Publ. Math. Debrecen
Supplementum **100** (2022), 639–653
DOI: 10.5486/PMD.2022.Suppl.6

Application of Velusqrt algorithm to Huff's curves

By Michał Wroński

Abstract. In 2020 Bernstein, De Feo, Leroux, and Smith presented a new odd-degree ℓ-isogeny computation method called Velusqrt. This method has complexity $\tilde{O}(\sqrt{\ell})$, compared to the complexity of $\tilde{O}(\ell)$ of the classical Vélu method. In this paper, the application of the Velusqrt method to Huff's curves is presented. It is shown how to compute odd-degree isogeny on Huff's curves using the Velusqrt algorithm and x-line arithmetic for different compression functions, especially for degree 4 compression function $f_{4,x^2} = x^2$.

1. Introduction

In [1] Bernstein, De Feo, Leroux, and Smith presented an odd-degree isogeny computation method called Velusqrt. They modified the algorithm for the evaluation of polynomials whose roots are powers $h_S(\alpha) = \prod_{s\in S}(\alpha - \zeta^s)$, with complexity $\tilde{O}(\sqrt{\#S})$, to use a similar technique with x-line arithmetic for points on an elliptic curve to evaluate $h_S(\alpha) = \prod_{s\in S}(\alpha - f([s]P))$, where $f : E \to \mathbb{F}_q$ is compression function. Such an algorithm has complexity $\tilde{O}(\sqrt{\ell})$, where ℓ is the degree of the isogeny.

Compression on elliptic curves (often called x-line arithmetic) is mainly used to reduce key sizes and protect solutions against side-channel attacks. If E is an elliptic curve over a field K and $f : E \to K$ is a rational function, for which $f(P) = f(-P)$ for all $P \in E$, then f is a compression function and for any $k \in \mathbb{Z}$ holds that $[k]f(P) = f([k]P)$. For example, on Weierstrass and Montgomery

Mathematics Subject Classification: 94A60, 14K02, 14H52.
Key words and phrases: general Huff's curves and Huff's curves and compression on elliptic curves and isogeny-based cryptography and Velusqrt method.

curves $f(x, y) = x$ is a compression function. Moreover, for compression function $f : E \to K$ there exist rational functions for doubling $D(x) \in K(x)$ and differential additions $\frac{F_1}{F_0}, \frac{F_2}{F_0} \in K(x, y)$ such that

$$\begin{aligned} f([2]P) &= D(f(P)), \\ f(P+Q) + f(Q-P) &= \frac{F_1(f(P), f(Q))}{F_0(f(P), f(Q))}, \\ f(P+Q)f(Q-P) &= \frac{F_2(f(P), f(Q))}{F_0(f(P), f(Q))} \end{aligned}$$

for any points $P, Q \in E$. After functions D and F_0 and F_1 or F_1 and F_2 are found, one can compute $[k]f(P)$ using values of f and the Montgomery ladder algorithm. There also exists a rational map $B : E \times K \times K \to E$ such that

$$Q = B(P, f(Q), f(P+Q)) \tag{1}$$

for generic points $P, Q \in E$, which we call the point recovery formula. Such formula allows for $P \in E$ to compute $[k]f(P)$ using the Montgomery ladder algorithm, which also gives $[k+1]f(P)$, and to recover point $[k]P$ on E given $P, [k]f(P), [k+1]f(P)$ substituting $Q = [k]P$ to the formula (1).

Many compression functions of different degrees have been considered in the case of alternative models of elliptic curves. The paper [4] has presented degree 2 compression functions on Huff's and general Huff's curves. Moreover, in [6] there have been presented in the case of Hessian curves compression functions of degree 6 and 18, in the case of Huff's curves, compression functions of degree 4, 8, 16, and in the case of Edwards curves, compression functions of degree 4 and 8. Comparing to the results obtained in the papers [4] and [6], in this paper, new degree 4 compression functions $f_{4,x^2} = x^2$ and $f_{4,y^2} = y^2$ have been presented, which are suitable for applications of the Velusqrt algorithm to Huff's curves. Computation of isogeny from kernel polynomials in the case of Huff's curves has been obtained using formulas obtained by Moody and Shumow [11] for general Huff's curves and formulas obtained by Dryło et al. [7] for Huff's curves.

It is tough to find suitable formulas using only elementary methods in many cases. That is why the Gröbner basis mechanism is often used, where searching for convenient functions can be automatized. Description of such method is presented in [3] for the compression function of degree 2 and in [6] for compression functions of high-degree, where program from [3] was modified.

In this paper, when necessary, the method described in [6] for searching for suitable functions will be used. The correctness of the formulas presented in the paper can be checked using the program *Huff_Correctness_x_square* from [5], which are analogous to the programs used for checking the correctness of the formulas presented in [7].

The core algorithm problem of the Velusqrt method is contained in a more general framework, which is an efficient evaluation of polynomial and rational functions over $\mathbb{F}_q$ whose roots are values of a function from a cyclic group to $\mathbb{F}_q$. In such case, one has to fix a cyclic group G, a generator P of G and a function $f : G \rightarrow \mathbb{F}_q$. For each finite subset S of $\mathbb{Z}$, one then defines polynomial $h_S(X) = \prod_{s\in S}(X - f([s]P))$, where $[s]P$ is the sum of s copies of P (group G is written additively).

So, given f and S, one wants then to evaluate $h_S(X)$ at point α, for any $\alpha \in \mathbb{F}_q$. The standard way of computation of $h_S(\alpha)$ requires $O(\#S)$ operations in $\mathbb{F}_q$. Even though, if S has enough additive structure and f is sufficiently compatible with the group structure on G, then one can compute $h_S(\alpha)$ in $\tilde{O}(\sqrt{\#S})$ operations in $\mathbb{F}_q$. For example, this idea is applied in Pollard's and Strassen's factorization algorithms.

We now define an index system.

Definition 1. [1, Definition 4.6] Let I and J be finite sets of integers.

(1) We say that (I, J) is an index system if the maps $I \times J \rightarrow Z$ defined by $(i, j) \rightarrow i + j$ and $(i, j) \rightarrow i - j$ are both injective and have disjoint images.

(2) If S is a finite subset of $\mathbb{Z}$, then we say that an index system (I, J) is an index system for S if $I + J$ and $I - J$ are both contained in S.

If (I, J) is an index system, then the sets $I + J$ and $I - J$ are in bijection with $I \times J$. We write $I \pm J$ for the union of $I + J$ and $I - J$.

The main result of [1] is an adaptation of Pollard's idea to evaluate $h_{I\pm J}(\alpha)$, where $h_{I\pm J}(\alpha)$ is the kernel polynomial. The biggest problem which had to be solved is that $f([i + j]P)$ cannot be represented only by $f([i]P)$ and $f([j]P)$.

Even though it is possible to do the following trick. If $f(P)$ is a compression function (whose degree is coprime with the degree of the isogeny) on elliptic curve E, then exist rational functions F_0, F_1 and F_2 such that $A_1 = \frac{F_1(f(P),f(Q))}{F_0(f(P),f(Q))} = f(P + Q) + f(P - Q)$ and $A_2 = \frac{F_2(f(P),f(Q))}{F_0(f(P),f(Q))} = f(P + Q)f(P - Q)$. Then $(X - f(P + Q))(X - f(P - Q)) = X^2 - \frac{F_1(f(P),f(Q))}{F_0(f(P),f(Q))}X + \frac{F_2(f(P),f(Q))}{F_0(f(P),f(Q))}$. This property then leads to the following equations

$$\begin{aligned} h_{I\pm J}(X) &= \prod_{(i,j)\in I\times J} (X - f([i+j]P))(X - f([i-j]P)) \\ &= \prod_{i\in I}\prod_{j\in J}(X^2 - A_1(f([i]P), f([j]P))X + A_2(f([i]P), f([j]P))). \end{aligned}$$

It means that most of S cannot be decomposed as $I+J$, but such decomposition involves both $I+J$ and $I-J$. Using these observations makes it possible to construct Algorithm 1.

```
Algorithm 1: Computing $h_S(\alpha) = \prod_{s\in S}(\alpha - f([s]P))$, based on [1, Algorithm 2]
  Data: a prime power $q$, an elliptic curve $E/\mathbb{F}_q, P \in E(\mathbb{F}_q)$, a finite
        subset $S \subset \mathbb{Z}$, an index system $(I, J)$ for $S$ such that
        $S \cap n\mathbb{Z} = I \cap n\mathbb{Z} = J \cap n\mathbb{Z} = \{\}$, where $n$ is the order of $P$
  Input: $\alpha \in \mathbb{F}_q$
  Output: $h_S(\alpha)$, where $h_S(X) = \prod_{s\in S}(X - f([s]P))$
  $h_I = \prod_{i\in I}(Z - f([i]P)) \in \mathbb{F}_q[Z]$
  $D_J = \prod_{j\in J} F_0(Z, f([j]P)) \in \mathbb{F}_q[Z]$
  $\Delta_{I,J} = Res_Z(h_I, D_J) \in \mathbb{F}_q$
  $E_J = \prod_{j\in J}\left(F_0(Z, f([j]P))\alpha^2 - F_1(Z, f([j]P))\alpha + F_2(Z, f([j]P))\right) \in \mathbb{F}_q[Z]$
  $R = Res_Z(h_i, E_j) \in \mathbb{F}_q$
  $h_K = \prod_{k\in S\setminus(I\pm J)}(\alpha - f([k]P)) \in \mathbb{F}_q$
  return $\frac{h_K \cdot R}{\Delta_{I,J}}$
```

Example 1. We use the following Example [1, Example 4.12]. Let us suppose that we want for Weierstrass curve, with compression $f(P) = x$ to evaluate $h_S(X) = \prod_{s\in S}(X - x([s]P))$, where $S = \{1, 3, \ldots, \ell - 2\}$. Let us note that set S can be replaced by any set of representatives of $((\mathbb{Z}/\ell\mathbb{Z}) \setminus \{0\})/\langle\pm1\rangle$.

Let $I = \{2b(2i+1) | 0 \leq i \leq b'\}$ and $J = \{1, 3, \ldots, 2b-1\}$ with $b = \left\lfloor \frac{\sqrt{\ell-1}}{2} \right\rfloor$ and (for $b > 0$) $b' = \left\lfloor \frac{\ell-1}{4b} \right\rfloor$. Then (I, J) is an index system for S. What is more $S \setminus (I \pm J) = K$, were $K = \{4bb' + 1, \ldots, \ell - 4, \ell - 2\}$. Algorithm 1 computes $h_S(\alpha)$ for any $\alpha \in \mathbb{F}_q$ in $\tilde{O}\left(\sqrt{\ell}\right)$.

As was shown in [1], the Velusqrt algorithm can be applied to the practical implementations of CSIDH and CSURF, obtaining faster solutions for $\ell \gtrapprox 110$ (it depends on many factors). The presented algorithm gives a 16% speedup for CSIDH-1024. In other presented situations, the speedup is less significant. Because the presented algorithm has much better asymptotic complexity than the method of Vélu for isogeny evaluation, for isogeny-based protocols with a higher level of security (e.g., CSIDH-2048, CSIDH-4096), the speedup should be much more significant.

What is more Chávez-Saab, Chi-Domínguez, Jaques and Rodríguez-Henríquez in [2] considered constant-time implementation of CSIDH using the Velusqrt method.

In the next sections, using these ideas, how to adapt the Velusqrt algorithm to Huff's model of elliptic curves will be shown.

2. Huff's curves and compression functions

In this section will be presented basic information on Huff's curves and applications of compression functions to Huff's curves arithmetic.

2.1. Huff's curves. Huff's curve over K is provided by the equation (see e.g. [8])

$$H_{a,b} \;:\; ax(y^2-1) = by(x^2-1),$$

where $a^2 \neq b^2$ and $a, b \neq 0$. The neutral element is the point $O = (0,0)$ and for any point $P = (x_P, y_P)$ the opposite point is equal to $-P = -(x_P, y_P) = (-x_P, -y_P)$. The addition law for two points $P = (x_P, y_P)$, $Q = (x_Q, y_Q)$ on $H_{a,b}$ is given by

$$\begin{cases} x_R = \dfrac{(x_P + x_Q)(1 + y_P y_Q)}{(1 + x_P x_Q)(1 - y_P y_Q)}, \\[2ex] y_R = \dfrac{(y_P + y_Q)(1 + x_P x_Q)}{(1 - x_P x_Q)(1 + y_P y_Q)}, \end{cases}$$

where $P + Q = (x_R, y_R)$.

Doubling and differential addition on Huff's curve using a degree 2 compression function $f_2(x,y) = xy$ are given by (see [7])

$$f_2([2]P) = \frac{4f_2(P)(f_2(P)^2 + \left(\frac{b}{a} + \frac{a}{b}\right) f_2(P) + 1)}{(f_2(P)^2 - 1)^2}, \tag{2}$$

$$f_2(P+Q)f_2(P-Q) = \left(\frac{f_2(P) - f_2(Q)}{f_2(P)f_2(Q) - 1}\right)^2. \tag{3}$$

One can also find in [7] the formula

$$\begin{aligned} &f_2(P+Q) + f_2(P-Q) \\ &\quad = \tfrac{2(f_2(P)f_2(Q)^2 + f_2(P)^2 f_2(Q) + 2\frac{b}{a} f_2(P)f_2(Q) + 2\frac{a}{b} f_2(P)f_2(Q) + f_2(Q) + f_2(P))}{(f_2(P)f_2(Q)-1)^2} \end{aligned}$$

using the method described in [3]. The formulas presented above can be checked using the program *GeneralHuff_Correctness_x_square* from [5].

Moreover, one can also find the formula for point recovery. For generic points $P = (x_P, y_P), Q = (x_Q, y_Q)$ on $H_{a,b}$ if we are given $P, f_2(Q), f_2(P+Q)$, then coordinates of Q are provided by

$$\begin{cases} x_Q = f_2(Q)\frac{(y_P f_2(P+Q)+x_P)(bf_2(Q)+a)+(af_2(Q)+b)(x_P f_2(P+Q)+y_P)}{(bf_2(Q)+a)(f_2(P+Q)-f_2(Q)+x_P y_P(f_2(Q)f_2(P+Q)-1))}, \\ y_Q = \frac{f_2(Q)}{x_Q}. \end{cases}$$

In projective coordinates, formulas (2) and (3) can be computed as efficiently as formulas [10] for Montgomery curves. In this way, doubling requires $2M+2S+c$, and differential addition has a cost equal to $4M+2S$. More details can be found in [7].

2.2. Huff's isogenies computation using compression functions. This subsection will present formulas for isogeny computation from [7], where it is also shown how to compute isogeny of an odd degree using a compression function $f_2(x, y) = xy$ of degree 2.

Theorem 1. [7, Theorem 4] *Let $F = \{(0,0), (\alpha_i, \beta_i), (-\alpha_i, -\beta_i) : i = 1 \ldots s\}$, where $-(\alpha_i, \beta_i) = (-\alpha_i, -\beta_i)$, be the kernel of an isogeny ψ. Let $A = \prod_{i=1}^{s} \alpha_i$ and $B = \prod_{i=1}^{s} \beta_i$. Let us define*

$$\psi(P) = \Big(x_P(-1)^s \textstyle\prod_{Q \neq (0,0) \in F} x_{P+Q}, y_P(-1)^s \prod_{Q \neq (0,0) \in F} y_{P+Q}\Big).$$

Then ψ is a ℓ-isogeny with kernel F, from the curve $H_{a,b}$, to the curve $H_{a',b'}$, where $a' = \frac{a}{A^2} = \frac{a}{\prod_{i=1}^{s} x_{Q_i}^2}$ and $b' = \frac{b}{B^2} = \frac{b}{\prod_{i=1}^{s} y_{Q_i}^2}$.

Corollary 1. [7, Corollary 2] *Let F be the kernel of the odd-degree isogeny ψ. For compression function of degree 2 given by $f_2(x, y) = xy$ let us note that $f_2(\psi(P))$ is provided by*

$$f_2(\psi(P)) = x_P(-1)^s \prod_{Q \neq (0,0) \in F} x_{P+Q} \cdot y_P(-1)^s \prod_{Q \neq (0,0) \in F} y_{P+Q},$$

which is equal to

$$f_2(\psi(P)) = x_P y_P \textstyle\prod_{Q \neq (0,0) \in F} x_{P+Q} y_{P+Q} = f_2(P) \prod_{Q \in F^+} f_2(P+Q) f_2(P-Q),$$

where F^+ is the set $\{(\alpha_i, \beta_i) : i = 1, \ldots, s\}$.

To find the coefficients a' and b' of Huff's curve $H_{a',b'}$, if $f_2(P) = x_P y_P = r_P$, one can use formulas from [7] for x^2 and y^2 as rational functions of r, where $r = xy$ is compression function of degree 2:

$$x^2 = \frac{r(ar+b)}{br+a}, \qquad y^2 = \frac{r(br+a)}{ar+b}.$$

Finally

$$a' = \frac{a}{\left(\prod_{i=1}^{s} x_{Q_i}\right)^2} = a\prod_{i=1}^{s} \frac{(br_{Q_i}+a)}{r_{Q_i}(ar_{Q_i}+b)},$$
$$b' = \frac{b}{\left(\prod_{i=1}^{s} y_{Q_i}\right)^2} = b\prod_{i=1}^{s} \frac{(ar_{Q_i}+b)}{r_{Q_i}(br_{Q_i}+a)}.$$

3. Application to Velusqrt

This section shows how to apply the Velusqrt algorithm for Huff's curves. From the computational point of view, the following corollary will be important.

Corollary 2. *If compression function f_d is of degree d and $GCD(d,\ell) = 1$, where ℓ is odd, one can evaluate $h_S(X) = \prod_{s\in S}(X - f_d([s]P))$ using set S and an index system from Example 1 for any $\alpha \in \mathbb{F}_q$ in $\tilde{O}\left(\sqrt{\ell}\right)$ operations.*

Let us note that in such case (the same as in the case of the compression function of degree 2), if F is a group of order ℓ, then for every $P_1, P_2 \in F$ holds that $f_d(P_1) = f_d(P_2)$ if and only if $P_1 = \pm P_2$, as same as for compression function of degree 2. It means that for every $i, j \in S$, holds that $f_d([i]P) = f_d([j]P)$ if and only if $i = \pm j$ so there will not be any additional redundancy.

3.1. Compression functions of degree 4. As will be shown later, to apply the Velusqrt technique to Huff's curves isogeny computation, it is convenient to use the formula for $f_{4,x^2}(P+Q) + f_{4,x^2}(P-Q), f_{4,x^2}(P+Q)f_{4,x^2}(P-Q)$ or $f_{4,y^2}(P+Q) + f_{4,y^2}(P-Q), f_{4,y^2}(P+Q)f_{4,y^2}(P-Q)$, where $f_{4,x^2}(P) = x_P^2$ is compression function of degree 4, similarly $f_{4,y^2}(P) = y_P^2$ is compression function of degree 4 also.

We will show that $f_{4,x^2}(P) = x_P^2$ is a compression function of degree 4.

In [9], Kohel was studied symmetric quartic models over binary fields with a rational 4-torsion point T. He showed that a genus one curve admits translations by rational points and translation morphism $\tau_T = P+T$ on curve E is projectively linear (induced by a linear transformation of the ambient projective space), if and only if E is a degree n model determined by a complete linear system in $\mathbb{P}^{n-1}$ and T is in the n-torsion subgroup. Such a method was used in [6] to obtain high-degree compression functions on many alternative models of elliptic curves.

This paper uses his ideas to find new compression functions of high degree (degree 4) for Huff's curves. The compression functions for which we are looking for are invariant on the action of involution and translation by specific point T, in this case, the point of order 2, which means that for the compression function of degree 4 holds that $f_4(P) = f_4(Q)$ if and only if $Q = \pm P + [k]T$, for $k = \overline{0,1}$.

Let us note that if $r = x^2$, then for each r we can find two distinct x's at most. Moreover, using Huff's curve equation, for each x one can find at most two distinct y's, which means that there are at most four distinct points $P_i, i = \overline{0,3}$, having the same value of compression $f_{4,x^2}(P_i) = r$. One can find that the compression function $f_{4,x^2}(P)$ is invariant under involution and translation by 2-torsion point $(0:1:0)$, because $(x,y)+(0:1:0) = (-x, \frac{1}{y})$. Then, $r = f_{4,x^2}(P)$ for $P \in \{(x,y), (-x,-y), (-x, \frac{1}{y}), (x, -\frac{1}{y})\}$.

We will firstly find formulas for $f_{4,x^2}(P+Q) + f_{4,x^2}(P-Q)$ and $f_{4,x^2}(P+Q)f_{4,x^2}(P-Q)$ on Huff's curve. Let $r_P = f_{4,x^2}(P)$ and $r_Q = f_{4,x^2}(Q)$, then

$$f_{4,x^2}(P+Q) + f_{4,x^2}(P-Q) = \frac{s_1(r_P, r_Q)}{s_0(r_P, r_Q)},$$

$$f_{4,x^2}(P+Q)f_{4,x^2}(P-Q) = \frac{s_2(r_P, r_Q)}{s_0(r_P, r_Q)},$$

where

$$\begin{aligned} s_0(r_P, r_Q) &= (r_P r_Q - 1)^2, \\ s_1(r_P, r_Q) &= -2\left({r_P}^2 r_Q + r_P {r_Q}^2 + \frac{8a^2-4b^2}{b^2} r_P r_Q + r_P + r_Q\right), \\ s_2(r_P, r_Q) &= (r_P - r_Q)^2. \end{aligned} \tag{4}$$

Formula for doubling $f_{4,x^2}([2]P)$ is equal to $\frac{N(r,a,b)}{D(r,a,b)}$, where

$$\begin{aligned} N(r,a,b) &= 4r\left(r^2 + \frac{4a^2-2b^2}{b^2} r + 1\right), \\ D(r,a,b) &= \left(r^2-1\right)^2. \end{aligned} \tag{5}$$

We can similarly find formulas for $f_{4,y^2}(P+Q)+f_{4,y^2}(P-Q)$ and $f_{4,y^2}(P+Q)f_{4,y^2}(P-Q)$

$$f_{4,y^2}(P+Q)+f_{4,y^2}(P-Q)=\frac{t_1(r_P,r_Q)}{t_0(r_P,r_Q)},$$

$$f_{4,y^2}(P+Q)f_{4,y^2}(P-Q)=\frac{t_2(r_P,r_Q)}{t_0(r_P,r_Q)},$$

where

$$\begin{aligned} t_0(r_P,r_Q) &= \left(r_P r_Q-1\right)^2, \\ t_1(r_P,r_Q) &= -2\left({r_P}^2 r_Q + r_P {r_Q}^2 + \frac{8b^2-4a^2}{a^2} r_P r_Q + r_P + r_Q\right), \\ t_2(r_P,r_Q) &= \left(r_P - r_Q\right)^2. \end{aligned} \tag{6}$$

Explanation 1. Formulas presented in (4) and (6) can be obtained using the method described in [6]. The correctness of the formulas presented above can be checked using the program *Huff_Correctness_x_square* from [5].

3.2. Velusqrt on Huff's curves. To obtain isogeny evaluation formulas for Huff's curves, we can use formulas for isogeny evaluation using kernel polynomials on general Huff's curves $G_{\overline{a},\overline{b}} : \overline{x}(\overline{ay}^2-1)=\overline{y}(\overline{b}\overline{x}^2-1)$ [12], presented in [11]. The formula for odd ℓ-isogeny is in this case given by

$$\overline{\psi}\left(\overline{P}\right)=\left(\frac{\overline{xg}(\overline{x})}{\overline{g}(0)(b\overline{x})^{2s}\overline{g}\left(\frac{1}{\overline{b}\overline{x}}\right)},\frac{\overline{y}\overline{h}(\overline{y})}{\overline{h}(0)(\overline{ay})^{2s}\overline{h}\left(\frac{1}{\overline{ay}}\right)}\right),$$

where $\overline{F}=\{(0,0),(\overline{\alpha}_i,\overline{\beta}_i),(-\overline{\alpha}_i,-\overline{\beta}_i) : i=1\dots s\}$ is the kernel of the ℓ-isogeny on general Huff's curve, $\ell=2s+1$, $\overline{a}'=\overline{a}^\ell\overline{h}(0)^2$, $\overline{b}'=\overline{b}^\ell\overline{g}(0)^2$, $\overline{g}(x)=\prod_{i=1}^s\left(x^2-\overline{\alpha}_i^2\right)$, and $\overline{h}(y)=\prod_{i=1}^s\left(y^2-\overline{\beta}_i^2\right)$.

Now let $F=\{(0,0),(\alpha_i,\beta_i),(-\alpha_i,-\beta_i) : i=1\dots s\}$, where $-(\alpha_i,\beta_i)=(-\alpha_i,-\beta_i)$, be the kernel of an isogeny ψ of degree ℓ, where $\ell=2s+1$.

Let us define functions $g(x)$ and $h(x)$ as $g(x)=\prod_{i=1}^s\left(x^2-\alpha_i^2\right)$, $h(y)=\prod_{i=1}^s\left(y^2-\beta_i^2\right)$. Let ξ be an isomorphism from Huff's curve $H_{a,b}$ to general Huff's curve $G_{\overline{a},\overline{b}}$, where $\overline{a}=\frac{1}{b^2},\overline{b}=\frac{1}{a^2}$. For $P=(x,y)$ the isomorphism ξ has the form $\overline{P}=\xi(P)=(ax,by)=(\overline{x},\overline{y})$,

Using isomorphism $\xi : G_{\overline{a},\overline{b}}\rightarrow H_{a,b}$ (see [7]), we can make the following transformations. Using ξ, we can transform the equation for isogeny evaluation using kernel polynomials on general Huff's curves as follows

$$\begin{aligned}\overline{\psi}(\overline{P}) &= \overline{\psi}(\overline{x},\overline{y}) = \overline{\psi}\left(\xi_x(P),\xi_y(P)\right)\\ &= \left(\frac{axa^{2s}g(x)}{a^{2s}g(0)\left(\overline{b}a^2x\right)^{2s}g\left(\frac{1}{\overline{b}a^2x}\right)}, \frac{byb^{2s}h(y)}{b^{2s}h(0)\left(\overline{a}b^2y\right)^{2s}h\left(\frac{1}{\overline{a}b^2y}\right)}\right)\\ &= \left(\frac{axg(x)}{g(0)\left(\frac{1}{a^2}a^2x\right)^{2s}g\left(\frac{1}{\frac{1}{a^2}a^2x}\right)}, \frac{byh(y)}{h(0)\left(\frac{1}{b^2}b^2y\right)^{2s}h\left(\frac{1}{\frac{1}{b^2}b^2y}\right)}\right)\\ &= \left(\frac{axg(x)}{g(0)\left(x\right)^{2s}g\left(\frac{1}{x}\right)}, \frac{byh(y)}{h(0)\left(y\right)^{2s}h\left(\frac{1}{y}\right)}\right).\end{aligned}$$

Making further transformations, one obtains

$$\begin{aligned}\psi(P) &= \xi\left(\overline{\psi}(\overline{P})\right) = \overline{\psi}\left(\frac{\xi_x(P)}{a'}, \frac{\xi_y(P)}{b'}\right)\\ &= \left(\frac{axg(x)}{(-1)^s\frac{a}{g(0)}g(0)\left(x\right)^{2s}g\left(\frac{1}{x}\right)}, \frac{byh(y)}{(-1)^s\frac{b}{h(0)}h(0)\left(y\right)^{2s}h\left(\frac{1}{y}\right)}\right)\\ &= \left(\frac{(-1)^sxg(x)}{x^{2s}g\left(\frac{1}{x}\right)}, \frac{(-1)^syh(y)}{y^{2s}h\left(\frac{1}{y}\right)}\right). \qquad (7)\end{aligned}$$

Additionally $a' = (-1)^s\frac{a}{g(0)}$ and $b' = (-1)^s\frac{b}{h(0)}$.

Using Corollary 2, we conclude that $g_2(\alpha)$ and $h_2(\alpha)$ can be computed using index system from Example 1 for any $\alpha \in \mathbb{F}_q$ in $\tilde{O}\left(\sqrt{\ell}\right)$ operations.

Theorem 2. *Using the compression function $f_2(P) = xy = r$ one obtains that*

$$f_2\left(\psi(P)\right) = \frac{rg_2\left(\frac{r(ar+b)}{br+a}\right)h_2\left(\frac{r(br+a)}{ar+b}\right)}{r^{2s}g_2\left(\frac{br+a}{r(ar+b)}\right)h_2\left(\frac{ar+b}{r(br+a)}\right)},$$

where $r_i = \alpha_i\beta_i, g_2(z) = \prod_{i=1}^{s}\left(z - \frac{r_i(ar_i+b)}{br_i+a}\right)$, $h_2(z) = \prod_{i=1}^{s}\left(z - \frac{r_i(br_i+a)}{ar_i+b}\right)$, $a' = (-1)^s\frac{a}{g_2(0)}$ and $b' = (-1)^s\frac{b}{h_2(0)}$.

Proof. The formula for evaluation of the isogeny $f_2\left(\psi(P)\right)$ is a straightforward adaptation of the formula (7). What is more, function $g_2(z)$ can be computed using Algorithm 1, where for this function holds that $f(P) = \frac{r(ar+b)}{br+a}$, where $r = xy$ for any $P = (x, y)$. Functions F_0, F_1, F_2 appearing in Algorithm 1 are equal to s_0, s_1, s_2 given by equation (4), respectively.

In the same manner, function $h_2(z)$ can be computed using Algorithm 1, where for this function holds that $f(P) = \frac{r(br+a)}{ar+b}$, where $r = xy$ for any $P = (x, y)$. Functions F_0, F_1, F_2 appearing in Algorithm 1 are equal to t_0, t_1, t_2 given by equation (6), respectively. □

Theorem 3. *Using the compression function $f_{4,x^2}(P) = x^2 = r$ one obtains that*

$$f_{4,x^2}\left(\psi(P)\right) = \frac{r g_{4,x^2}\left(r\right)^2}{r^{2s} g_{4,x^2}\left(\frac{1}{r}\right)^2},$$

where $\tilde{D}(r_i, a, b) = \frac{\tilde{L}(r_i,a,b)}{\tilde{M}(r_i,a,b)}$ is a rational function of r_i returning $f_{4,y^2}([2]P)$ having $r_i = f_{4,x^2}(Q_i) = \alpha_i^2$. Functions $\tilde{L}(r, a, b), \tilde{M}(r, a, b)$ are given by equation (8) and $g_{4,x^2}(z) = \prod_{i=1}^{s}\left(z - r_i^2\right)$, $h_{4,x^2}(z) = \prod_{i=1}^{s}\left(z - \tilde{D}(r_i, a, b)\right)$ and $a' = (-1)^s \frac{a}{g_{4,x^2}(0)}$ and $b' = (-1)^s \frac{b}{h_{4,x^2}(0)}$.

Proof. We begin by showing some observations. Let us note that for elements of kernel F, having compression $r_i = \alpha_i^2$, we cannot decide what the value of is β_i, because all points $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i), (\alpha_i, -\frac{1}{\beta_i}), (-\alpha_i, \frac{1}{\beta_i})$ lie on the curve $H_{a,b}$ and all these points have the same value of compression f_{4,x^2}, but only two of these points belong to the kernel F: $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)$. Of course, having $r_i = \alpha_i^2$, one can find α_i by computing roots of degree 2 polynomial $r - x^2$. In such a case, both roots α_i and $-\alpha_i$ are proper because points $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)$ both belong to the kernel F of ℓ-isogeny.

In the next step, having α_i (or $-\alpha_i$, we may assume that we have α_i) it is necessary to find proper β_i. Having α_i and using Huff's curve equation, one can find two possible values of y-coordinate: $y_1 = \beta_i$ or $y_2 = -\frac{1}{\beta_i}$. Unfortunately, in this case, only one value is proper. Let us note that if one computes ℓ-isogeny, where ℓ is an odd number, then each element (α_i, β_i) of the kernel F has odd order, but $-(\alpha_i, \beta_i) + (0 : 1 : 0) = (\alpha_i, -\frac{1}{\beta_i})$ has order equal to 2ℓ. So one can, for both possible values of y-coordinates $y_1 = \beta_i, y_2 = \frac{1}{\beta_i}$, check the order of element (α_i, y_j), for $j = 1, 2$ and then decide which element is the correct element of the kernel F. Unfortunately, this method seems slow and generally useless in practical implementations.

Now we will show another, much faster way of obtaining necessary compressions $\{\beta_i^2 : i = i, \dots, s\}$ of points of kernel F. First, let us note that we are interested in the computation of ℓ-degree isogenies, where ℓ is odd. For simplicity of the proof presented below, we also assume that ℓ is prime. What is more, if (α_i, β_i) belongs to the kernel of the isogeny, then such point has order ℓ. Points $(-\alpha_i, \frac{1}{\beta_i}), (\alpha_i, -\frac{1}{\beta_i})$ can be obtained by translation of points $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)$ by 2-torsion point $(0 : 1 : 0)$, so their order must be equal to 2ℓ. Now we will show the most important observation. Let us note that for $P \in \{(\alpha_i, \beta_i), (-\alpha_i, -\beta_i), (\alpha_i, -\frac{1}{\beta_i}), (-\alpha_i, \frac{1}{\beta_i})\}$, however, one can obtain two different values of compressions $f_{4,y^2}(Q_i)$, because $f_{4,y^2}(Q_i) = \beta_i^2$ or $f_{4,y^2}(Q_i) = \frac{1}{\beta_i^2}$, but for point $[2]Q_i$ there is only one possible value of compression $f_{4,y^2}([2]Q_i)$. What is more, if $Q_i \in F$, then $[2]Q_i \in F$ if and only if $\#F$ is odd.

Points $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)$ are of order ℓ and points $(\alpha_i, -\frac{1}{\beta_i}) = -(\alpha_i, \beta_i) + (0 : 1 : 0), (-\alpha_i, \frac{1}{\beta_i}) = (\alpha_i, \beta_i) + (0 : 1 : 0)$ are of order 2ℓ. Now let us note that for $Q_i \in \{(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)\}$, point $[2]Q_i$ is of order ℓ and also $[2]Q_i$ belongs to the kernel F generated by any of points $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)$.

On the other hand, for $Q_i \in \{(\alpha_i, -\frac{1}{\beta_i}), (-\alpha_i, \frac{1}{\beta_i})\}$, however, point Q_i is of order 2ℓ, but point $[2]Q_i$ is of order ℓ and also $[2]Q_i$ belongs to the kernel F generated by any of points $(\alpha_i, \beta_i), (-\alpha_i, -\beta_i)$, because $[2](\alpha_i, -\frac{1}{\beta_i}) = [2](-(\alpha_i, \beta_i) + (0 : 1 : 0)) = -[2](\alpha_i, \beta_i)$ and similarly $[2](-\alpha_i, \frac{1}{\beta_i}) = [2]((\alpha_i, \beta_i) + (0 : 1 : 0)) = [2](\alpha_i, \beta_i)$. Because points $[2](\alpha_i, \beta_i)$ and $-[2](\alpha_i, \beta_i)$ are opposite, their values of compression f_{4,y^2}, are the same.

It means that for any point $Q_i \in \{(\alpha_i, \beta_i), (-\alpha_i, -\beta_i), (\alpha_i, -\frac{1}{\beta_i}), (-\alpha_i, \frac{1}{\beta_i})\}$, the value of compression $f_{4,y^2}([2]Q_i)$ is the same.

Let us note that if $\tilde{D}(r_i, a, b)$ is a rational function of r_i returning $f_{4,y^2}([2]Q_i)$ having $r_i = f_{4,x^2}(Q_i) = \alpha_i^2$, then $\prod_{i=1}^{s} \left(z - \beta_i^2\right) = \prod_{i=1}^{s} \left(z - \tilde{D}(r, a, b)\right)$, where holds that $f_{4,y^2}(Q_i) = \beta_i^2$.

What is more, having compression of any element of the kernel $f_{4,y^2}(Q_i) = \beta_i^2$ we can also obtain other elements. It is possible using, for example, formulas for differential addition given in (4) and doubling, obtained by using the method described in [6].

This formula is given by $\frac{L(r,b,a)}{M(r,b,a)}$ (remember that Huff's curve is symmetric: $H(x, y)_{a,b} = H(y, x)_{b,a}$), where $L(r, a, b)$ and $M(r, a, b)$ are provided by (5).

The formula for evaluation of the isogeny $f_{4,x^2}(\psi(P))$ is a straightforward adaptation of the formula (7). What is more, function $g_{4,x^2}(z)$ can be computed using Algorithm 1, where for this function holds $f(P) = r$, where $r = x^2$ for any $P = (x, y)$. Functions F_0, F_1, F_2 appearing in Algorithm 1 are equal to s_0, s_1, s_2 given by equation (4), respectively.

In the same manner, function $h_{4,x^2}(z)$ can be computed using Algorithm 1, where for this function holds that one can replace $f(P) = r$, where $r = y^2$

for any $P = (x, y)$ by $f_{4,y^2}([2]P) = \tilde{D}(r, a, b)$. Functions F_0, F_1, F_2 appearing in Algorithm 1 are equal to t_0, t_1, t_2 given by equation (6), respectively.

The correctness of the formulas presented above can be checked using the program *Huff_Correctness_x_square* from [5]. □

Now we will show how to compute $f_{4,y^2}([2]P)$ having $f_{4,x^2}(P) = r$.

Theorem 4. *If $f_{4,x^2}(P) = r$, then $f_{4,y^2}([2]P) = \frac{\tilde{L}(r,a,b)}{\tilde{M}(r,a,b)}$, where*

$$\begin{aligned}\tilde{L}(r,a,b) &= 4\frac{a^2}{b^2}r(r+1)^2,\\ \tilde{M}(r,a,b) &= r^4 + \frac{4a^2-4b^2}{b^2}r^3 + \frac{-8a^2+6b^2}{b^2}r^2 + \frac{4a^2-4b^2}{b^2}r + 1.\end{aligned} \tag{8}$$

Explanation 2. This formula can be found using the program from [3] and using modifications for high-degree compressions described in [6]. Formula for $f_{4,y^2}([2]P)$ knowing $f_{4,x^2}(P) = r$ can be found using the method described in [6]. The correctness of the formulas presented above can be checked using the program *Huff_Correctness_x_square* from [5].

Using Corollary 2, we conclude that $g_{4,x^2}(\alpha)$ and $h_{4,x^2}(\alpha)$ can be computed using index system from Example 1 for any $\alpha \in \mathbb{F}_q$ in $\tilde{O}\left(\sqrt{\ell}\right)$ operations.

4. Conclusion

This paper presents the Velusqrt method's application to Huff's curve model. Although the formula for the computation of ℓ-isogeny using kernel polynomial for general Huff's curve is known and was given in [11], we found a similar formula for the case of Huff's curves. What is more, we presented different compression functions suitable for such applications. Presented by us, compression functions of degree 4 seem to be efficient for evaluating ℓ-isogeny. They seem to be also reasonable for computation of the ℓ-isogenous curves.

For all presented by us compression functions, one can use the same index system as presented in Example 1 to apply the Velusqrt algorithm (Algorithm 1).

It is also worth noting that it is possible to obtain very similar formulas for isogeny evaluation and computation formulas using the Velusqrt method in the case of general Huff's curves. In such a case, one can use the compression function $\overline{f}_{4,x^2} = x^2$ or $\overline{f}_{4,y^2} = y^2$. Because obtaining these formulas is straightforward, we omit their computations in this paper.

Acknowledgments. I want to thank Robert Dryło and Tomasz Kijko for joint works on elliptic curves' compression functions. Moreover, I want to thank Robert Dryło for automating the process of finding differential addition and doubling formulas for a given compression function using the Gröbner basis mechanism and being the lead author of the programs from [3] and [6] that I adapted in this paper.

I also want to thank anonymous reviewers for their comments and suggestions, that improved the paper.

References

[1] D. Bernstein, L. D. Feo, A. Leroux and B. Smith, Faster computation of isogenies of large prime degree, *arXiv:2003.10118* (2020), https://arxiv.org/abs/2003.10118.

[2] J. Chávez-Saab, J.-J. Chi-Domínguez, S. Jaques and F. Rodríguez-Henríquez, The SQALE of CSIDH: Square-root Vélu Quantum-resistant isogeny Action with Low Exponents, *IACR Cryptol. ePrint Arch.* (2020), https://eprint.iacr.org/2020/1520.

[3] R. Dryło, T. Kijko and M. J. Wroński, Determining Formulas Related to Point Compression on Alternative Models of Elliptic Curves, *Fundamenta Informaticae* **169** (2019), 285–294.

[4] R. Dryło, T. Kijko and M. J. Wroński, Efficient Montgomery-like formulas for general Huff's and Huff's elliptic curves and their applications to the isogeny-based cryptography, *IACR Cryptol. ePrint Arch.* (2020), https://eprint.iacr.org/2020/526.

[5] R. Dryło, T. Kijko and M. J. Wroński, 2020,
https://github.com/Michal-Wronski/Huff-compression.git
https://github.com/Michal-Wronski/Huff-compression/blob/master/
Huff_Correctness_x_square.magma and
Huff_diff_add_doub_rec_correctness_checking.magma.

[6] R. Dryło, T. Kijko and M. J. Wroński, High-degree compression functions on alternative models of elliptic curves and their applications, *Fundamenta Informaticae* **184** (2021), 107–139.

[7] R. Dryło, T. Kijko and M. J. Wroński, Arithmetic using compression on elliptic curves in Huff's form and its applications, *International Journal of Electronics and Telecommunications* **67** (2021), 193–200.

[8] M. Joye, M. Tibouchi and D. Vergnaud, Huff's model for elliptic curves, In: International Algorithmic Number Theory Symposium, 2010, 234–250.

[9] D. Kohel, Efficient arithmetic on elliptic curves in characteristic 2, In: International Conference on Cryptology in India, 2012, 378–398.

[10] P. L. Montgomery, Speeding the Pollard and elliptic curve methods of factorization, *Mathematics of Computation* **48** (1987), 243–264.

[11] D. Moody and D. Shumow, Analogues of Vélu's formulas for isogenies on alternate models of elliptic curves, *Mathematics of Computation* **85**, 1929–1951.

[12] H. Wu and R. Feng, Elliptic curves in Huff's model, *Wuhan University Journal of Natural Sciences* **17** (2012), 473–480.

MICHAŁ WROŃSKI
DEPARTMENT OF CYBERNETICS
MILITARY UNIVERSITY OF TECHNOLOGY
KALISKIEGO STR. 2, WARSAW
POLAND

E-mail: michal.wronski@wat.edu.pl