JSIAM Letters Vol.9 (2017) pp.29–32 ©2017 Japan Society for Industrial and Applied Mathematics JSIAM Letters

Constructing an efficient hash function from 3-isogenies

Hikari Tachibana[1], Katsuyuki Takashima[2] and Tsuyoshi Takagi[3]

[1] Graduate School of Mathematics, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka, Japan
[2] Information Technology R&D Center, Mitsubishi Electric Corporation, 5-1-1 Ofuna, Kamakura, Kanagawa, Japan
[3] Institute of Mathematics for Industry, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka, Japan

E-mail *ma215025@math.kyushu-u.ac.jp, Takashima.Katsuyuki@aj.MitsubishiElectric.co.jp*

Received October 25, 2016, Accepted January 23, 2017

Abstract

Charles et al. proposed hash functions based on the difficulty of computing isogenies between supersingular elliptic curves. Yoshida and Takashima then improved the 2-isogeny hash function computation by using some specific properties of 2-torsion points. In this paper, we extend the technique to 3-isogenies and give the efficient 3-isogeny hash computation based on a simple representation of the (backtracking) 3-torsion point. Moreover, we implement the 2- and 3-isogeny hash functions using Magma and show our 3-isogeny proposal has a comparable efficiency with the 2-isogeny one.

Keywords elliptic curve, isogeny, post-quantum cryptography

Research Activity Group Algorithmic Number Theory and Its Applications

1. Introduction

A hash function maps an arbitrary length bit string to a fixed length bit string and the hash value should be efficiently computed. A cryptographic hash function have to be computationally difficult to find two distinct inputs that have the same outputs and to find an input that has given output. These properties are called collision resistance and preimage resistance, respectively.

Charles et al. constructed cryptographic hash functions from Pizer's Ramanujan graphs whose vertex set is $\bar{\mathbb{F}}_p$-isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{p^2}$ and edge set is ℓ-isogenies between two supersingular elliptic curves [1]. In particular, they showed an explicit algorithm for the $\ell = 2$ case. Yoshida and Takashima proposed more efficient computations of 2-isogeny sequences by using the relations of the roots and the coefficients of a quadratic equation [2]. Their algorithms consist of one multiplication and one square root calculation over $\mathbb{F}_{p^2}$. Although the Charles et al. algorithm can be extended for the $\ell = 3$ case, an explicit description of the algorithm is not yet given and there exist several unclear points for explicit estimations of the computation time.

Therefore, in this paper, we give an explicit description this hash function with 3-isogenies by expressing an input as ternary expansion and assigning $\{0, 1, 2\}$ to three edges of each vertex that are not backtracking. When we compute a 3-isogeny, we use Vélu's formula like the $\ell = 2$ case and we show that a proposition in the $\ell = 2$ case can be extended to $\ell = 3$ for computation of backtracking points.

In the algorithm, we solve the cubic polynomial equation that is the factor of the 3rd division polynomial by Cardano's formula and give the efficient algorithm that computes the 3-isogeny by fifteen multiplications, one square root calculation and one cube root calculation. The roots of the 3rd division polynomial are equal to x-coordinates of 3-torsion points on the elliptic curve. Moreover, we implemented the 2- and 3-isogeny hash functions using Magma [3] when their security levels are 128-bit security (against classical computers) and show these functions can be computed in almost the same time.

2. Elliptic curves and Vélu's formulas

2.1 Elliptic curves

Let p be a prime greater than 3. Let $\mathbb{F}_p$ be a finite field with p elements and $\bar{\mathbb{F}}_p$ its algebraic closure. An elliptic curve E over $\bar{\mathbb{F}}_p$ is given by the Weierstrass normal form $y^2 = x^3 + Ax + B$, where A and $B \in \bar{\mathbb{F}}_p$ such that $4A^3 + 27B^2 \neq 0$.

The j-invariant of E is defined by $j(E) = j(A, B) = 1728 \times [4A^3/(4A^3 + 27B^2)]$. Conversely, an elliptic curve E that has j-invariant $j \in \bar{\mathbb{F}}_p$ ($j \neq 0, 1728$) can be obtained by setting its coefficients $A(j) := 3j/(1728 - j)$ and $B(j) := 2j/(1728 - j)$. Two elliptic curves have the same j-invariant if and only if they are isomorphic over $\bar{\mathbb{F}}_p$.

The set of $\bar{\mathbb{F}}_p$ rational points on E is $E(\bar{\mathbb{F}}_p) = \{(x, y) \in \bar{\mathbb{F}}_p^2 | y^2 = x^3 + Ax + B\} \cup \{O_E\}$, where O_E denotes the point at infinity on E. For each integer n, let $[n]$ be the multiplication-by-n map on E. Let $E[n] = \{P \in E(\bar{\mathbb{F}}_p) | [n]P = O_E\}$ be the set of n-torsion points. If $p \nmid n$, $E[n] \cong \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$.

For two elliptic curves E_1, E_2 over $\bar{\mathbb{F}}_p$, a homomorphism $\phi : E_1 \to E_2$ is a rational map that sends O_{E_1} to O_{E_2}. A non-zero homomorphism is called an isogeny, and a separable isogeny with the cardinality ℓ of the ker-

nel is called an ℓ-isogeny. For any ℓ-isogeny $\phi : E_1 \to E_2$, there exists an unique ℓ-isogeny $\hat{\phi} : E_2 \to E_1$ such that $\hat{\phi} \circ \phi = [\ell]$. It is called the dual isogeny of ϕ.

If an elliptic curve E over $\bar{\mathbb{F}}_p$ satisfies $E[p] = \{O_E\}$, then we say that E is supersingular. The j-invariants of supersingular elliptic curves are always in $\mathbb{F}_{p^2}$ [4, Th.V.3.1]. The number of j-invariants of supersingular elliptic curves over $\bar{\mathbb{F}}_p$ is $\lfloor p/12 \rfloor + \epsilon$, where $\epsilon \in \{0, 1, 2\}$ depending on the congruence class of p modulo 12 [4, Th.V.4.1(c)].

2.2 Vélu's formulas

We use Vélu's formulas [5] to compute 3-isogenies. When an elliptic curve E and a subgroup C of E are given, Vélu's formulas give the explicit formulas of the isogeny $\phi : E \to E'$ with $\ker \phi = C$ and the equation of E'. It is for any degree ℓ. In this paper, we use the formulas when $\ell = 3$. Let C be a subgroup of order 3 of E, then there exists a 3-isogeny $\phi : E \to E'$ with $\ker \phi = C$. Denote E' by E/C. In this paper, if $C = \langle (\alpha_x, \alpha_y) \rangle$, then we say the 3-isogeny ϕ is constructed by (α_x, α_y), where (α_x, α_y) is a 3-torsion point.

When $\ell = 3$, let (α_x, α_y) be a 3-torsion point on E and $C = \langle (\alpha_x, \alpha_y) \rangle$, then an elliptic curve E/C is given by the equation

$$Y^2 = X^3 - (9A + 30\alpha_x^2)X - (70\alpha_x^3 + 42A\alpha_x + 27B). \quad (1)$$

The isogeny $\phi : E \ni (x, y) \mapsto (X, Y) \in E/C$ is also given by the following:

$$X = x + \frac{2(3\alpha_x^2 + A)}{x - \alpha_x} + \frac{4\alpha_y^2}{(x - \alpha_x)^2}, \quad (2)$$

$$Y = y - \frac{8\alpha_y^2 y}{(x - \alpha_x)^3} - \frac{2(3\alpha_x^2 + A)y}{(x - \alpha_x)^2},$$

$\phi(O_E) = O_{E/C}$ and $\phi((\alpha_x, \alpha_y)) = O_{E/C}$.

If E is supersingular, E/C is also supersingular.

The x-coordinates of 3-torsion points on E are equal to the solutions of the 3rd division polynomial for E

$$\psi_3(x) = 3x^4 + 6Ax^2 + 12Bx - A^2. \quad (3)$$

3. Pizer's graphs and hash functions

3.1 Expander graphs and Pizer's graphs

Let $G = (\mathcal{V}, \mathcal{E})$ be a graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. A graph G is called an expander graph with expansion constant $c > 0$ if, for any subgroups $\mathcal{U} \subset \mathcal{V}$ such that $\sharp\mathcal{U} \geq \sharp\mathcal{V}/2$, the boundary $\Gamma(\mathcal{U})$ of $\mathcal{U}$ satisfies $\sharp\Gamma(\mathcal{U}) \geq c \cdot \sharp\mathcal{U}$ where $\Gamma(\mathcal{U}) = \{v \in \mathcal{V} | \exists u \in \mathcal{U}, \{u, v\} \in \mathcal{E}\} - \mathcal{U}$. Any expander graph is connected. A random walk on an expander graph has the rapidly mixing property. After $O(\log \sharp\mathcal{V})$ steps, the end point of the random walk approximates the uniform distribution on $\mathcal{V}$.

Let p and ℓ be two distinct primes. Pizer's graph $G(p, \ell)$ has $\bar{\mathbb{F}}_p$-isomorphism classes of supersingular elliptic curves over $\mathbb{F}_{p^2}$ as its vertex set $\mathcal{V}$. We represent each vertex by its j-invariant. Let the edge set $\mathcal{E}$ be ℓ-isogenies between two isomorphism classes. The Pizer graph is $(\ell + 1)$-regular graph and has the rapidly mixing property. In particular, this graph is one of the Ramanujan graphs, a special type of an expander graph. For details, see [1] or [6].

3.2 Hash functions using Pizer graphs

Charles et al. constructed hash functions using random walks on Pizer graphs. The integer input of the hash functions is used to determine the direction of the random walk, and the end point of the random walk is the output of the hash functions.

Let E_0 be a starting point and n the length of the walk. Then the walk is represented by the sequence of the elliptic curves $E_0 \to E_1 \to \cdots \to E_n$. A vertex E_i is connected to E_{i+1} by an ℓ-isogeny ϕ_i, which can be computed by Vélu's formula. Therefore the hash value of the hash function is given by computing the isogeny $\phi_i : E_i \to E_{i+1}$ repeatedly, where $i \in \{0, 1, \ldots, n-1\}$.

Each edge ϕ_i from E_i to E_{i+1} is chosen as follows. Let ω be the input of the hash function. The input ω is converted into base-ℓ number $b_0 b_1 \cdots b_{n-1} \in \{0, 1, \ldots, \ell - 1\}^n$. A vertex E_i has $\ell + 1$ edges, and one of them is connected to E_{i-1} and called a backtracking. We assign $\{0, 1, \ldots, \ell - 1\}$ to the other ℓ edges, then the edge ϕ_i is the one assigned b_i or the $(i + 1)$-th digit of the input. The length of the random walk is $\lceil \log_\ell \omega \rceil$. This means the ℓ-isogeny computation is repeated $\lceil \log_\ell \omega \rceil$ times.

3.3 Security of the Pizer graph hash functions

Hash functions from expander graphs have been constructed by Cayley graphs and Pizer graphs. The Zémor hash function and the LPS hash function are Cayley graph hash functions. The polynomial-time attacks on these Cayley graph hash functions have already known [7, 8]. However, no polynomial-time attacks on Pizer graph hash function have been found for now.

The security of the Pizer graph hash function is based on the difficulties of the following problems defined by Charles et al [1, Sec.5].

Problem 1 *Find a pair of supersingular elliptic curves E_1, E_2 over $\mathbb{F}_{p^2}$ and two distinct isogenies $f_1 : E_1 \to E_2$, $f_2 : E_1 \to E_2$ of degree ℓ^n.*

Problem 2 *Given a supersingular elliptic curve E over $\mathbb{F}_{p^2}$, find an endomorphism $f : E \to E$ ($f \neq [\ell^n]$) of degree ℓ^{2n}.*

Problem 3 *Given two supersingular elliptic curves E_1, E_2 over $\mathbb{F}_{p^2}$, find an isogeny $f : E_1 \to E_2$ of degree ℓ^n.*

Problem 3 is called the isogeny problem and the preimage resistance of this hash function is based on hardness of this problem. The isogeny problems are classified according to whether the elliptic curves are ordinary or supersingular.

The complexity of the best known attacks on this problem are following. In the case of ordinary elliptic curves, Galbraith and Stolbunov gave a classical exponential $\tilde{O}(\sqrt[4]{p})$ algorithm in 2013 [9] and Childs, Jao and Soukharev proposed a quantum subexponential $L_p[1/2, \sqrt{3}/2]$ algorithm in the same year [10], where $L_p[a, c] = \exp((c + o(1))(\log p)^a (\log \log p)^{1-a})$. In the case of supersingular elliptic curves, Delfs and Galbraith proposed a classical exponential $\tilde{O}(\sqrt{p})$ algorithm in 2013 [11] and Biasse, Jao and Sankar gave quantum exponential $\tilde{O}(\sqrt[4]{p})$ algorithm in 2014 [12] based on [11].

4. Proposed method: A hash function using 3-isogenies

We propose an efficient hash function using 3-isogenies by extending [2, Prop.1] to the case of 3-torsion points. 3-isogeny hash function repeats 3-isogeny computation for $\lceil m/\log_2 3 \rceil$ times, where m is the bit length of the input of the hash function.

4.1 Notation and selector functions

For each integer $i \geq 0$, we fix the notation about 3-torsion points on E_i. We denote a point that constructs an isogeny $\phi_i : E_i \to E_{i+1}$ by (α_i^x, α_i^y), i.e.

$$\phi_i : E_i \to E_{i+1} = E_i/\langle(\alpha_i^x, \alpha_i^y)\rangle.$$

We call a torsion point that constructs a dual isogeny of ϕ_{i-1} a backtracking point and denote it by (β_i^x, β_i^y).

We define selector functions to determine the isogeny. Each vertex in 3-isogeny graph has four edges, so we assign $\{0, 1, 2\}$ to three edges from the vertex that is not the backtracking. We fix a generator τ such that $\mathbb{F}_{p^2} = \mathbb{F}_p[\tau] = \mathbb{F}_p\tau + \mathbb{F}_p \cong (\mathbb{F}_p)^2$ to use a natural lexicographic order in $(\mathbb{F}_p)^2$ and define the following selector function for $\lambda_0, \lambda_1, \lambda_2 \in \mathbb{F}_{p^2}$ and $b \in \{0, 1, 2\}$:

$$\texttt{select}(\lambda_0, \lambda_1, \lambda_2, b) = \begin{cases} \texttt{min}(\lambda_0, \lambda_1, \lambda_2) & \text{if } b = 0 \\ \texttt{mid}(\lambda_0, \lambda_1, \lambda_2) & \text{if } b = 1 \\ \texttt{max}(\lambda_0, \lambda_1, \lambda_2) & \text{if } b = 2 \end{cases}$$

At the starting vertex E_0, we can choose any edges from E_0 as the next edge. So we will use the following selector function $\texttt{select}_0$ to determine α_0^x. Let λ_0, λ_1, λ_2, $\lambda_3 \in \mathbb{F}_{p^2}$ be four roots of the 3rd division polynomial $\psi_3(x) = 3x^4 + 6A_0x^2 + 12B_0x - A_0^2$ for E_0 in ascending order. Given A_0, $B_0 \in \mathbb{F}_{p^2}$ and $b \in \{0, 1, 2\}$, $\texttt{select}_0(A_0, B_0, b)$ returns λ_b as α_0^x.

4.2 3-torsion points' properties

For generators P and Q of 3-torsion points on E, i.e., $E[3] = \langle P, Q\rangle$, four edges around E in the Pizer graph are represented by four cyclic groups $\langle P\rangle, \langle Q\rangle, \langle P+Q\rangle$ and $\langle P-Q\rangle$, which are given as kernels of isogenies.

Lemma 4 *Let P and Q be generators of 3-torsion points on E and $\phi : E \to E' = E/\langle P\rangle$ a 3-isogeny. Then, $\phi(Q)$, $\phi(P+Q)$ and $\phi(P-Q)$ construct dual isogenies of ϕ, i.e. define the backtracking of ϕ.*

Proof Since $\phi(P) = O_{E'}$ and $\phi(P+Q) = \phi(P) + \phi(Q) = \phi(Q)$ and $\phi(P-Q) = -\phi(Q)$, it suffices to show $\phi(Q)(\neq O_{E'})$ is the backtracking point of ϕ. Then, let ϕ' be a 3-isogeny from E' to $E'' := E'/\langle\phi(Q)\rangle$. Hence, since we have

$$\begin{aligned} E'' &= E'/\langle\phi(Q)\rangle = (E/\langle P\rangle)/\langle\phi(Q)\rangle \\ &= E/\langle P, Q\rangle = E/E[3] \simeq E, \end{aligned}$$

the map $\phi' \circ \phi : E \to E'' \simeq E$ is equal to the multiplication-by-3 map on E up to isomorphisms. Therefore the isogeny ϕ' is the dual isogeny of ϕ, i.e. the point $\phi(Q)$ defines the backtracking of ϕ.

(QED)

Proposition 5 *The x-coordinate β_{i+1}^x of the backtracking point on E_{i+1} is given by*

$$\beta_{i+1}^x = -3\alpha_i^x.$$

Proof From Lemma 4, $(\beta_{i+1}^x, \beta_{i+1}^y) = \phi_i(\beta_i^x, \beta_i^y)$ where $\phi_i : E_i \to E_{i+1}$ is a 3-isogeny constructed by (α_i^x, α_i^y). For simplicity, let $\alpha_i := \alpha_i^x$, $\beta_i := \beta_i^x$ and $\beta_{i+1} := \beta_{i+1}^x$. From Vélu's formula (2),

$$\begin{aligned} &\beta_{i+1} \\ &= \beta_i + \frac{2(3\alpha_i^2 + A_i)}{\beta_i - \alpha_i} + \frac{4(\alpha_i^3 + A_i\alpha_i + B_i)}{(\beta_i - \alpha_i)^2} \\ &= \frac{-2\alpha_i^3 + 7\beta_i\alpha_i^2 - 2\beta_i^2\alpha_i + \beta_i^3 + (2A_i\alpha_i + 2A_i\beta_i + 4B_i)}{(\beta_i - \alpha_i)^2}. \end{aligned} \tag{4}$$

Since α_i and β_i are the x-coordinates of 3-torsion points on E_i, the 3rd division polynomial $\psi_3(x)$ given by (3) for E_i vanishes at α_i and β_i. Then, $(x - \alpha_i)(x - \beta_i)$ can divide $\psi_3(x)$, and by substituting α_i for x into degree three polynomial $\psi_3(x)/(x - \beta_i)$, we have a relation

$$\alpha_i^3 + \beta_i\alpha_i^2 + (2A_i + \beta_i^2)\alpha_i + (\beta_i^3 + 2A_i\beta_i + 4B_i) = 0,$$

which is equivalent to

$$2A_i\alpha_i + 2A_i\beta_i + 4B_i = -\alpha_i^3 - \beta_i\alpha_i^2 - \beta_i^2\alpha_i - \beta_i^3.$$

Therefore, by using this expression we can eliminate both A_i and B_i from (4), i.e.,

$$\beta_{i+1} = \frac{-3\alpha_i^3 + 6\beta_i\alpha_i^2 - 3\beta_i^2\alpha_i}{(\beta_i - \alpha_i)^2} = -3\alpha_i.$$

(QED)

4.3 Computation of 3-isogeny sequence

In this subsection, we explain an algorithm to compute a 3-isogeny sequence, which executes 3-isogeny computations repeatedly (Algorithm 1). Each 3-isogeny computation consists of the following four steps. By Proposition 5, we can omit extra operations in steps 2 and 3 compared to the straightforward computation (Algorithm 2). Let $\beta_{i+1} := \beta_{i+1}^x$ and $\alpha_i := \alpha_i^x$ for simplicity below.

1. Compute the curve E_{i+1}, i.e. (A_{i+1}, B_{i+1}) by Vélu's formula (1). Note that we keep intermediate values $\xi_1 := \alpha_i^2, \xi_2 := \alpha_i^3$ and $\xi_3 := A_i\alpha_i$ for step 3.
2. Compute the x-coordinate β_{i+1} of the backtracking point on E_{i+1}.
3. Solve the cubic equation $f(x) = 0$ that is the factor of the 3rd division polynomial $\psi_3(x) = (x - \beta_{i+1})f(x) = (x + 3\alpha_i)f(x)$ for E_{i+1} by Cardano's formula.
4. Choose α_{i+1}^x from the above solutions using the function `select` and return A_{i+1}, B_{i+1} and α_{i+1}^x.

In step 2, from Proposition 5, we have $\beta_{i+1} = -3\alpha_i$. This gains three multiplications and one inversion efficiency from Vélu's formula (2).

In step 3, the factor is $f(x) = x^3 - 3\alpha_i x^2 + (2A_{i+1} + 9\alpha_i^2)x + 4B_{i+1} - 6A_{i+1}\alpha_i - 27\alpha_i^3$. Let ω_0 be a solution of $x^2 + x + 1 = 0$. By Cardano's formula, we have three

Algorithm 1 3-isogeny sequence computation

Input: $j_0 = j(E_0)$, walkdata $\omega = b_0 b_1 \dots b_{n-1} \in \{0,1,2\}^n$
Output: $j_n = j(E_n)$

1: $(A_0, B_0) \leftarrow (A(j_0), B(j_0))$, $\alpha_0^x \leftarrow \mathtt{select}_0(A_0, B_0, b_0)$
2: **for** $i = 0$ to $n-2$ **do**
3: $\quad (A_{i+1}, B_{i+1}, \alpha_{i+1}^x) \leftarrow \mathtt{Isog}_3(A_i, B_i, \alpha_i^x, b_{i+1})$
4: **end for**
5: $\xi_1 \leftarrow (\alpha_{n-1}^x)^2$, $\xi_2 \leftarrow \alpha_{n-1}^x \xi_1$, $\xi_3 \leftarrow A_{n-1}\alpha_{n-1}^x$,
$A_n \leftarrow -(9A_{n-1} + 30\xi_1)$,
$B_n \leftarrow -(70\xi_2 + 42\xi_3 + 27B_{n-1})$, $j_n \leftarrow j(A_n, B_n)$
6: **return** j_n

Algorithm 2 $\mathtt{Isog}_3$: 3-isogeny computation

Input: A_i, B_i, α_i^x, $b_{i+1} \in \{0,1,2\}$
Output: A_{i+1}, B_{i+1}, α_{i+1}^x

1: $\xi_1 \leftarrow (\alpha_i^x)^2$, $\xi_2 \leftarrow \alpha_i^x \xi_1$, $\xi_3 \leftarrow A_i \alpha_i^x$,
$A_{i+1} \leftarrow -(9A_i + 30\xi_1)$,
$B_{i+1} \leftarrow -(70\xi_2 + 42\xi_3 + 27B_i)$ /* Vélu's formula */
2: /* Solve the cubic equation */
$t \leftarrow -6(3\xi_1 + A_i)$, $s \leftarrow -6(15\xi_2 + 11\xi_3 + 9B_i)$,
$\zeta \leftarrow \sqrt{s^2 + t^3}$, $u \leftarrow \sqrt[3]{-s + \zeta}$, $v \leftarrow -t/u$,
$\lambda_0 \leftarrow \alpha_i^x + u + v$, $\lambda_1 \leftarrow \alpha_i^x + \omega_0 u + \omega_0^2 v$,
$\lambda_2 \leftarrow \alpha_i^x + \omega_0^2 u + \omega_0 v$ /* ω_0, ω_0^2 are precomputed. */
3: $\alpha_{i+1}^x \leftarrow \mathtt{select}(\lambda_0, \lambda_1, \lambda_2, b_{i+1})$
4: **return** A_{i+1}, B_{i+1}, α_{i+1}^x

solutions λ_k ($k \in \{0,1,2\}$) of $f(x) = 0$ as

$$\lambda_k = \alpha_i + \omega_0^k \sqrt[3]{-s + \sqrt{s^2 + t^3}} + \omega_0^{3-k} \sqrt[3]{-s - \sqrt{s^2 + t^3}},$$

where $t = (2A_{i+1} + 6\alpha_i^2)/3$ and $s = 2B_{i+1} - 2A_{i+1}\alpha_i - 10\alpha_i^3$. Applying Vélu's formula (1) gives $t = -6(3\alpha_i^2 + A_i) = -6(3\xi_1 + A_i)$ and $s = -6(15\alpha_i^3 + 11A_i\alpha_i + 9B_i) = -6(15\xi_2 + 11\xi_3 + 9B_i)$. Here, ξ_1, ξ_2, ξ_3 are already computed in step 1, and then this omits three extra multiplications. Note that since $\sqrt[3]{-s + \sqrt{s^2 + t^3}} \cdot \sqrt[3]{-s - \sqrt{s^2 + t^3}} = -t$, the cube root computation is needed only once. Algorithm 2 is a 3-isogeny version of [2, Algorithm 4], which is used for 2-isogeny sequences.

4.4 Implementation results

We implemented Yoshida-Takashima's algorithm (Algorithm YT) [2, Alg.5] and Algorithm 1 on Magma. For three primes $p = 2^{255} + r$ ($r \in \{141, 95, 821\}$) and 256-bit random inputs of these hash functions, we measured the average timing of their calculations, respectively. Our implementation was done on an Intel Core i7 processor with 8.00 GB RAM running at 2.30 GHz. We used Magma V2.19-9. All the results are shown in Table 1.

In our Magma implementation, we examine the proportion of the timing of square and cube root calculations in the whole computation of 2- and 3-isogeny sequences, respectively. We used the `SquareRoot` and `Root` Magma commands to calculate a square root and a cube root, respectively. We compare the costs of computing 2- and 3-isogeny sequences for the same input length, that is, the numbers of iterations of 2- and 3-isogeny computations in Algorithm YT and Algorithm 1, $m \approx \log_2 \omega$ and $n \approx \log_3 \omega$ respectively, where input ω is given as an integer. We set $m := 256$, $n := 161$ in Table 1 for 256-bit input ω. We see that the 3-isogeny hash function can be computed faster than or as fast as the 2-isogeny hash function when the input lengths of the hash functions are appropriately set. For example, for the prime $p = 2^{255} + 141$, the cost of 3-isogeny sequence computation is 0.84 times of that of 2-isogeny.

Table 1. The running time in second of Algorithm YT and Algorithm 1 with the iteration number $m := 256$ (resp., $n := 161$) of Algorithm YT (resp., Algorithm 1) and the proportion of square and cube root computations in the total computations [%].

prime p	Algorithm YT	Algorithm 1
$2^{255} + 141$	0.872 s (98.4%)	0.732 s (97.3%)
$2^{255} + 95$	1.34 s (99.0%)	1.16 s (98.0%)
$2^{255} + 821$	1.16 s (98.9%)	1.46 s (98.4%)

5. Conclusion

In this paper, we proposed an efficient hash function using 3-isogenies. The proposed 3-isogeny hash can be computed a little faster than the 2-isogeny one or has a comparable efficiency with it. Nowadays post-quantum cryptosystems are actively studied, and then we expect that the result in this paper motivates further investigations for a thorough comparison of the two hash functions.

Acknowledgments

This work was supported by CREST, JST.

References

[1] D. X. Charles, E. Z. Goren and K. E. Lauter, Cryptographic hash functions from expander graphs, J. Cryptology, **22** (2009), 93–111.
[2] R. Yoshida and K. Takashima, Computing a sequence of 2-isogenies on supersingular elliptic curves, IEICE Trans. Fundamentals, **E96-A** (2013), 158–165.
[3] W. Bosma, J. J. Cannon, C. Fieker and A. Steel, Handbook of Magma functions, Edition 2.19, 2013.
[4] J. H. Silverman, The Arithmetic of Elliptic Curves, 2nd ed., Springer-Verlag, New York, 2009.
[5] J. Vélu, Isogenies entre courbes elliptiques, C. R. Acad. Sci. Paris, Series A, **273** (1971), 238–241.
[6] A. K. Pizer, Ramanujan graphs and Hecke operators, B. Am. Math. Soc., **23** (1990), 127–137.
[7] J. P. Tillich and G. Zémor, Group-theoretic hash functions, in: Proc. of Algebraic Coding 1993, G. Cohen et al. eds., LNCS, Vol. 781, pp.90–110, Springer-Verlag, Berlin, 1994.
[8] J. P. Tillich and G. Zémor, Collisions for the LPS expander graph hash function, in: Proc. of EUROCRYPT 2008, N. P. Smart eds., LNCS, Vol. 4965, pp.254–269, Springer-Verlag, Berlin, 2008.
[9] S. D. Galbraith and A. Stolbunov, Improved algorithm for the isogeny problem for ordinary elliptic curves, Appl. Algebra Eng. Comm., **24** (2013), 107–131.
[10] A. Childs, D. Jao and V. Soukharev, Constructing elliptic curve isogenies in quantum subexponential time, J. Math. Crypt., **8** (2013), 1–29.
[11] C. Delfs and S. D. Galbraith, Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$, Design. Code. Cryptogr., **78** (2016), 425–440.
[12] J. F. Biasse, D. Jao and A. Sankar, A quantum algorithm for computing isogenies between supersingular elliptic curves, in: Proc. of INDOCRYPT 2014, W. Meier et al. eds., LNCS, Vol. 8885, pp.428–442, Springer-Verlag, Berlin, 2014.