A Note on Scalar Multiplication Using Division Polynomials

Binglong Chen, Chuangqiang Hu and Chang-An Zhao

Abstract

Scalar multiplication is the most important and expensive operation in elliptic curve cryptosystems. In this paper we improve the efficiency of the Elliptic Net algorithm to compute scalar multiplication by using the equivalence of elliptic nets. The proposed method saves $four$ multiplications in each iteration loop. Experimental results also indicates that our algorithm will be more efficient than the previously known results in this line.

Index Terms

elliptic curve cryptosystems, division polynomial, elliptic net, scalar multiplication, side channel attack.

I. Introduction

Elliptic curve cryptosystems (ECC) proposed by Koblitz [1] and Miller [2] independently play a key role in secure transmission through an insecure channel. ECC may be a valid candidate in restricted environments such as smart cards because of short key sizes. Since scalar multiplication is the most vital and expensive operation in the implementation of ECC, improving the speed of scalar multiplication has always been the most important focus in elliptic curve cryptography.

The basic and classical method to compute scalar multiplication is the double-and-add and tangent-and-chord algorithm. A lot of important techniques have been presented to accelerate the computation of scalar multiplication in this research. We refer the reader to the books of [3], [4], [5], [6] for more comprehensive treatments.

On the other hand, Miller pointed out that elliptic curve scalar multiplication can be computed via division polynomials in polynomial time while he did not give a concrete and explicit algorithm [2]. In fact, division polynomials are closely related to elliptic divisibility sequences which were studied by Ward [7]. Shipsey provided an explicit method to compute the m-th term of an elliptic divisibility sequence in polynomial time [8]. Later, Stange generalized the elliptic divisibility sequences to higher rank and gave the definition of elliptic nets [9]. In some sense, elliptic nets (or elliptic divisibility sequences) can be regarded as a powerful tool for computation. For example, Shipsey made use of them to solve the elliptic curve discrete logarithm problem in certain cases [8], Stange proposed the Elliptic Net algorithm to compute the pairings [9] and Kanayama $et\ al.$ adapted Stange's algorithm to compute elliptic curve scalar multiplication [9].

The Elliptic Net algorithm to compute scalar multiplication have three main advantages. First, there are no inverse operations involved in the whole loop of the algorithm. Therefore, it may be suitable for a restricted environments of computational power. Second, the cost of the $Double$ step is the same as that of the $DoubleAdd$ step in each iteration loop. In other words, the cost of each iteration loop is independent of the Hamming weight of the scalar. Hence this algorithm can resist side channel attacks. Finally, the code of the main loop can be easily written and suitable for different finite fields. However, the efficiency of this algorithm should be improved compared to that of the classical double-and-add and tangent-and-chord method.

In this paper, we speed up the Elliptic Net algorithm to compute scalar multiplication by using the equivalence of elliptic nets (or elliptic divisibility sequences). Inspired by the work of [9], we propose a novel $Double$ and $DoubleAdd$ steps for scalar multiplication on curves in large prime characteristics. In certain cases, the improvement will save $four$ multiplications

B. Chen, C. Hu and C.-A. Zhao are with the School of Mathematics and Computational Science, School of Mathematics and Computational Science, Sun Yat-sen University, Guangzhou 510275, P.R.China. {mcscbl, zhaochan3}@mail.sysu.edu.cn, huchuangqiang@qq.com

in each iteration loop compared to the previous results. This indicates that we can attain higher performance with the increase of the bit-length of the scalar. We also discuss the case in even characteristic which may be advantageous to design hardware accelerators to provide the desired performance levels. Experimental results indicate that the proposed algorithm will be more efficient than the previous results [10] although it is slower than the traditional double-and-add method for scalar multiplication in Jacobian coordinate systems.

The rest of the paper is organized as follows. In Section II, we recall the definition of elliptic curves and the Elliptic Net algorithm to compute scalar multiplication. Section III gives our main results. Section IV provides efficiency analysis and implementation results. Finally, we draw our conclusion in Section V.

II. Preliminaries

In this section, we will recall the definition of elliptic curves with Weierstrass equation briefly and the Elliptic Net algorithm to compute scalar multiplication (i.e., the scalar multiplication algorithm using division polynomials).

A. Elliptic curves

Let K be a field. Denote by $Char(K)$ the characteristic of K. Suppose that E is an elliptic curve over K. If $Char(K) \neq 2, 3$, then the Weierstrass equation of the curve E can be given as follows.

$$y^2 = x^3 + Ax + B, \quad 4A^3 + 27B^2 \neq 0. \tag{1}$$

Let m be a positive integer. One can define *division polynomials* $\psi_m \in \mathbb{Z}[A, B, x, y]$ [11] inductively as follows:

$$\begin{aligned} &\psi_0 = 0,\ \psi_1 = 1,\ \psi_2 = 2y, \\ &\psi_3 = 3x^4 + 6Ax^2 + 12Bx - A^2, \\ &\psi_4 = 4y(x^6 + 5Ax^4 + 20Bx^3 - 5A^2x^2 - 4ABx - 8B^2 - A^3). \end{aligned} \tag{2}$$

For $n \geq 5$, we can define ψ_n recursively:

$$\begin{aligned} \psi_{2m+1}\psi_1 &= \psi_{m+2}\psi_m^3 - \psi_{m-1}\psi_{m+1}^3 \quad (m \geq 2), \\ \psi_{2m}\psi_2 &= \psi_m(\psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2) \quad (m \geq 3). \end{aligned} \tag{3}$$

We define two auxiliary polynomials ϕ_m and ω_m by

$$\begin{aligned} \phi_m &= x\psi_m^2 - \psi_{m+1}\psi_{m-1} \\ 4y\omega_m &= \psi_{m+2}\psi_{m-1}^2 - \psi_{m-2}\psi_{m+1}^2. \end{aligned} \tag{4}$$

Let $P = (x_1, y_1)$ be a point on E. Then for $m \geq 2$

$$[m]P = (\frac{\phi_m(P)}{\psi_m(P)^2}, \frac{\omega_m(P)}{\psi_m(P)^3}). \tag{5}$$

If $Char(K) = 2$, then the equation of a non-supersingular elliptic curve E is in the form

$$y^2 + xy = x^3 + a_2x^2 + a_6. \tag{6}$$

In this case, the first few *division polynomials* $\psi_m \in \mathbb{Z}[A, B, x, y]$ [1] will be:

$$\begin{aligned} &\psi_0 = 0,\ \psi_1 = 1,\ \psi_2 = x, \\ &\psi_3 = x^4 + x^3 + a_6,\ \psi_4 = x^6 + x^2a_6. \end{aligned} \tag{7}$$

Similarly, the division polynomials ψ_n for $n \geq 5$ can be defined by Equation (3) recursively. Let $P = (x_1, y_1)$ be a point on E. Then for $m \geq 2$ [1]

$$[m]P = (\frac{\phi_m(P)}{\psi_m(P)^2}, x_1 + y_1 + (1 + x_1 + \frac{y_1}{x_1})\frac{\psi_{m-1}(P)\psi_{m+1}(P)}{\psi_m(P)^2} + \frac{\psi_{m+1}(P)^2\psi_{m-2}(P)}{\psi_2(P)\psi_m(P)^3}), \tag{8}$$

where the polynomial ϕ_m has the same meaning as in Equation (4). The computation of scalar multiplication $[m]P$ plays a key role in the implementation of elliptic curve cryptosystems. The classical method is the double-and-add and tangent-and-chord algorithm. However, Miller also pointed out that scalar multiplication can be computed by using division polynomials [2] in polynomial time. Kanayama $et\ al.$ implemented this method [10] which was essentially adapted from the Elliptic Net algorithm to computing the pairings by Stange [9].

B. Elliptic net algorithm to compute scalar multiplication

In this subsection, we define the elliptic divisibility sequence and recall the Elliptic Net algorithm to compute scalar multiplication [10].

We define $\psi_{-n}(P) = -\psi_n(P)$. For any integer n, let $W(n) = \psi_n(P)$. Then the sequence $\{W(n)\}$ is exactly an elliptic divisibility sequence which was first studied by Ward [7]. An elliptic divisibility sequence is **defined over the field** K if all of the terms in the sequence $\{W(n)\}$ belong to K. Two elliptic divisibility sequences $\{u_n\}$ and $\{v_n\}$ are **equivalent** [8], [9] if there exist two nonzero constants α and β such that $u_n = \alpha\beta^{n^2}v_n$.

Stange generalized the concept of elliptic divisibility sequences to higher rank and gave the definition of elliptic nets associated to an elliptic curve E and arbitrary many points on E [9]. In this sense, the sequence $\{W(n)\}$ can regarded as an elliptic net of rank one. Note that Rachel Shipsey provided a double-and-add method of calculating the m-th term of an elliptic divisibility sequence in polynomial time by using the recurrence formulas (3) in her thesis [8]. In addition, if the terms W_λ for $m-2 \leq \lambda \leq m+2$ can be obtained, then the coordinates of $[m]P$ will be determined. Kanayama $et\ al.$ adapted the elliptic net algorithm by Stange [9] for computing elliptic curve scalar multiplication. Here we cite Algorithms 1 and 2 of [10] to compute the terms W_λ for $m-2 \leq \lambda \leq m+2$ as shown Algorithms 2.1 and 2.2 for convenience. Denote the multiplication, squaring, and inversion in the finite field by M, S and I, respectively. We also give the cost of each line in Algorithm 2.1.

Note that the term $W(2)$ is not equal to one generally. Hence this will cost $4M$ for multiplying with $W(2)$ in each iteration loop (see Algorithm 2.1). Inspired by this observation and making use of results of Stange [9], we will give an improvement of Algorithm 2.1 and save $4M$ in each iteration loop.

Algorithm 2.1 Double and DoubleAdd Algorithm

Input: Block $V = [V[0], V[1], \cdots, V[7]]$ centered at k boolean add

Output: Block centered at $2k$ if $add == 0$ centered at $2k+1$ if $add == 1$

1. for $i = 0$ to 5 do

$S[i] \leftarrow V[i+1]^2$; $P[i] \leftarrow V[i] * V[i+2]$; //

$6 \cdot (1S + 1M)$

end for;

2. if $add == 0$ then

for $i = 1$ to 4 do

$V[2*i-2] \leftarrow S[i-1]*P[i] - S[i]*P[i-1]$; //$4\cdot(2M)$

$V[2*i-1] \leftarrow (S[i-1]*P[i+1] - S[i+1]*P[i-1]) \cdot W(2)$; //$4\cdot(3M)$

end for;

else

for $i = 1$ to 4 do

$V[2*i-2] \leftarrow (S[i-1]*P[i+1] - S[i+1]*P[i-1]) \cdot W(2)$; //$4\cdot(2M)$

$V[2*i-1] \leftarrow S[i]*P[i+1] - S[i+1]*P[i]$; //$4\cdot(3M)$

end for;

end if;

3. return V

Algorithm 2.2 Scalar Multiplication Algorithm via Elliptic Nets

Input: Integer $m = (d_l d_{l-1} \cdots d_0)_2$ with $d_k = 1$. The first few terms $a = W(2)$, $b = W(3)$, and $c = W(4)$ of the elliptic nets of rank *one* associated to the point P.

Output: the values of $W(m-2)$, $W(m-1)$, $W(m)$, $W(m+1)$, and $W(m+2)$ in the elliptic nets of rank *one* associated to the point P.

1. $V \leftarrow [-a, -1, 0, 1, a, b, c, a^3c - b^3]$
2. for $i := l - 1$ to 0 by 1 do
   if $d_i = 0$ then
   $V \leftarrow Double(V)$
   $V \leftarrow DoubleAdd(V)$
   end if
   end for
3. return $V[1]$, $V[2]$, $V[3]$, $V[4]$, and $V[5]$. // terms $W(m-2)$, $W(m-1)$, $W(m)$, $W(m+1)$, and $W(m+2)$ respectively

III. Main results

In this section, our main results will be given. Observe that $W(2)$ will always appear in the Double and DoubleAdd algorithm. Inspired by the optimization ideas of the Elliptic Net algorithm [9], we will speed up Algorithm 2.2 by changing $W(2)$ to *one*. This will save *four* multiplications in each iteration loop.

The following lemma will turn out to be useful in the sequel.

Lemma 1: Let K be a finite field with q elements, where q satisfies that $\gcd(q-1, 3) = 1$. Let $\{W(i)\}$ be the elliptic divisibility sequence defined over K with $W(2) \neq 0$. Then there exists an elliptic divisibility sequence $\{\hat{W}(i)\}$ defined over K such that $\hat{W}(2) = 1$ which is equivalent to the sequence $\{W(i)\}$.

Proof: According to the assumption, the elliptic divisibility sequence $\{W(i)\}$ is defined over K. Since $\gcd(q-1, 3) = 1$, one can find that a cubic root θ of $W(2)^{-1}$ lies in K, i.e, $\theta^3 = W(2)^{-1}$. Let $\hat{W}(i) = \theta^{i^2-1} W(i)$ for any integer i. Then the sequence $\{\hat{W}(i)\}$ is defined over K since both θ and $W(i)$ belong to K. One can verify that $\hat{W}(2) = 1$. This completes the proof of Lemma 1. □

Lemma 2: Let K be a finite field with q elements and $Char(K) \geq 5$. Suppose that $\gcd(q-1, 3) = 1$. Let E be an elliptic curve over K defined by Equation (1): $y^2 = x^3 + Ax + B$. Assume that $P = (x_1, y_1)$ is a rational point on $E(K)$. Let $\{W(i)\}$ be the elliptic divisibility sequence with $W(2) \neq 0$ associated to E and P from the division polynomials. Choose θ such that $\theta^3 = W(2)^{-1} = (2y_1)^{-1}$. Define a sequence $\{\hat{W}(i)\}$ by $\hat{W}(i) = \theta^{i^2-1} W(i)$ with $\hat{W}(2) = 1$ over K. Then the coordinates of $[m]P = (x_m, y_m)$ can be calculated from the sequence $\{\hat{W}(i)\}$, i.e.,

$$x_m = x_1 - \theta^{-2} \frac{\hat{W}(m-1)\hat{W}(m+1)}{\hat{W}(m)^2}$$

and

$$y_m = \frac{y_1(\hat{W}(m-1)^2\hat{W}(m+2) - \hat{W}(m+1)^2\hat{W}(m-2))}{\hat{W}(m)^3}.$$

Proof: Note that $W(i) = \theta^{1-i^2}\hat{W}(i)$ for any integer i and $\theta^3 = (2y_1)^{-1}$. By Equation (5) we get

$$\begin{aligned} x_m &= x_1 - \frac{W(m-1)W(m+1)}{W(m)^2} \\ &= x_1 - \frac{\theta^{1-(m-1)^2}\hat{W}(m-1)\theta^{1-(m+1)^2}\hat{W}(m+1)}{\theta^{2(1-m^2)}\hat{W}(m)^2} \\ &= x_1 - \theta^{-2}\frac{\hat{W}(m-1)\hat{W}(m+1)}{\hat{W}(m)^2} \end{aligned}$$

and

$$\begin{aligned} y_m &= \frac{W(m-1)^2W(m+2) - W(m+1)^2W(m-2)}{4y_1W(m)^3} \\ &= \frac{\theta^{2(1-(m-1)^2)+1-(m+2)^2}\hat{W}(m-1)^2\hat{W}(m+2) - \theta^{2(1-(m+1)^2)+1-(m-2)^2)}\hat{W}(m+1)^2\hat{W}(m-2)}{4y_1\theta^{3(1-m^2)}\hat{W}(m)^3} \\ &= \frac{y_1(\hat{W}(m-1)^2\hat{W}(m+2) - \hat{W}(m+1)^2\hat{W}(m-2))}{\hat{W}(m)^3}. \end{aligned}$$

This completes the proof of the lemma. □

For an elliptic curve over finite fields with even characteristic, we can also obtain the similar results as follows. For our interests, we just consider the non-supersingular elliptic curves although it is also clear in the case of supersingular curves.

Lemma 3: Let K be a finite field with q elements and $Char(K) = 2$. Suppose that $\gcd(q-1, 3) = 1$. Let E be an elliptic curve over K defined by Equation (6): $y^2 + xy = x^3 + a_2x^2 + a_6$. Assume that $P = (x_1, y_1)$ is a rational point on $E(K)$. Let $\{W(i)\}$ be the elliptic divisibility sequence with $W(2) \neq 0$ associated to E and P from the division polynomials. Choose θ such that $\theta^3 = W(2)^{-1} = x_1^{-1}$. Define a sequence $\{\hat{W}(i)\}$ by $\hat{W}(i) = \theta^{i^2-1}W(i)$ with $\hat{W}(2) = 1$ over K. Then the coordinates of $[m]P = (x_m, y_m)$ can be calculated from the sequence $\{\hat{W}(i)\}$, i.e.,

$$x_m = x_1 + \theta^{-2}\frac{\hat{W}(m-1)\hat{W}(m+1)}{\hat{W}(m)^2}$$

and

$$y_m = x_1 + y_1 + (1 + x_1 + \frac{y_1}{x_1})\theta^{-2}\frac{\hat{W}(m-1)\hat{W}(m+1)}{\hat{W}(m)^2} + \frac{x_1\hat{W}(m+1)^2\hat{W}(m-2)}{\hat{W}(m)^3}.$$

Proof: The calculation can be done by a similar way in the proof of Lemma 2. □

On the basis of the above lemmas, one can compute the terms of the sequences $\{\hat{W}(i)\}$. It should be remarked that the whole elliptic divisibility sequence with $\hat{W}(1) = \hat{W}(2) = 1$ can be defined over K under the condition $\gcd(q-1, 3) = 1$. This will save the cost of multiplying with $\hat{W}(2)$. In the next section, we will give a detailed efficiency analysis.

IV. Efficiency consideration and implementation results

In this section, the efficiency of the proposed method will be given. Also, we will compare it with the previous methods and show some experimental results for implementation of scalar multiplication based on the elliptic divisibility sequences (or the elliptic net). We will consider the implementation of elliptic curve cryptosystems over finite fields in large prime characteristic.

In the following, we always assume that the cardinality of the finite field q satisfies $(q-1, 3) = 1$. This implies that we can use the equivalent elliptic divisibility sequence $\{\hat{W}(i)\}$ for scalar multiplication. Note that in the implementation of elliptic curve cryptosystems the rational point P will be often chosen to be a public parameter and $W(2)$ is only related to the coordinates of the point P. Thus we neglect the cost of computing the cubic root of $W(2)$.

As in the above lemmas, let θ be an element of K such that $\theta^3 = W(2)^{-1}$. Then we have

$$\begin{aligned} &\hat{W}(1) = 1, \hat{a} = \hat{W}(2) = 1, \hat{b} = \hat{W}(3) = \theta^8 \cdot W(3), \\ &\hat{c} = \hat{W}(4) = \theta^{15} \cdot W(4), \hat{W}(5) = \hat{a}^3\hat{c} - \hat{b}^3 = \hat{c} - \hat{b}^3. \end{aligned}$$

Then the initial vector should be changed to

$$[-1, -1, 0, 1, 1, \hat{W}(3), \hat{W}(4), \hat{W}(5)] = [-1, -1, 0, 1, 1, \hat{b}, \hat{c}, \hat{c} - \hat{b}^3],$$

in Algorithm 2.2.

It is easy to see that the main advantage of Algorithm 2.1 is to resist side channel attack. However, further optimization will be required compared to the classical method for elliptic curve scalar multiplication. Here we give some possible ways of further optimization for large prime characteristic and even characteristic, respectively.

A. Large prime characteristic

In this subsection, we will focus on the optimization of scalar multiplication on elliptic curves over finite fields with large prime characteristic. On the one hand, the number of the terms of the vector which should be updated can be reduced to $seven$, which still makes Algorithms 2.1 and 2.2 work correctly (see [12] for details). This adjustment will make each $Double$ step save the cost of $1S + 4M$, but each $DoubleAdd$ step involves one inverse operation. Therefore it only works in the case of the scalar m with low Hamming weight and does not consider the resistance of side channel attacks when computing $[m]P$.

On the other hand, there exist some possible trade-offs arising from different squaring/multiplication ratios in the case of large prime characteristic. Note that the values $V[i]^2$ for $1 \leq i \leq 6$ will be computed first in Algorithm 2.1. If the cost of dividing by 2 can be neglected, then we can compute

$$V[i] \cdot V[i+2] = ((V[i] + V[i+2])^2 - V[i]^2 - V[i+2]^2)/2$$

for $1 \leq i \leq 4$. This observation will convert $4M$ to $4S$ in the cost of Algorithm 2.1.

We will consider the implementation on elliptic curves over a finite field $K = \mathbb{F}_q$, where q is a large prime satisfying $(q-1, 3) = 1$. It should be remarked that there exist certain elliptic curves satisfying this property, for example, the NIST elliptic curve P-192 used in [10]. We summarize the cost of operations for $Double$ and $DoubleAdd$ steps by using different methods in Table I.

TABLE I
COMPARISON OF THE COST OF OPERATIONS FOR $Double$ AND $DoubleAdd$ STEPS IN LARGE PRIME CHARACERISTIC

Method	$Double$	$DoubleAdd$
Affine [4]	$2M + 2S + I$	$4M + 3S + 2I$
Affine-Jacobian($A = -3$) [13]	$4M + 4S$	$12M + 7S$
Algorithm 1 [10]	$26M + 6S$	$26M + 6S$
This work	$22M + 6S$	$22M + 6S$

B. Even characteristic

Now we consider the cost of operations for $Double$ and $DoubleAdd$ steps on elliptic curves over $\mathbb{F}_{2^n}$, where n is odd. In real implementations n is actually chosen to be an odd prime. Therefore, the proposed method can be applied in this case since $\gcd(2^n - 1, 3) = 1$. It is seen that squaring/multiplication trade-offs can not be used. However, the cost of one squaring can be neglected compared to that of one multiplication. We summarize the cost of operations for $Double$ and $DoubleAdd$ steps in Table II and cite the basic methods in affine and Jacobian coordinate systems for benchmarks.

TABLE II
COMPARISON OF THE COST OF OPERATIONS FOR $Double$ AND $DoubleAdd$ STEPS IN EVEN CHARACERISTIC

Method	$Double$	$DoubleAdd$
Affine [4]	$M + I$	$2M + 2I$
Affine-Jacobian [4]	$5M$	$15M$
This work	$22M$	$22M$

C. Implementation results

We implement the different methods to compute scalar multiplication on the NIST elliptic curve P-192 over finite fields with large prime characteristic used in [10](or see pages 261-262 of [4]). Our running environment specifications are listed

as follows: Windows 7 64bits, Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz$\times$ 8, and memory, 16GB. The code was written in c++ and based on NTL library [14]. The bit-length of the scalar m is 191. From Table III, we see that the proposed method will be more efficient than the previous algorithm [10] indeed while it is slower than the traditional double-and-add and tangent-and-chord method. This shows that more improvements will be required for the Elliptic Net algorithm to compute scalar multiplication .

TABLE III
TIMINGS OF SCALAR MULTIPLICATION BY DIFFERENT METHODS IN LARGE PRIME CHARACTERISTIC

Method	Time(ms)
Jacobian coordinate [13]	2.41
Algorithm 2 of [10]	4.66
This work	4.32

V. CONCLUSION

In this paper, we showed how to speed up the computation of $Double$ and $DoubleAdd$ steps by using division polynomials when computing scalar multiplication. The Elliptic Net algorithm to compute scalar multiplication can be implemented efficiently on personal modern computers and resist side channel attacks while it is slower than the classical double-and-add and tangent-and-chord method. We hope that this work may lead to more developments of scalar multiplication in this line.

ACKNOWLEDGMENTS

The work of Binglong Chen is partially supported by the by the NSFC(Grant No. 11025107). The work of Chang-An Zhao is partially supported by the NSFC (Grant No. 61472457).

REFERENCES

[1] N. Koblitz, "Constructing elliptic curve cryptosystems in characteristic 2," in *Advances in Cryptology-CRYPT0 90*, ser. Lecture Notes in Computer Science, A. J. Menezes and S. A. Vanstone, Eds. Springer Berlin Heidelberg, 1991, vol. 537, pp. 156–167.

[2] V. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology CRYPTO 85 Proceedings*, ser. Lecture Notes in Computer Science, H. Williams, Ed. Springer Berlin Heidelberg, 1986, vol. 218, pp. 417–426.

[3] I. F. Blake, G. Seroussi, and N. Smart, *Elliptic curves in cryptography*. Cambridge university press, 1999, vol. 265.

[4] D. Hankerson, S. Vanstone, and A. J. Menezes, *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2004.

[5] H. Cohen and G. Frey, Eds., *Handbook of elliptic and hyperelliptic curve cryptography*. CRC Press, 2005.

[6] S. D. Galbraith, *Mathematics of public key cryptography*. Cambridge University Press, 2012.

[7] M. Ward, "Memoir on elliptic divisibility sequences," *American Journal of Mathematics*, pp. 31–74, 1948.

[8] R. Shipsey, "Elliptic divisibility sequences," Ph.D. dissertation, Goldsmiths, University of London, 2001.

[9] K. E. Stange, "The Tate pairing via elliptic nets," in *Pairing-Based Cryptography - Pairing 2007*, ser. Lecture Notes in Computer Science, T. Takagi, T. Okamoto, E. Okamoto, and T. Okamoto, Eds. Springer Berlin Heidelberg, 2007, vol. 4575, pp. 329–348.

[10] N. Kanayama, Y. Liu, E. OKamoto, K. Saito, T. Teruya, and S. Uchiyama, "Implementation of an elliptic curve scalar multiplication method using division polynomials," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E97.A, no. 1, pp. 300–302, 2014.

[11] J. Silverman, *The arithmetic of elliptic curves*. Springer-Verlag, New York, 1986.

[12] B. Chen and C.-A. Zhao, "An improvement of the Elliptic Net algorithm," preprint, 2015.

[13] H. Cohen, A. Miyaji, and T. Ono, "Efficient elliptic curve exponentiation using mixed coordinates," in *Advances in Cryptology ASIACRYPT98*, ser. Lecture Notes in Computer Science, K. Ohta and D. Pei, Eds. Springer Berlin Heidelberg, 1998, vol. 1514, pp. 51–65.

[14] V. Shoup, "NTL: A library for doing number theory," available from http://shoup.net/ntl/index.html.