ of degree coprime to 2. Therefore,

$$e_{T,2^a}(R, R)^{\deg(\psi)} = e_{T,2^a}(\hat{\psi}(R), \hat{\psi}(R)).$$

It implies that $e_{T,2^a}(R, R) = 1$ if and only if $e_{T,2^a}(\hat{\psi}(R), \hat{\psi}(R)) = 1$ for any $R \in E(\mathbb{F}_{p^2})[2^a]$.

As $E_0(\mathbb{F}_p)[2^a] \cong \mathbb{Z}/2^a\mathbb{Z}$, one can select a point $P_0 \in E_0(\mathbb{F}_p)$ of order 2^a such that $[2^{a-1}]P_0 = (0,0)$. Since $\mathrm{End}(E_0) \cong \mathcal{O}_0 = \langle 1, i, \frac{i+j}{2}, \frac{1+k}{2} \rangle$, we set $Q_0 = \iota(P_0) \in E_0(\mathbb{F}_{p^2})$, where ι corresponds to $\frac{i+j}{2}$. Due to the fact that

$$\frac{i+j}{2}(1+i) = \frac{-1+i+j-k}{2} \notin 2\mathcal{O}_0,$$

we have $\frac{i+j}{2} \notin \mathcal{O}_0(1+i)$. This implies that $\iota((0,0)) \neq \infty_{E_0}$. Since

$$[2^{a-1}]Q_0 = [2^{a-1}]\iota(P) = \iota([2^{a-1}]P) = \iota((0,0)) \neq \infty_{E_0},$$

we have the order of Q_0 is also 2^a. Now we show $\langle P_0, Q_0 \rangle = E_0(\mathbb{F}_p^2)[2^a]$. Suppose for contradiction that $[2^{a-1}]Q_0 = [2^{a-1}]P_0 = (0,0)$. Then $[2^{a-1}]\iota(P_0) = \iota((0,0)) = (0,0)$, i.e.,

$$(\iota - 1)((0,0)) = \infty_{E_0}.$$

However, noting that $\mathrm{Nrd}\left(\frac{i+j}{2} - 1\right) = 1 + \frac{p+1}{4}$ is odd, the point $(\iota-1)((0,0))$ is not equal to ∞_{E_0}, which is a contradiction.

As a consequence, there exist $r, s \in \mathbb{Z}/2^a\mathbb{Z}$ such that $\hat{\psi}(R) = [r]P_0 + [s]Q_0$. Using the properties of the reduced Tate pairing,

$$\begin{aligned}
&e_{T,2^a}(\hat{\psi}(R), \hat{\psi}(R))\\
=&e_{T,2^a}([r]P_0 + [s]Q_0, [r]P_0 + [s]Q_0)\\
=&e_{T,2^a}([r]P_0, [r]P_0)e_{T,2^a}([r]P_0, [s]Q_0)e_{T,2^a}([s]Q_0, [r]P_0)e_{T,2^a}([s]Q_0, [s]Q_0)\\
=&e_{T,2^a}(P_0, P_0)^{r^2}e_{T,2^a}(P_0, Q_0)^{rs}e_{T,2^a}(Q_0, P_0)^{rs}e_{T,2^a}(Q_0, Q_0)^{s^2}.
\end{aligned}$$

Since

$$\begin{aligned}
e_{T,2^a}(P_0, Q_0)^{rs}e_{T,2^a}(Q_0, P_0)^{rs} =&e_{T,2^a}(P_0, \iota(P_0))^{rs}e_{T,2^a}(\iota(P_0), P_0)^{rs}\\
=&e_{T,2^a}(P_0, \iota(P_0))^{rs}e_{T,2^a}(P_0, \hat{\iota}(P_0))^{rs}\\
=&e_{T,2^a}(P_0, \iota(P_0) + \hat{\iota}(P_0))^{rs}\\
=&e_{T,2^a}(P_0, \mathrm{Trd}(\iota)(P_0))^{rs}\\
=&e_{T,2^a}(P_0, \infty_{E_0})^{rs}\\
=&1,
\end{aligned}$$

and

$$\begin{aligned}
e_{T,2^a}(Q_0, Q_0)^{s^2} =&e_{T,2^a}(\iota(P_0), \iota(P_0))^{s^2}\\
=&e_{T,2^a}(P_0, \hat{\iota} \circ \iota(P_0))^{s^2}\\
=&e_{T,2^a}(P_0, \mathrm{Nrd}(\iota)(P_0))^{s^2}\\
=&e_{T,2^a}(P_0, P_0)^{\mathrm{Nrd}(\iota)s^2},
\end{aligned}$$

we have

$$e_{T,2^a}(\hat{\psi}(R), \hat{\psi}(R)) =e_{T,2^a}(P_0, P_0)^{r^2+\mathrm{Nrd}(\iota)s^2}.$$

Note that P_0 is defined on $E_0(\mathbb{F}_p)$ and the final exponentiation is an exponentiation to the power $\frac{p^2-1}{2^a}$. We can deduce that $e_{T,2^a}(P_0, P_0) = 1$ and therefore,

$$e_{T,2^a}(\hat{\psi}(R), \hat{\psi}(R)) = 1.$$

This concludes the proof. $\square$

It remains to explore how to efficiently compute the reduced Tate pairings. In the SIDH/SIKE implementation [2], Naehrig et al. [30] used the dual isogeny to pull back the pairing computations from the image curve to the starting curve. However, this technique does not work here because

$$h_0 = e_{2^a}(\hat{\varphi}_J(P), \hat{\varphi}_J(Q)) = e_{2^a}(P,Q)^{\deg(\varphi_J)} = e_{2^a}(P,Q)^{2^{a+\bullet}} = 1.$$

Similarly, we have $h_1 = h_2 = 1$. Therefore, we have to compute the three pairings in Equation (3) on the image curve E, as did in [3,11].

The reduced Tate pairing computations mainly contain two procedures: Miller function construction and the final exponentiation. Compared to the latter, the former one consumes larger computational resources because of the low embedding degree. In the SIDH/SIKE implementation, the state-of-the-art improves the Miller loop computation by the following formula with precomputation [27]:

$$\operatorname{div}(f_{4^{n+1},R}) = \operatorname{div}\left(\frac{\left[f_{4^n,R}^2 \cdot \left(\lambda_1(x - x_{[2\cdot 4^n]R}) - (y + y_{[2\cdot 4^n]R})\right)\right]^2}{\lambda_2(x - x_{[2\cdot 4^n]R}) - (y + y_{[2\cdot 4^n]R})}\right), \quad (4)$$

where the function $f_{N,R}$ is rational with divisor $\operatorname{div}(f_{N,R}) = N(R) - ([N]R) - (N-1)(\infty_E)$, the value λ_1 is the slope of the line passing through $[4^n]R$ twice and the value λ_2 is the slope of the line passing through $[-2 \cdot 4^n]R$ twice. In our case, we are not able to apply the precomputation technique since the two arguments are unknown, but we can still use Equation (4) instead of adapting the usual doubling step:

$$\operatorname{div}(f_{2^{n+1},R}) = \operatorname{div}\left(f_{2^n,R}^2 \frac{\lambda_1(x - x_{[2^n]R}) - (y - y_{[2^n]R})}{x - x_{[2^{n+1}]R}}\right), \quad (5)$$

where λ_1 is the slope of the line passing through $[2^n]R$ twice.

For efficiency, we use modified Jacobian coordinates to compute the pairings. The doubling operation requires only $3\boldsymbol{M} + 5\boldsymbol{S}$ [5], where $\boldsymbol{S}, \boldsymbol{M}$ are the cost of an $\mathbb{F}_{p^2}$ field squaring and multiplication, respectively. Another advantage of adapting modified Jacobian coordinates is that during the computation of doubling/quadrupling of R one could also obtain λ_1 and λ_2 easily.

According to our estimate, each quadrupling Miller loop using Equation (4) with modified Jacobian coordinates costs $17\boldsymbol{M} + 13\boldsymbol{S}$. It saves $3\boldsymbol{M} + 1\boldsymbol{S}$ compared to computing two doubling Miller loops using Equation (5) with modified Jacobian coordinates.

The final exponentiation is an exponentiation to the power $\frac{p^2-1}{2^a} = (p-1) \cdot \frac{p+1}{2^a}$. Raising to the power $p-1$ is an easy part, since it only costs one application of the Frobenius map and one inversion in $\mathbb{F}_{p^2}$. As for the exponentiation to the power $\frac{p+1}{2^a}$, one could use the efficient formulas in the cyclotomic subgroup $\mu_{p+1} = \{h^{p+1} = 1 | h \in \mathbb{F}_{p^2}\}$ [11, Section 5.1]. Another effective method, which is proposed by Scott et al. [36], is to raise the power with the help of Lucas sequences [34, Section 3.6.3]. In the implementation, we employ the latter one since it performs better.

In fact, we can further optimize the computation from the relations of h_0 and h_1. Adapting the reduced Tate pairings in Equation (3),

$$\begin{aligned} h_0 &= e_{T,2^a}(P_2, Q_2) = f_{2^a,P_2}(Q_2)^{\frac{p^2-1}{2^a}}, \\ h_1 &= e_{T,2^a}(P_2, P_1) = f_{2^a,P_2}(P_1)^{\frac{p^2-1}{2^a}}, \\ h_2 &= e_{T,2^a}(Q_2, P_1) = f_{2^a,Q_2}(P_1)^{\frac{p^2-1}{2^a}}. \end{aligned} \tag{6}$$

Note that the first two pairings share the same first argument. Therefore, when applying the reduced Tate pairing, we can further improve the computations of h_0 and h_1 by combining them together and one Miller function construction can be saved.

Remark 3. The techniques proposed above can not be directly applied into the case when the order of the pairing is $2^{a'}$, where $a' < a$. It is because given a class in $E(\mathbb{F}_{p^2})/2^{a'}E(\mathbb{F}_{p^2})$, we may not find an element in $E(\mathbb{F}_{p^2})[2^{a'}]$ to be the representative of the class. Assume that $\langle P, Q\rangle = E(\mathbb{F}_{p^2})[2^{a'}]$. Since $\langle [2^{a-a'}]P, [2^{a-a'}]Q\rangle \in [2^{a'}]E(\mathbb{F}_{p^2})$ and the second argument of the reduced Tate pairing is a representative of the class in $E(\mathbb{F}_{p^2})/2^{a'}E(\mathbb{F}_{p^2})$, the order of the reduced Tate pairing $e_{2^{a'}}(P, Q)$ is of order $2^{2a'-a}$ in $\mathbb{F}_{p^2}$. For example, set $a' = a - 1$. In this situation, all the points in $E(\mathbb{F}_{p^2})[2]$ represent the same class $[\infty_E]$ in $E(\mathbb{F}_{p^2})/2E(\mathbb{F}_{p^2})$. If the second argument is a point of order 2^{a-1}, then $e_{2^{a-1}}(P, Q)$ is of order 2^{a-2} in $\mathbb{F}_{p^2}$. Especially, if we consider the pairing of order $2^{a'}$ satisfying $2a' < a$, the value $e_{2^{a'}}(P, Q)$ is always equal to 1. Fortunately, we always handle the case $a' = a$ except for the last step of ideal to isogeny translation.

An alternative approach to compute pairings in Equation (3) is to utilize the Weil pairing:

$$\begin{aligned} e_{W,2^a}(P_2, P_1) &= \frac{f_{2^a,P_2}(P_1)}{f_{2^a,P_1}(P_2)}, \\ e_{W,2^a}(P_2, Q_2) &= \frac{f_{2^a,P_2}(Q_2)}{f_{2^a,Q_2}(P_2)}, \\ e_{W,2^a}(Q_2, P_1) &= \frac{f_{2^a,Q_2}(P_1)}{f_{2^a,P_1}(Q_2)}. \end{aligned} \tag{7}$$

Clearly, we need to construct three Miller functions. For the reduced Tate pairing computation, Miller function construction is more expensive than the final exponentiation. Furthermore, according to Equation (6), only two Miller function constructions are needed, while there are three Miller functions to be constructed in Equation (7). Therefore, the Weil pairing computation is still not as efficient as the reduced Tate pairing computation. But in parallel implementation the Weil pairing computation would be more competitive since it does not need the final exponentiation and all Miller function evaluations could be executed simultaneously. Another advantage compared to the reduced Tate pairing is that one can apply the Weil pairing into the situation when the order of the pairing is less than 2^a.

4.3 Discrete logarithm computations in μ_{2^a}

Since the order of μ_{2^a} is smooth, one can use the Pohlig-Hellman algorithm with an optimal strategy to translate discrete logarithms in μ_{2^a} into discrete logarithms in μ_{2^w}, where w is a small integer. It remains to compute discrete logarithms in μ_{2^w} efficiently. In this subsection, we first consider the two methods proposed in [28] which could be applied to improve the performance. Second, we give a novel approach to compute discrete logarithms when the storage is available. Finally, we give a comparison between the three methods by estimating the computational costs.

The authors in [28] proposed two methods to accelerate discrete logarithm computations. The first one is to compute a lookup table with respect to the base h_0:

$$T_1^{sgn}[r][c] = (h_0)^{(c+1)2^{wr+m}}, r = 0, 1, \cdots, \lfloor\frac{a}{w}\rfloor - 1, c = 0, 1, \cdots, 2^{w-1} - 1, \quad (8)$$

where $m \equiv a \bmod w$. Since h_0 is not fixed, we can not compute the lookup table in advance. As the base power w increases, the lookup table construction would be more expensive, and it would consume more storage at the same time, while the discrete logarithm computations would be more efficient.

The second method proposed in [28] is to compute only the first column and the last row of the lookup table in Equation (8):

$$\begin{aligned} FC &= \left\{T_1^{sgn}[r][0] = (h_0)^{2^{wr+m}}, i = 0, 1, \cdots, \lfloor\frac{a}{w}\rfloor - 1\right\}, \\ LR &= \left\{T_1^{sgn}[\lfloor\frac{a}{w}\rfloor - 1][c] = (h_0)^{(c+1)2^{a-w}}, c = 0, 1, \cdots, 2^{w-1}\right\}. \end{aligned} \quad (9)$$

The discrete logarithm computations with Equation (9) would be more expensive compared to that of the former method. However, the construction of Equation (9) is more efficient than the entire lookup table construction. Furthermore, the latter method would be preferred in storage restrained environments.

In the following, we give another effective approach to improve the performance of discrete logarithms in μ_{2^a}.

At first glance, as h_0 is not fixed, it seems that we can not use precomputation to save the computational cost. However, since the cyclotomic group μ_{2^a} in $\mathbb{F}_{p^2}$ is fixed, one can find a primitive element g of μ_{2^a} in advance. Instead of computing the two discrete logarithms of h_1, h_2 to the base h_0, we compute three discrete logarithms of h_0, h_1, h_2 to the base g:

$$h_0 = g^{s'_0}, h_1 = g^{s'_1}, h_2 = g^{s'_2}. \tag{10}$$

Hence, when the storage is available, we can use the precomputation to further speed up the discrete logarithm computations in Equation (10). In this case, the lookup table with respect to g is as follows:

$$T_1^{sgn}[r][c] = g^{(c+1)2^{wr+m}}, r = 0, 1, \cdots, \lfloor \frac{a}{w} \rfloor - 1, c = 0, 1, \cdots, 2^{w-1} - 1. \tag{11}$$

Note that h_0 is also a primitive element in μ_{2^a}. Therefore, we can recover the solutions by one inversion and two multiplications in $\mathbb{Z}/2^a\mathbb{Z}$:

$$s_1 = (s'_0)^{-1} s'_1, s_2 = (s'_0)^{-1} s'_1.$$

Compared with the first two methods, our method avoids the lookup table computation, but requires one more discrete logarithm computation in μ_{2^a}. We respectively estimate the computational costs by utilizing the three methods we presented above when setting the prime p as p_{3923} ($a = 65$). For simplicity, we only consider multiplications and squarings, and assume that the cost of one $\mathbb{F}_p$ multiplication is approximately equal to that of one $\mathbb{F}_p$ squaring. As shown in Table 1, when the base power is small, the previous methods proposed in [28] are more efficient than our new method. As the base power w increases, our new method saves more computational resources. When the storage is limited, one can adapt Method 2 proposed in [28] since it requires the least storage for the lookup table.

Method	$w = 1$	$w = 2$	$w = 3$	$w = 4$	$w = 5$	$w = 6$
Method 1 proposed in [28]	2068	1510	1219	1180	1171	1438
Method 2 proposed in [28]	2068	1560	1338	1375	1184	1389
Our method	2910	1980	1421	1179	855	825

Table 1: Cost estimates (in $\mathbb{F}_p$ multiplications) for the discrete logarithm computation by different methods.

Based on [28, Algorithm 6], we present Algorithm 6 to solve discrete logarithms. Since the algorithm is non-recursive, it would be more attractive in parallel environments.

Algorithm 6 PH_DLP(h, g, w, T_1^{sgn}, Str)

Require: The challenge h, a primitive element g in the multiplicative group μ_{2^a}, the base power w, the lookup table T_1^{sgn} in Equation (11), the optimal strategy Str.

Ensure: The array D such that $h = g^{\left(D[\lfloor \frac{a}{w} \rfloor - 1] \cdots D[1]D[0]\right)_{2^w}}$.

1: Initialize a Stack $Stack$, which contains tuples of the form (h_t, e_t, l_t), where $h_t \in \mu_{2^a}$, $e_t, l_t \in \mathbb{N}$.

$LR \leftarrow$ the last row of the lookup table T_1^{sgn}, $i \leftarrow 0$, $j \leftarrow 0$, $k \leftarrow 0$, $m \leftarrow 2^a \bmod w$, $h_t \leftarrow h$, $y \leftarrow 1$;
$h_t \leftarrow (h_t)^{2^m}$;
Push the tuple (h_t, j, k) into $Stack$;
while $k \neq \lfloor \frac{e_\ell}{w} \rfloor - 1$ **do**
  while $j + k \neq \lfloor \frac{e_\ell}{w} \rfloor - 1$ **do**
    $j \leftarrow j + Str[i]$;
    $h_t \leftarrow (h_t)^{2^{w \cdot Str[i]}}$;
    Push the tuple $(h_t, j + k, Str[i])$ into $Stack$;
    $i \leftarrow i + 1$;
  end while
  Pop the top tuple (h_t, e_t, l_t) from $Stack$;
  Find x_t such that $h_t = (LR[0])^{x_t}$ with the help of LR;
  $D[k] \leftarrow x_t$;
  for each tuple (h_t, e_t, l_t) in $Stack$ **do**
    if $x_t \neq 0$ **then**
      if $x_t > 0$ **then**
        $h_t \leftarrow h_t \cdot \overline{T_1^{sgn}[e_t][x_t - 1]}$;
      else
        $h_t \leftarrow h_t \cdot T_1^{sgn}[e_t][-x_t - 1]$;
      end if
    end if
  end for
  $j \leftarrow j - l_t$, $k \leftarrow k + 1$;
end while
Pop the top tuple (h_t, e_t, l_t) from $Stack$;
Find x_t such that $h_t = (LR[0])^{x_t}$ with the help of LR;
$D[k] \leftarrow x_t$;
if $m \neq 0$ **then**
  $y_0 \leftarrow g^{D[0]}$;
  for i_2 from 1 to $\lfloor \frac{e_\ell}{w} \rfloor - 1$ **do**
    if $D[i_2] < 0$ **then**
      $y \leftarrow y \cdot \overline{T_1^{sgn}[i_2 - 1][-D[i_2] - 1]}$;
    end if
    if $D[i_2] > 0$ **then**
      $y \leftarrow y \cdot T_1^{sgn}[i_2 - 1][D[i_2] - 1]$;
    end if
  end for
  $y \leftarrow y^{2^{w-m}}$;
  $y \leftarrow y_0 \cdot y$, $y \leftarrow h \cdot \overline{y}$;
  Find x_t such that $y = (LR[0])^{x_t}$ with the help of LR;
  $D[k+1] \leftarrow \frac{x_t}{2^{w-m}}$;
end if
return D.

5 Other Improvements

In this section, we propose other techniques to speed up the signing phase. Some of the improvements also benefit the performances of key generation and verification.

5.1 Torsion point generation

To accelerate torsion basis generation in compressed SIDH, Costello et al. [11] proposed a method to find out a torsion basis of $E(\mathbb{F}_{p^2})[2^a]$. The main idea is as follows: Firstly, precompute a list L of non-squares in $\mathbb{F}_{p^2}$. Then, randomly select $v_1 \in L$ until $v_1^3 + Av_1^2 + v_1$ is a square. It confirms that $(v_1, \sqrt{v_1^3 + Av_1^2 + v_1})$ is a point on $E(\mathbb{F}_{p^2})$. According to [23, Ch. 1((§4))], the order of the point is divided by 2^a, thus one can perform scalar multiplication to obtain a point P of order 2^a. Similarly, one can generate a point Q of order 2^a until $\langle P, Q\rangle = E(\mathbb{F}_{p^2})[2^a]$, which can be checked by $[2^{a-1}]P \neq [2^{a-1}]Q$. In this subsection, we will show how to adapt this method to benefit the implementation of SQISign.

As we all know, the $2^\bullet$-isogeny $\sigma \circ \varphi_{I_2}$ can be composed by multiple 2-isogenies. In addition, for any 2-isogeny ϕ whose kernel does not contain the point $(0, 0)$, i.e.,

$$\phi : (x, y) \mapsto (f(x), y \cdot f'(x)),$$

where

$$f(x) = x \cdot \left(\frac{x \cdot x_P - 1}{x - x_P}\right), \tag{12}$$

and $f'(x)$ is its derivative. It is easy to see that when applying the above formula (it is also what the current implementation did), we have

$$\phi((0, 0)) = (0, 0). \tag{13}$$

Furthermore, we can imply that the composition of them maps $(0, 0)$ on the original curve to $(0, 0)$ on the image curve. In the following, we will utilize this property to show that in each ideal to isogeny translation the first step of $\hat{\varphi}_J$ has kernel $\langle(0, 0)\rangle$, which confirms the point $P \in E[2^a] \cap \ker(\hat{\varphi}_J)$ has the property that $[2^{a-1}]P = (0, 0)$.

As we mentioned in Lemma 1, if the first step of φ_J corresponds to the ideal $\mathcal{O}_0(1+i)$, then it is an endomorphism of E_0. Besides, we have $\ker(1+i) = \langle(0, 0)\rangle$ and $1 + i$ maps $(\pm i, 0)$ to $(0, 0)$. Therefore, the dual of $1 + i$ has kernel $\langle(0, 0)\rangle$. Since the isogeny φ_J is cyclic, it is clear that the second step of φ_J is a 2-isogeny from E_0 to E_6 whose kernel is not $\langle(0, 0)\rangle$. According to Equation (12), the dual of the second step of φ_J has kernel $\langle(0, 0)\rangle$. From φ_{I_2} is cyclic, the group $\langle(0, 0)\rangle$ is not the kernel of the third step of φ_J. Analogously, one can deduce that except for the first step of φ_J, the kernels of all the other steps do not contain $(0, 0)$. It implies that from the second step, the image of $(0, 0)$ is equal to $(0, 0)$. Therefore, the dual of each step of φ_J has kernel $\langle(0, 0)\rangle$. In particular, we have $(0, 0) \in E[2^a] \cap \ker(\hat{\varphi}_J)$, i.e., the generator P of the group $E[2^a] \cap \ker(\hat{\varphi}_J)$

satisfies that $[2^{a-1}]P = (0,0)$. Likewise, it is easy to deduce $[2^{a-1}]P = (0,0)$ if the first step of φ_J is a 2-isogeny from E_0 to E_6.

Therefore, in each ideal to isogeny translation, the point P always satisfies that $[2^{a-1}]P = (0,0)$. Now we need another point Q such that $\langle P, Q\rangle = E(\mathbb{F}_{p^2})[2^a]$, i.e., $[2^{a-1}]Q \neq (0,0)$. Obviously, the above method presented by Costello et al. is exactly suitable for speeding up the generation of Q. Further, there is no need to check $[2^{a-1}]Q \neq (0,0)$ when applying this method since P and Q are always linearly independent, according to Theorem 3.

Theorem 3. *Assume that $E_A : y^2 = x^3 + Ax^2 + x$ is a supersingular elliptic curve defined on the finite field $\mathbb{F}_{p^2}$, where $2^a \| p+1$ and $E_A[2^a] \subseteq E_A(\mathbb{F}_{p^2})$. Suppose that $Q = (x_Q, y_Q) \in E_A(\mathbb{F}_{p^2})$ and denote $\mathrm{ord}(Q)$ the order of Q. If $2^a \| \mathrm{ord}(Q)$, then $(x_Q)^{\frac{p^2-1}{2}} = -1$ if any only if $\left[\frac{\mathrm{ord}(Q)}{2}\right] Q \neq (0,0)$.*

Proof. Suppose that P is a point of order 2^a defined on $E_A/\mathbb{F}_{p^2}$. Firstly, we prove that P and Q are linearly independent if and only if $e_{T,2^a}(P,Q)$ is a primitive element of the group μ_{2^a}.

Let $Q \in E_A(\mathbb{F}_{p^2})$ be a rational point such that P and Q are linearly independent. Suppose for contradiction that $e_{T,2^a}(P,Q)$ is not a primitive element of the group μ_{2^a}. Then we have

$$e_{T,2}([2^{a-1}]P, Q) = e_{T,2^a}(P,Q)^{2^{a-1}} = e_{T,2^a}\left(P, \left[\frac{\mathrm{ord}(Q)}{2^a}\right] Q\right)^{2^{a-1}} = 1, \qquad (14)$$

From Theorem 2 we can deduce that

$$e_{T,2}([2^{a-1}]P, P) = e_{T,2^a}(P,P)^{2^{a-1}} = e_{T,2^a}(P, [2^{a-1}]P) = 1. \qquad (15)$$

Since $E_A(\mathbb{F}_{p^2})/2E_A(\mathbb{F}_{p^2}) = \{\infty_{E_A} + 2E_A(\mathbb{F}_{p^2}), P + 2E_A(\mathbb{F}_{p^2}), Q + 2E_A(\mathbb{F}_{p^2}), P + Q + 2E_A(\mathbb{F}_{p^2})\}$, we have $e_{T,2}([2^{a-1}]P, R) = 1$ for any $R \in E_A(\mathbb{F}_{p^2})/2E_A(\mathbb{F}_{p^2})$. According to the non-degeneracy property of the reduced Tate pairing, $[2^{a-1}]P = \infty_{E_A}$. This is a contradiction and thus $e_{T,2^a}(P,Q)$ is a primitive element of the group μ_{2^a}.

On the other hand, if $e_{T,2^a}(P,Q)$ is of order 2^a, then

$$e_{T,2^a}(P,Q)^{2^{a-1}} = e_{T,2^a}(P, [2^{a-1}]Q) = e_{T,2^a}\left(P, \left[\frac{\mathrm{ord}(Q)}{2}\right] Q\right) = -1.$$

It follows from Equation (15) that $\left[\frac{\mathrm{ord}(Q)}{2}\right] Q \neq [2^{a-1}]P$. Hence, we can deduce that P and Q are linearly independent.

Now assume that $[2^{a-1}]P = (0,0)$. If $[\frac{\mathrm{ord}(Q)}{2}]Q \neq (0,0)$, then Q and P are linearly independent. Therefore, $e_{T,2^a}(P,Q)$ is a primitive element of μ_{2^a}, i.e.,

$$e_{T,2^a}(P,Q)^{2^{a-1}} = e_{T,2}((0,0), Q) = (x_Q)^{\frac{p^2-1}{2}} = -1. \qquad (16)$$

Conversely, if $(x_Q)^{\frac{p^2-1}{2}} = -1$, from Equation (16) we can imply that P and Q are linearly independent. It ensures that $[\frac{\mathrm{ord}(Q)}{2}]Q \neq (0,0)$. This completes the proof. $\square$

With Theorem 3, we can efficiently generate the point Q in Algorithm 7. It should be noted that this improvement benefits all the procedures of SQISign, especially the verifying phase.

Algorithm 7 DeterministicSecondPoint(A)

Require: The coefficient A of the Montgomery curve $E_A : y^2 = x^3 + Ax^2 + x$.
Ensure: A point Q defined on E_A of order 2^a such that $[2^{a-1}]Q \neq (0, 0)$.
1: Select a non-square element $x_Q \in \mathbb{F}_{p^2}$ such that $x_Q^3 + Ax_Q^2 + x_Q$ is a square;
2: $Q \leftarrow (x_Q, \sqrt{x_Q^3 + Ax_Q^2 + x_Q})$;
3: $Q \leftarrow [\frac{p+1}{2^a}]Q$;
4: **return** Q.

5.2 Image curve recovery with three points in isogeny computations

In each ideal to isogeny translation, we need to construct the large degree isogeny ϕ_2 and evaluate ϕ_2 at P, Q and $P + Q$. Invoking efficiency reasons, the computation of ϕ_2 is composed by multiple odd degree isogeny computations. At each odd degree isogeny computation, not only we need to evaluate it at the three points, but the image curve should also be obtained. Fortunately, one can use Equation (17), which is proposed in [12, Remark 4], to recover the image curve coefficient A:

$$A = \frac{(1 - x_{P'}x_{Q'} - x_{P'}x_{Q'-P'} - x_{Q'}x_{Q'-P'})^2}{4x_{P'}x_{Q'}x_{Q'-P'}} - x_{P'} - x_{P'} - x_{Q'-P'}, \qquad (17)$$

where P' and Q' are two points defined on the image curve. For large prime degree isogeny computations, applying Equation (17) to obtain the image curve coefficient is much more efficient than computing it with Vélu's formula [38,4]. This trick not only accelerates the performance of signing but that of key generation. Note that this technique could also be adapted in the implementation of other isogeny-based protocols, such as M-SIDH and MD-SIDH [20].

5.3 Precomputation for φ_1

In the first execution of ideal to isogeny translation for σ, we compute φ_1 with $\mathcal{O}_A$, I_2 and φ_{I_2}. All of these are obtained in the key generation phase, and thus some procedures used to compute φ_1 can be saved via precomputation in the key generation phase. For example, the endomorphism $\theta \in \mathcal{O}_A$ can be computed in advance, and hence we are able to evaluate $Q = \theta(P)$ before signing. Indeed, except for C and D, all the other information does not depend on the ideal I_σ. Therefore, one could precompute them to speed up the translation from the ideal $\langle I_\sigma, 2^a \rangle$ to the isogeny φ_1. Consequently, we can compute φ_1 efficiently by Algorithm 8, which avoids large degree isogeny computations. Although the precomputation increases the required computational resources of key generation, it reduces the signing cost.

Algorithm 8 FirstIdealToIsogenyEichler$_{2^a}(\mathcal{O}_A, I, K, P, \theta, Q)$

Require: A left $\mathcal{O}_A$-ideal I of reduced norm 2^a, a left $\mathcal{O}_A$-ideal $K = \overline{I_2} + 2\mathcal{O}_A$, a generator P of $E[2^a] \cap \ker(\hat{\varphi}_{I_2})$, an endomorphism $\theta \in \mathcal{O}_A \backslash (\mathbb{Z} + K)$ and the point $Q = \theta(P)$.

Ensure: φ_1 of degree 2^a.

1: Select $\alpha \in I$ such that $I = \mathcal{O}\langle \alpha, 2^a \rangle$;

2: Compute C, D such that $\alpha(C + D\theta) \in K$ and $\gcd(C, D, 2) = 1$;

3: Compute φ_1 of kernel $\langle [C]P + [D]Q \rangle$;

4: **return** φ_1.

---

6 Implementation Results

In this section, we present the implementation results of the procedures we have improved in the signing phase, and report the performance of SQISign with our techniques. We also give a concrete comparison between the previous work and ours on efficiency. Based on the code[1] provided in [17], we compile and benchmark our code on Intel(R) Core(TM) i9-12900K 3.20 GHz with TurboBoost and hyperthreading features disabled. Except for the improvements we mentioned in this paper, we also adapt some techniques proposed in the literature to further improve the implementation. For example, one can adapt the three-point ladder algorithm [18] when computing the kernel generator of the isogeny.

Table 2 reports the performance of the procedures we improved in the signing phase. For elliptic curve discrete logarithm computations, we apply our new method to compute discrete logarithms in the group μ_{2^a} and set the base power $w = 5$. The results show that the performance are significantly accelerated with our techniques. It should be noted that except for the ideal generation with the improved SigningKLPT algorithm, all the other procedures are executed multiple times during the key generation and signing phase. In addition, the verification phase needs to generate the second torsion point frequently, thus our improved algorithms for torsion point generation also save the verifying cost.

As shown in Table 3, we improve the performance of all the procedures in SQISign without the technique proposed in Section 5.3. When using the precomputation technique, the key generation phase is less efficient, but we further improve the signing phase. In particular, the signing performance is 11.93% faster than that of the previous work. This would be preferred in the case when the signer needs to sign a number of messages using the same secret key.

[1] `https://github.com/SQISign/sqisign-ec23`

Phase	p_{6983}			p_{3923}		
	SQISign2 [17]	This work	Speedup	SQISign2 [17]	This work	Speedup
Generation of I_σ	58550	37331	36.2%	55614	37927	31.8%
Computations for s_1 and s_2	4177	1260	69.8%	6149	1287	79.1%
Torsion point generation	881	604	31.4%	605	400	33.9%
Isogeny computation of φ_2	30254	27439	9.3%	23872	21628	9.4%

Table 2: Implementation results of the improved procedures in the signing phase of SQISign. The results are expressed in thousands of clock cycles.

Setting	Phase	SQISign2 [17]	This work			
			without precomp.	Speedup	precomp.	Speedup
p_{6983}	Keygen	2029.1	1965.0	3.16%	2039.1	-0.5%
	Sign	3671.1	3398.5	7.43%	3312.5	9.77%
	Verify	50.2	38.6	23.11%	38.6	23.11%
p_{3923}	Keygen	361.5	329.6	8.82%	399.6	-10.54%
	Sign	1677.7	1535.1	8.50%	1477.6	11.93%
	Verify	26.4	21.4	18.94%	21.4	18.94%

Table 3: Implementation results of each phases in SQISign. The results are reported in millions of clock cycles. We execute 1000 times and record the average costs of key generation and signature. For verification we report the average of 2500 instances.

7 Conclusion

In this paper, we mainly focused on ideal to isogeny translation in the signing phase of SQISign, and proposed several novel techniques to enhance its performance. For each procedure we have considered, the improvements led to a significant speedup. The implementation results showed that we also improved the key generation phase and the verification phase of SQISign. As a future work, we would like to explore how to further accelerate the implementation of SQISign.

Acknowledgments

We thank Jintai Ding for his valuable comments and proofreading an earlier version of this work. This work is supported by Guangdong Major Project of Basic and Applied Basic Research (No. 2019B030302008), the National Natural Science Foundation of China (No. 61972428).

References

1. Atapoor, S., Baghery, K., Cozzo, D., Pedersen, R.: CSI-SharK: CSI-FiSh with Sharing-friendly Keys. Cryptology ePrint Archive (2022)
2. Azarderakhsh, R., Campagna, M., Costello, C., De Feo, L., Hess, B., Hutchinson, A., Jalali, A., Jao, D., Karabina, K., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Pereira, G., Renes, J., Soukharev, V., Urbanik, D.: Supersingular Isogeny Key Encapsulation (2020), `http://sike.org`
3. Azarderakhsh, R., Jao, D., Kalach, K., Koziel, B., Leonardi, C.: Key Compression for Isogeny-Based Cryptosystems. In: Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography. pp. 1–10 (2016)
4. Bernstein, D.J., de Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. In: Galbraith, S. (ed.) ANTS-XIV - 14th Algorithmic Number Theory Symposium. Proceedings of the Fourteenth Algorithmic Number Theory Symposium (ANTS-XIV), vol. 4, pp. 39–55. Mathematical Sciences Publishers, Auckland, New Zealand (Jun 2020). https://doi.org/10.2140/obs.2020.4.39, `https://hal.inria.fr/hal-02514201`
5. Bernstein, D.J., Lange, T.: Explicit-formulas database, `http://www.hyperelliptic.org/EFD`
6. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: efficient isogeny based signatures through class group computations. In: Advances in Cryptology–ASIACRYPT 2019: 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8–12, 2019, Proceedings, Part I. pp. 227–247. Springer (2019)
7. Castryck, W., Decru, T.: An Efficient Key Recovery Attack on SIDH. In: Advances in Cryptology–EUROCRYPT 2023: 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V. pp. 423–447. Springer (2023)
8. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III 24. pp. 395–427. Springer (2018)
9. Chi-Domínguez, J.J.: A Note on Constructing SIDH-PoK-based Signatures after Castryck-Decru Attack. Cryptology ePrint Archive, Paper 2022/1479 (2022), `https://eprint.iacr.org/2022/1479`
10. Colò, L., Kohel, D.: Orienting supersingular isogeny graphs. Journal of Mathematical Cryptology **14**(1), 414–437 (2020)
11. Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient Compression of SIDH Public Keys. In: Coron, J.S., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017. pp. 679–706. Springer International Publishing, Cham (2017)
12. Costello, C., Longa, P., Naehrig, M.: Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016. pp. 572–601. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
13. Dartois, P., Leroux, A., Robert, D., Wesolowski, B.: SQISignHD: New Dimensions in Cryptography. Cryptology ePrint Archive, Paper 2023/436 (2023), `https://eprint.iacr.org/2023/436`

14. De Feo, L., Dobson, S., Galbraith, S.D., Zobernig, L.: SIDH proof of knowledge. In: Advances in Cryptology–ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part II. pp. 310–339. Springer (2023)
15. De Feo, L., Galbraith, S.D.: SeaSign: compact isogeny signatures from class group actions. In: Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38. pp. 759–789. Springer (2019)
16. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: Compact Post-quantum Signatures from Quaternions and Isogenies. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020. pp. 64–93. Springer International Publishing, Cham (2020)
17. De Feo, L., Leroux, A., Longa, P., Wesolowski, B.: New Algorithms for the Deuring Correspondence: Towards Practical and Secure SQISign Signatures. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 659–690. Springer Nature Switzerland, Cham (2023)
18. Faz-Hernández, A., López, J., Ochoa-Jiménez, E., Rodríguez-Henríquez, F.: A Faster Software Implementation of the Supersingular Isogeny Diffie-Hellman Key Exchange Protocol. IEEE Transactions on Computers **67**(11), 1622–1636 (2018)
19. Fiat, A., Shamir, A.: How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) Advances in Cryptology — CRYPTO' 86. pp. 186–194. Springer Berlin Heidelberg, Berlin, Heidelberg (1987)
20. Fouotsa, T.B., Moriya, T., Petit, C.: M-SIDH and MD-SIDH: Countering SIDH Attacks by Masking Information. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 282–309. Springer Nature Switzerland, Cham (2023)
21. Galbraith, S.: Pairings, pp. 183–214. London Mathematical Society Lecture Note Series, Cambridge University Press (2005)
22. Galbraith, S.D., Petit, C., Silva, J.: Identification protocols and signature schemes based on supersingular isogeny problems. In: Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23. pp. 3–33. Springer (2017)
23. Husemöller, D.: Elliptic Curves. Graduate Texts in Mathematics 111, Springer New York, 2nd ed edn. (2004)
24. Jao, D., De Feo, L.: Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies. In: Yang, B.Y. (ed.) Post-Quantum Cryptography. pp. 19–34. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
25. Kohel, D., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion ℓ-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
26. Li, S., Ouyang, Y., Xu, Z.: Neighborhood of the supersingular elliptic curve isogeny graph at j= 0 and 1728. Finite Fields and Their Applications **61**, 101600 (2020)
27. Lin, K., Lin, J., Wang, W., Zhao, C.A.: Faster Public-key Compression of SIDH with Less Memory. IEEE Transactions on Computers pp. 1–9 (2023). https://doi.org/10.1109/TC.2023.3259321
28. Lin, K., Wang, W., Wang, L., Zhao, C.A.: An Alternative Approach for Computing Discrete Logarithms in Compressed SIDH. Cryptology ePrint Archive, Paper 2021/1528 (2021), `https://eprint.iacr.org/2021/1528`

29. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: Advances in Cryptology–EUROCRYPT 2023: 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V. pp. 448–471. Springer (2023)
30. Naehrig, M., Renes, J.: Dual Isogenies and Their Application to Public-Key Compression for Isogeny-Based Cryptography. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019. pp. 243–272. Springer International Publishing, Cham (2019)
31. Onuki, H.: On oriented supersingular elliptic curves. Finite Fields and Their Applications **69**, 101777 (2021)
32. Pizer, A.K.: Ramanujan graphs and Hecke operators. Bulletin of the American Mathematical Society **23**(1), 127–137 (1990)
33. Pohlig, S., Hellman, M.: An improved algorithm for computing logarithms over GF(p) and its cryptographic significance (Corresp.). IEEE Transactions on Information Theory **24**(1), 106–110 (1978)
34. Richard Crandall, C.B.P.: Prime numbers: a computational perspective. Springer, 2nd ed edn. (2005)
35. Robert, D.: Breaking SIDH in polynomial time. In: Advances in Cryptology–EUROCRYPT 2023: 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part V. pp. 472–503. Springer (2023)
36. Scott, M., Barreto, P.S.L.M.: Compressed Pairings. In: Franklin, M. (ed.) Advances in Cryptology – CRYPTO 2004. pp. 140–156. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
37. Silverman, J.H.: The Arithmetic of Elliptic Curves, 2nd Edition. Graduate Texts in Mathematics. Springer (2009)
38. Vélu, J.: Isogénies entre courbes elliptiques. C. R. Acad. Sci., Paris, Sér. A **273**, 238–241 (1971)
39. Voight, J.: Quaternion algebras. Springer Graduate Texts in Mathematics series. (2021)