On hybrid SIDH schemes using Edwards and Montgomery curve arithmetic

Michael Meyer[1,2], Steffen Reith[1], and Fabio Campos[1]

[1]Department of Computer Science, University of Applied Sciences Wiesbaden
[2]Department of Mathematics, University of Würzburg

Abstract

Supersingular isogeny Diffie-Hellman (SIDH) is a proposal for a quantum-resistant key exchange. The state-of-the-art implementation works entirely with Montgomery curves and basically can be divided into elliptic curve arithmetic and isogeny arithmetic. It is well known that twisted Edwards curves can provide a more efficient elliptic curve arithmetic. Therefore it was hinted by Costello and Hisil, that by using only Edwards curves for isogeny and curve arithmetic, or a hybrid scheme, that uses Edwards curve arithmetic and switches between the models whenever needed, a speedup in the computation may be gained.

Following the latter case, we investigated how to efficiently switch between Montgomery and twisted Edwards curves in SIDH, and how to insert Edwards arithmetic in the current state-of-the-art implementation.
We did not gain a speedup compared to the results of Costello, Longa, and Naehrig, but in some cases the performance of Edwards arithmetic is almost equally fast. Thus, we suppose that a hybrid scheme does not improve the performance of SIDH, but still can be interesting for platforms having special hardware acceleration for Edwards curves. However, a full Edwards SIDH version may give a speedup, if fast Edwards isogeny formulas can be found.

1 Introduction

In the recent years, the topic of post-quantum cryptography (PQC) has gained a massive boost of attention and research. The threat of the possibility of building large quantum computers soon, that could break e.g. elliptic curve cryptography like ECDH by Shor's algorithm (see [12]), led to a competition by the NIST for the

This work was partially supported by Elektrobit.

standardization of PQC primitives (see [10]).

One proposal for a PQC key establishment protocol is based on isogenies between elliptic curves. First proposed by Rostovtsev and Stolbunov in 2006 (see [11]), using ordinary elliptic curves, Jao and De Feo (see [8]) proposed the use of supersingular elliptic curves in 2011, in order to obtain a quantum-resistant scheme. Since 2016, when Costello, Longa and Naehrig published an efficient algorithm for the isogeny-based key exchange (see [6]), SIDH has gained a lot of attention and research. The use of twisted Edwards curves has been suggested by Costello and Hisil in [5].

2 Preliminaries

In the SIDH key exchange primitive, isogenies of large degree are computed as a composition of small degree isogenies. We work over a field $\mathbb{F}_{p^2}$ with a prime of the form $2^m 3^n f \pm 1$ with a small integer f. We choose an initial supersingular elliptic curve E_0 over $\mathbb{F}_{p^2}$ and want to compute a 2^m- and 3^n-isogeny respectively for each party, in the following called Alice and Bob. Therefore for generating the kernels of the isogenies, whose sizes determine the degrees of the isogenies, we choose initial points P_A, Q_A, P_B, Q_B lying in the corresponding torsion groups, namely $P_A, Q_A \in E_0[2^m]$ and $P_B, Q_B \in E_0[3^n]$.
In the following, we describe Alice's key generation, while Bob's key generation works in an analogous way.
Alice chooses a random integer m_A and computes $R_0 = P_A + [m_A]Q_A$ as generator of the kernel for computing her 2^m-isogeny. However, since computing large degree isogenies is expensive, m isogenies of degree 2 are computed. Therefore, we compute $[2^{m-1}]R_0$ as generator of the kernel of the 2-isogeny φ_0, that maps to $E_1 = E_0/\langle [2^{m-1}]R_0 \rangle$. φ_0 can be computed by Vélu's formulas (see [13]). In addition, $R_1 = \varphi_0(R_0)$ is computed. Following this approach, the next step is to compute $[2^{m-2}]R_1$, φ_1, $E_2 = E_1/\langle [2^{m-2}]R_1 \rangle$, and $R_2 = \varphi_1(R_1)$.
Following this pattern, we obtain a 2^m-isogeny φ_A as composition of m isogenies of degree 2, that maps from E_0 to a curve E_A.
The public key then consists of E_A (in terms of the curve parameters), $\varphi_A(P_B)$, and $\varphi_A(Q_B)$. Alice receives Bob's computed public key, which contains the curve E_B, $\varphi_B(P_A)$, and $\varphi_B(Q_A)$.
Following the same strategy again, Alice then computes $R_{BA} = \varphi_B(P_A) + [m_A]\varphi_B(Q_A)$ and uses this point as generator for computing the 2^m-isogeny φ_{BA}, again as a composition of 2-isogenies, that maps from E_B to E_{BA}. Similarly, Bob obtains a 3^n-isogeny φ_{AB}, that maps from E_A to E_{AB}. The shared secret can then be computed, since the j-invariants of the resulting curves E_{AB} and E_{BA} are equal.

Instead of a field $\mathbb{F}_{p^2}$ with a prime of the form $2^m 3^n f \pm 1$, any other prime of the form $\ell_a^m \ell_b^n f \pm 1$ with ℓ_a, ℓ_b coprime can be used. However, the above mentioned choice seems to be the most efficient for SIDH. Anyway, in [6] 4-isogenies are used instead of 2-isogenies.

Furthermore, we note that there are better strategies to obtain isogenies of degree ℓ^m than described above. See [8] for a detailed analysis of optimal strategies.

3 Montgomery Arithmetic

In the current state-of-the-art implementation of SIDH in [6], Costello, Longa and Naehrig use elliptic curves in Montgomery form. They are given by an equation over a field K of the form

$$E_{a,b} : by^2 = x^3 + ax^2 + x.$$

To avoid inversions during point additions and doublings, projective coordinates can be used. Instead of the affine coordinates $(x, y) \in E_{a,b}$, we use $(X : Y : Z) \in \mathbb{P}^2$ with $x = X/Z$ and $y = Y/Z$, and $\mathcal{O}_E = (0 : 1 : 0)$ as point at infinity. If we embed this into $\mathbb{P}^1$ by dropping the Y-coordinate, we can use the efficient arithmetic given by Montgomery in [9]. Given a point $P_n = (X_n : Z_n)$, we can compute $[2]P_n = (X_{2n} : Z_{2n})$ by

$$\begin{aligned}
&4X_nZ_n = (X_n + Z_n)^2 - (X_n - Z_n)^2,\\
&X_{2n} = (X_n + Z_n)^2(X_n - Z_n)^2,\\
&Z_{2n} = (4X_nZ_n)((X_n - Z_n)^2 + ((A + 2)/4)(4X_nZ_n)).
\end{aligned}$$

Given another point $P_m = (X_m : Z_m)$ and the difference $P_{m-n} = P_m - P_n = (X_{m-n} : Z_{m-n})$, we can compute the sum $P_{m+n} = P_m + P_n = (X_{m+n} : Z_{m+n})$ by

$$\begin{aligned}
X_{m+n} &= Z_{m-n}((X_m - Z_m)(X_n + Z_n) + (X_m + Z_m)(X_n - Z_n))^2,\\
Z_{m+n} &= X_{m-n}((X_m - Z_m)(X_n + Z_n) - (X_m + Z_m)(X_n - Z_n))^2.
\end{aligned}$$

Thus an addition can be done using $4\mathbf{M} + 2\mathbf{S}$, or $3\mathbf{M} + 2\mathbf{S}$ if P_{m-n} is normalized, while a doubling needs $2\mathbf{M} + 2\mathbf{S} + 1\mathbf{C}$. As usual, we denote field multiplications by $\mathbf{M}$, field squarings by $\mathbf{S}$, and multiplications by constants by $\mathbf{C}$. We note that the formulas above lose information, since we do not distinguish between the possible corresponding coordinates Y and $-Y$. However, in the case of SIDH we do not need this information and it suffices to work entirely with $(X : Z) \in \mathbb{P}^1$.

In the SIDH implementation in [6], not only the point coordinates, but also the curve parameters are projectivized. Instead of a Montgomery curve of the form given above, we work with an equation of the form

$$E_{(A:B:C)} : By^2 = Cx^3 + Ax^2 + Cx,$$

where $(A : B : C) \in \mathbb{P}^2(K)$ such that $a = A/C$ and $b = B/C$ for the correspondig curve $E_{a,b}$. The j-invariants of the curves are then given by

$$j(E_{a,b}) = \frac{256(a^2 - 3)^3}{a^2 - 4}, \quad \text{and} \quad j(E_{(A:B:C)}) = \frac{256(A^2 - 3C^2)^3}{C^4(A^2 - 4C^2)}.$$

From these formulas we see that the j-invariant does not depend upon b or B, respectively. Therefore, it suffices to work with $(A : C) \in \mathbb{P}^1(K)$ in the projective model. Furthermore, neither the Montgomery arithmetic given above, nor the isogeny computations during SIDH require the coefficients b or B, respectively. However, we note that formulas that make use of the parameter a, like the Montgomery doubling above, must be trivially modified by substituting $a = A/C$ in order to work on $E_{(A:B:C)}$, as described in [6]. In this case a doubling costs $2\mathbf{M} + 2\mathbf{S} + 2\mathbf{C}$.

4 Edwards Arithmetic

When it comes to elliptic curve computations, Edwards curves are often said to be a model equipped with fast arithmetic. However, in the context of SIDH, we have to compare the Edwards arithmetic formulas with the efficient XZ-only Montgomery arithmetic. On the other hand, most of the discussions about fast Edwards arithmetic focus on full coordinate models.

Twisted Edwards curves over K are given by equations of the form

$$E_{E,a,d} : aX^2 + Y^2 = 1 + dX^2Y^2,$$

with $a, d \neq 0$, $d \neq 1$, and $a \neq d$. For $a = 1$, the twisted Edwards curve $E_{E,1,d} = E_{E,d}$ is called Edwards curve. As seen in the Montgomery case, projective coordinates can be used in order to avoid inversions during additions and doublings. However, in the Edwards case there are three more models, as described in [2].

Similarly to the Montgomery case above, we can use projective coordinates $(X : Y : Z) \in \mathbb{P}^2$ with $x = X/Z$ and $y = Y/Z$ for the corresponding affine point (x, y) on $E_{a,d}$. The projective curve equation is given by $aX^2Z^2 + Y^2Z^2 = Z^4 + dX^2Y^2$. A projective point $P = (X_1 : Y_1 : Z_1)$ can be doubled using

$$\begin{array}{l} B = (X_1 + Y_1)^2; \quad C = X_1^2; \quad D = Y_1^2; \quad E = aC; \\ F = E + D; \quad H = Z_1^2; \quad J = F - 2H; \\ X_3 = (B - C - D) \cdot J; \quad Y_3 = F \cdot (E - D); \quad Z_3 = F \cdot J, \end{array}$$

where $[2]P = (X_3 : Y_3 : Z_3)$. Therefore, a doubling needs $3\mathbf{M} + 4\mathbf{S} + 1\mathbf{C}$ (see [1]).

Another model for twisted Edwards curves is given by the extended curve equation $aX^2 + Y^2 = Z^2 + dT^2$ with $XY = ZT$. Points on this curve are represented by $(X : Y : Z : T) \in \mathbb{P}^3$, where the corresponding affine point (x, y) is represented by $(x : y : 1 : xy)$. According to [7], there is a fast way to add two points $(X_1 : Y_1 : Z_1 : T_1)$ and $(X_2 : Y_2 : Z_2 : T_2)$ by

$$\begin{array}{l} A = X_1 \cdot X_2; \quad B = Y_1 \cdot Y_2; \quad C = Z_1 \cdot T_2; \quad D = T_1 \cdot Z_2; \quad E = D + C; \\ F = (X_1 - Y_1) \cdot (X_2 + Y_2) + B - A; \quad G = B + aA; \quad H = D - C; \\ X_3 = E \cdot F; \quad Y_3 = G \cdot H; \quad Z_3 = F \cdot G; \quad T_3 = E \cdot H, \end{array}$$

where $(X_1 : Y_1 : Z_1 : T_1) + (X_2 : Y_2 : Z_2 : T_2) = (X_3 : Y_3 : Z_3 : T_3)$. The cost of this computation is $9\mathbf{M} + 1\mathbf{C}$.

Two more models are given by the inverted curve $aY^2Z^2 + X^2Z^2 = X^2Y^2 + dZ^4$ with points $(X : Y : Z) \in \mathbb{P}^2$ and the completed curve $aX^2T^2 + Y^2Z^2 = Z^2T^2 + dX^2Y^2$ with points $((X : Z), (Y : T)) \in \mathbb{P}^1 \times \mathbb{P}^1$ (see [2]). However, for these models there are no known doubling or addition formulas, that are more efficient than the ones mentioned above.

Similar to the XZ-only Montgomery arithmetic, Castryck et al. introduced a YZ-only doubling formula for Edwards curves in [4]. In the same way as presented in the mentioned paper, we can derive such a formula for twisted Edwards curves. For an affine point $P = (x, y)$, the twisted Edwards doubling formula is given by

$$[2]P = (x_2, y_2) = \left(\frac{2xy}{1 + dx^2y^2}, \frac{y^2 - ax^2}{1 - dx^2y^2}\right)$$

For the y-coordinate we have

$$\frac{y^2 - ax^2}{1 - dx^2y^2} = \frac{y^2(a - dy^2) - a(1 - y^2)}{(a - dy^2) - dy^2(1 - y^2)} = \frac{-dy^4 + 2ay^2 - a}{dy^4 - 2dy^2 + a},$$

where we make use of the curve equation $ax^2 + y^2 = 1 + dx^2y^2$. Expressing this in projective coordinates with $P = (X : Y : Z)$ and $[2]P = (X_2 : Y_2 : Z_2)$, we obtain

$$\begin{aligned} Y_2 &= -dY^4 + 2aY^2Z^2 - aZ^4, \text{ and} \\ Z_2 &= dY^4 - 2dY^2Z^2 + aZ^4. \end{aligned}$$

This can be computed using $5\mathbf{S} + 4\mathbf{C}$. In the case of an Edwards curve we have $a = 1$, so the cost decreases by $2\mathbf{C}$. In [4] it is also pointed out that if ad is a square and $\sqrt{ad}$ is known, a doubling can be modified to cost $1\mathbf{M} + 3\mathbf{S} + 6\mathbf{C}$. Similarly, in the Edwards case, if d is a square and $\sqrt{d}$ is known, a doubling takes $1\mathbf{M}+3\mathbf{S}+3\mathbf{C}$.

As described in [1], the j-invariant of a twisted Edwards curve is given by

$$\frac{16(a^2 + 14ad + d^2)^3}{ad(a - d)^4},$$

which shows that in this case, we need both parameters a and d for the computation of the j-invariant.

5 Switching between Montgomery and twisted Edwards curves

If we want to combine some of the ideas for Montgomery and twisted Edwards curves from above, we need an efficent way to switch between these two models.

We slightly change our notation here in the following way: For a Montgomery curve over a field K with $A \in K\backslash\{-2, 2\}$ and $B \in K\backslash\{0\}$ we write

$$E_{M,A,B} : Bv^2 = u^3 + Au^2 + u.$$

For a twisted Edwards curve with distinct $a, d \in K\backslash\{0\}$ and $d \neq 1$ we write

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

Then it is shown in [1] that $E_{E,a,d}$ is birationally equivalent to $E_{M,A,B}$, where

$$A = \frac{2(a+d)}{a-d} \quad \text{and} \quad B = \frac{4}{a-d},$$

and a birational equivalence is given by the map

$$(x, y) \mapsto (u, v) = \left(\frac{1+y}{1-y}, \frac{1+y}{(1-y)x}\right)$$

and its inverse

$$(u, v) \mapsto (x, y) = \left(\frac{u}{v}, \frac{u-1}{u+1}\right).$$

Note that these maps are not defined everywhere. For a way to handle exceptional points, we refer to [1].

However, if we use projective coordinates, particularly the XZ-only Montgomery arithmetic and the YZ-only twisted Edwards arithmetic, switching between these models is very simple. As seen in [4], a Montgomery point $(X_M : Z_M)$ can be transformed to the corresponding Edwards YZ-coordinates $(Y_E : Z_E)$ by the map

$$(X_M : Z_M) \mapsto (Y_E : Z_E) = (X_M - Z_M : X_M + Z_M).$$

A twisted Edwards point $(Y_E : Z_E)$ can be transformed to the corresponding Montgomery XZ-coordinates $(X_M : Z_M)$ by the map

$$(Y_E : Z_E) \mapsto (X_M : Z_M) = (Y_E + Z_E : Z_E - Y_E).$$

Therefore, switching between these two models costs only two additions.

6 Elliptic curve arithmetic in SIDH

There are different stages of SIDH, where elliptic curve arithmetic takes place. In [6], all the arithmetic is done in XZ-only Montgomery coordinates.

1. During the key generation, Alice and Bob compute $P_i + [m_i]Q_i$ for some initially chosen points P_i and Q_i and a random integer m_i. Thus, the starting curve parameters and full coordinates of P_i and Q_i are known.

2. During the computation of the isogenies, Alice and Bob have to compute several doublings or triplings, respectively. Since we work on different curves after each computed isogeny, the curve parameters are not fixed here. Furthermore, only the X- and Z-coordinates of the points R_i that have to be doubled, or tripled, respectively, are known.
3. During the computation of the shared secret, Alice and Bob again have to compute $\hat{P}_i + [m_i]\hat{Q}_i$ for their secret integer m_i and some received points $\hat{P}_i$ and $\hat{Q}_i$ from the public key. Here, only the normalized curve parameter A and the normalized X-coordinates of $\hat{P}_i$, $\hat{Q}_i$, and their difference $\hat{P}_i - \hat{Q}_i$ are known.

7 Edwards arithmetic in SIDH

Stage 1

The situation of Alice and Bob is equal here, except for the different initial points provided and the probably different random numbers. Therefore, we don't have to distinguish between the two parties here. In the implementation from [6], we start with the Montgomery curve $y^2 = x^3 + x$ and the computation of $[m]Q$ is done using a Montgomery ladder. Per bit of the integer m, one doubling and one addition are performed in Montgomery XZ-only arithmetic. Since the computation entirely takes place in the basefield $\mathbb{F}_p$ and because of the choice of the curve parameters, the cost of this is $5\mathbf{M} + 4\mathbf{S}$ per bit of m.

As seen above, and mentioned in [4], we can replace each doubling of a point Q by a doubling in Edwards coordinates:

1. Compute the corresponding Edwards point Q_E,
2. Compute $[2]Q_E$ by a twisted Edwards doubling,
3. Switch back to Montgomery coordinates to obtain $[2]Q$.

However, in the context of SIDH, this can be optimized as follows: Since the Montgomery parameters of the starting curve are $A = 0$ and $B = 1$, the corresponding Edwards parameters are $a = 2$ and $d = -2$. Therefore, all multiplications by curve parameters can be replaced by additions. Plugging the parameters into the doubling formulas, we obtain

$$Y_2 = 2Y^4 + 4Y^2Z^2 - 2Z^4 \quad \text{and} \quad Z_2 = -2Y^4 + 4Y^2Z^2 + 2Z^4.$$

Instead of computing the Edwards doubling and transforming back to Montgomery coordinates afterwards by computing $Y_2 + Z_2$ and $Z_2 - Y_2$, we can combine these steps, since

$$Y_2 + Z_2 = 2Y^2Z^2 \quad \text{and} \quad Z_2 - Y_2 = Z^4 - Y^4,$$

so we don't have to compute Y_2 and Z_2 explicitly. Therefore, we save a few additions. Furthermore, we get the transformation to Edwards coordinates for free,

since in each ladder step, a Montgomery differential addition is performed, during which the required values occur. In total, the Edwards doubling costs $5\mathbf{S}$ here, and the combined Montgomery differential addition and Edwards doubling, and hence one step in the ladder, costs $3\mathbf{M} + 7\mathbf{S}$.

A different approach to compute a multiple $[m]Q$ of a point by Edwards arithmetic is given in the context of the elliptic curve method for factorization in [2], where multiples of points are computed in full coordinates. The fastest way described there is the combination of the doubling in projective coordinates and the addition in extended coordinates that are stated above. Bernstein et al. use 'signed sliding fractional window' addition-subtraction chains that are defined in [3], that define the sequence of doublings and additions for a fast computation of $[m]Q$. Using such a chain, only one doubling and ε additions are required per bit of m, where ε converges to 0 for increasing bitlength of m. For Alice, the random integer m has a maximal bitlength of 372. We see from [2], that a bitlength in this magnitude requires approximately 0.99 doublings and 0.19 additions per bit. Thus, calculating with a bitlength of 372, we end up with roughly $1740\mathbf{M} + 1473\mathbf{S}$ in total for the computation of $[m]Q$, ignoring further computations for switching between the coordinate models. In comparison, the ladder from [6] needs $5\mathbf{M} + 4\mathbf{S}$ per bit of m, and therefore a total of $1860\mathbf{M} + 1488\mathbf{S}$. However, since m is randomly chosen each time, we always have to compute a fast chain, and in addition, more storage is required for the chain and the saved intermediate values.

Stage 2

In this stage of the algorithm, the situation of Alice and Bob is slightly different, since they have to compute doublings or triplings, respectively. We first focus on Alice's computations, where doublings are needed. The first thing to note here is that in every step, we work on a new curve with different parameters, and therefore a multiplication by a curve coefficient costs as much as a general field multiplication $\mathbf{M}$, since we cannot expect the parameters to stay small. For a YZ- only Edwards doubling we thus need $4\mathbf{M} + 5\mathbf{S}$. A doubling in full projective coordinates would be slightly cheaper, using $4\mathbf{M}+4\mathbf{S}$, but we don't have the X-coordinate here. Although it is clear, that compared to the Montgomery arithmetic from [6], the computation is more expensive this way, we show how it can be done.

For using these formulas, we first have to recover the corresponding Edwards parameters, in the following denoted by a_E and d_E. Since in the implementation of [6], the curve parameters are in projective form $(A : C)$, where the usual Montgomery parameter $a = A/C$, and b, or B, respectively, is discarded completely, it is not possible to use the formulas above directly to recover the Edwards parameters. However, we can rewrite them as

$$\frac{A}{C} = \frac{2(a_E + d_E)}{a_E - d_E}, \quad \text{and} \quad \frac{B}{C} = \frac{4}{a_E - d_E}.$$

We can then fix $B = 1$ and thus obtain

$$a_E = A + 2C, \quad \text{and} \quad d_E = A - 2C.$$

Therefore, all the doublings in this section of the algorithm can be done using Edwards coordinates in the following way:

1. Compute the corresponding Edwards point (cost: 2 additions)
2. Recover the Edwards parameters a_E and d_E (cost: 3 additions)
3. Compute all the required doublings (cost: $4\mathbf{M} + 5\mathbf{S}$ each)
4. Transform the resulting point back into Montgomery coordinates (cost: 2 additions)

On Bob's side, we have to compute triplings. For using Edwards arithmetic, we have to combine the Edwards doubling with the Montgomery differential addition, leading to a cost of $8\mathbf{M} + 7\mathbf{S}$ per tripling. This is again more expensive than the tripling from [6], which costs $8\mathbf{M} + 4\mathbf{S}$.

Stage 3

Similar to stage 1, we want to compute $P + [m]Q$ here. However the circumstances are different here. We are given the public keys, namely the normalized X-coordinates of P, Q and $P - Q$, and the normalized Montgomery curve parameter A, which is calculated from these values. In [6], the three-point-ladder from [8] is used, which computes one differential addition and one combined doubling and differential addition per step.
For the deployment of Edwards curves, we can again replace the doubling of the combined step by an Edwards doubling. However, in contrast to stage 1, we cannot expect the coefficients $a_E = A + 2$ and $d_E = A - 2$ to be small, so we have to count them as full multiplications here as well. This leads to an extra cost of $4\mathbf{M}$ compared to the computation in stage 1, and thus a cost of $7\mathbf{M} + 7\mathbf{S}$ per combined doubling and addition. A complete ladder step, which includes one more differential addition, costs $10\mathbf{M} + 9\mathbf{S}$ in this case. In comparison, a ladder step in [6] costs only $9\mathbf{M} + 6\mathbf{S}$.

As in stage 1, another option would be to compute $P + [m]Q$ in full Edwards projective and extended coordinates. The problem here is that the Y-coordinates of P and Q are not given.

8 Implementation results

We implemented SIDH with an optimal strategy based on [6] in Python 2.7. This was used as a reference for performance comparisons to the described Edwards arithmetic. The Python scripts can be found on `https://gitlab.cs.hs-rm.de/pqcrypto/SIDH`. In the Edwards script, we implemented the stages 1 and 2 from

above, namely in stage 1 a basefield ladder, that uses a combination of an Edwards doubling and a Montgomery differential addition, and in stage 2 Edwards doublings and triplings as described. In total, the computational effort increased by roughly 10% with all these changes. However this gap will probably a little smaller, if we adjust the optimal strategy for this case.
Work on a detailed comparison of the different operations in a C implementation is still in progress.

9 Conclusion and future work

As a result we can suppose, that a hybrid SIDH scheme, that uses Edwards arithmetic whenever possible, does not yield a speedup for SIDH. However, since the computations are almost as efficient as in the state-of-the-art implementation [6], it is still possible, that a full Edwards version of SIDH with efficient Edwards isogeny formulas can improve the performance of SIDH. In this case, if YZ-only arithmetic is used, it may be even advantageous to switch to Montgomery curves in some cases, e.g. to speed up doublings. However, we leave this question open for further investigation.

References

[1] Daniel J. Bernstein, Peter Birkner, Mark Joye, Tanja Lange, and Christiane Peters. Twisted edwards curves. In Serge Vaudenay, editor, *Progress in Cryptology - AFRICACRYPT 2008*, pages 389–405. Lecture Notes in Computer Science, 5023, Springer, 2008.

[2] Daniel J. Bernstein, Peter Birkner, Tanja Lange, and Christiane Peters. ECM Using Edwards Curves. *Mathematics of Computation*, 82(282):1139–1179, 2013.

[3] Daniel J. Bernstein and Tanja Lange. Analysis and optimization of elliptic-curve single-scalar multiplication. In Gary L. Mullen, Daniel Panario, and Igor E. Shparlinski, editors, *Finite Fields and Applications: Papers from the 8th International Conference*, pages 1–19. Contemporary Mathematics, 461, American Mathematical Society, 2008.

[4] Wouter Castryck, Steven Galbraith, and Reza Rezaeian Farashahi. Efficient arithmetic on elliptic curves using a mixed edwards-montgomery representation. Cryptology ePrint Archive, Report 2008/218, 2008. `http://eprint.iacr.org/2008/218`.

[5] Craig Costello and Huseyin Hisil. A simple and compact algorithm for sidh with arbitrary degree isogenies. Cryptology ePrint Archive, Report 2017/504, 2017. `https://eprint.iacr.org/2017/504`.

[6] Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny diffie-hellman. In M. Robshaw and J. Katz, editors, *Advances in Cryptology - CRYPTO 2016*, pages 572–601. Lecture Notes in Computer Science, 9814, Springer, 2016.

[7] Huseyin Hisil, Kenneth Koon-Ho Wong, Gary Carter, and Ed Dawson. Twisted edwards curves revisited. In Josef Pieprzyk, editor, *Advances in Cryptology - ASIACRYPT 2008*, pages 326–343. Lecture Notes in Computer Science, 5350, Springer, 2008.

[8] David Jao, Luca De Feo, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[9] Peter L. Montgomery. Speeding the Pollard and Elliptic Curve Methods of Factorization. *Mathematics of Computation*, 48(177):243–264, 1987.

[10] The National Institute of Standards and Technology (NIST). Submission requirements and evaluation criteria for the post-quantum cryptography standardization process, 2016.

[11] A. Rostovtsev and A. Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. `http://eprint.iacr.org/2006/145`.

[12] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In M. Robshaw and J. Katz, editors, *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 124–134. 1994.

[13] J. Vélu. Isogénies entre courbes elliptiques. *C.R. Acad. Sc. Paris, Série A.*, 271:238–241, 1971.