Journal of Mathematical Sciences: Advances and Applications
Volume 25, 2014, Pages 43-56

TWO DIMENSIONAL SUB-DECOMPOSITION METHOD FOR POINT MULTIPLICATION ON ELLIPTIC CURVES

RUMA KAREEM K. AJEENA and HAILIZA KAMARULHAILI

School of Mathematical Sciences
Universiti Sains Malaysia
11800 Penang
Malaysia
e-mail: ruma.usm@gmail.com
hailiza@cs.usm.my

Abstract

In this paper, we developed a method for accelerating multiplication operation on classes of elliptic curve that are efficiently-computable based on the GLV method of Gallant, Lambert and Vanstone that was initially proposed in the year 2001, which uses a fast endomorphisms ψ with minimal polynomial $X^2 + rX + s$ to compute the multiple kP of a point P of order n lying on an elliptic curve. In the GLV method, the value k is decomposed into the values k_1 and k_2. Both values are expected to be bounded by $\pm \sqrt{n}$. However, when both values, or one of the values is not within the range, then, in the GLV method, one needs to choose a new k and obtain a new generator to generate a new decomposition values that fall within the range. Even so, this new chosen k does not guarantee the decomposition values would fall within the range. Finding k with decomposition values that fall within the range will usually acquires big loops and more computation complexity. In this work, we propose a new method called the integer sub-decomposition (ISD) to complement the GLV method and

2010 Mathematics Subject Classification: 06-xx, 18B35, 06Bxx, 03G10.

Keywords and phrases: elliptic curve cryptosystem, scalar multiplication, integer sub-decomposition, efficiently computable endomorphisms.

Received November 30, 2013

to overcome this problem. In addition, this paper highlighted the computation complexity of ISD method that is obtained by computing the cost of elliptic curve and finite field operations of all algorithms that assemble the whole process. This result is then compared to the GLV method in one cycle operation. Finally, from our experimental results, it appears that the ISD method has helped to increase the successful computation percentage of kP in comparison with the GLV method. This improvement has a direct impact on the scalar multiplication techniques in elliptic curve cryptography, and will promote and help to maintain the implementation of the elliptic curve cryptosystem in the future.

1. Introduction

Elliptic curve cryptography (ECC) was introduced in 1985 and known for its mathematical rigorousness [6, 10]. It combines area from the group theory and algebraic curve. Elliptic curve cryptography rely its hard problem on the difficulty in solving the discrete logarithm problem and it has been an alternative to the RSA cryptography [9, 11, 12]. Another advantage of ECC is that it requires short bandwidth in comparison to the RSA cryptography. For instance, it is known that a 160-bit elliptic curve key can supply us with the same level of security as a 1024-bit RSA key. Therefore, this short length of key makes ECC more appropriate for resource-constrained environment. For instance, smart card, PDA, and high-bandwidth digital content protection (HDCP), etc. have embedded ECC protocols in their security products.

The main computational operation in ECC implementation is the scalar multiplication of points on the curves. It is not only the main computation, but the most time-consuming process, too. Therefore, the operational efficiency of scalar multiplication directly determines the performance of ECC. The accomplishment of ECC over prime fields is studied and improved in this paper through proposition of an efficient new algorithm for faster elliptic curve scalar multiplication that offers incremental percentage of a successful computation of kP. In this work, a new modified algorithm, namely, integer sub-decomposition (ISD) is developed. It is a sub-decomposition of a multiplier k as an elliptic scalar in the multiplication operation kP, where P is a point satisfies the elliptic curve equation E over prime field F_p.

The operation of the sub-decomposition of the integer k is performed based on the Gallant et al. idea that had been presented in crypto 2001 [8] to compute the multiplication kP. Several researchers analyzed the original Gallant et al. method and has found several gaps and weakness points, among them is in their decomposition operation, where their method only caters for the returned values k_1 and k_2 that fall inside the range, that is, $-\sqrt{n} < k_1, k_2 < \sqrt{n}$. For those values k that has the returned values k_1 and k_2, which are lying outside the range $\pm\sqrt{n}$, the GLV method will not work and one has to choose another value of k that has the returned values k_1 and k_2 that fall within the required range. This point is worth mentioning in detail and should be taken into consideration. The referred issue will be handled by the proposed method ISD that will be discussed in the paper. In addition, we presented the comparison between these two methods through the computational cost of each one in one cycle operation.

2. Preliminaries

Definition 1 ([1, 4]). An elliptic curve E over a field K is defined by an equation

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \tag{1}$$

where $a_1, a_2, a_3, a_4, a_6 \in K$ and $D_E \neq 0$, where D_E is the discriminant of E.

Definition 2 ([1]). A Weierstrass equation defined over K in Equation (1) can be simplified considerably by applying admissible changes of variables. If the $Char(K) \neq 2$ or 3, then the admissible change of variables is

$$(x, y) \to \left(\frac{x - 3a_1^2 - 12a_2}{36}, \frac{y - 3a_1x}{216} - \frac{a_1^3 + 4a_1a_2 - 12a_3}{24}\right),$$

transforms E to the curve

$$E' : y^2 = x^3 + ax + b, \tag{2}$$

where $a, b \in K$. The discriminant of this curve is $D_E = -16(4a^3 + 27b^2)$. If the elliptic curve E' defined over prime field F_p, then Equation (2) is expressed as follows:

$$E' : y^2 = x^3 + ax + b \pmod{p}, \tag{3}$$

where $a, b \in F_p$. The curve E' is said to be non-singular if it has no double zeroes, which means the discriminant $D_E = -16(4a^3 + 27b^2) \neq 0 (\text{mod } p)$.

Definition 3 ([1, 4])**.** Let an elliptic curve be defined as $E : y^2 = x^3 + ax + b \pmod{p}$ over the finite field with $Char(K) \neq 2, 3$. Then, the following arithmetic properties of E should be considered:

1. Identity. $P + \infty = \infty + P = P$ for all $P \in E(K)$.

2. Negative. If $P = (x, y) \in E(K)$, then $(x, y) + (x, -y) = \infty$. The point $(x, -y)$ is denoted by $-P$ and is called the negative of P, note that $-P$ is indeed a point in $E(K)$. Also, $-\infty = \infty$.

3. Point addition. Let $P = (x_1, y_1) \in E(K)$ and $Q = (x_2, y_2) \in E(K)$, where $P \neq \pm Q$. Then $P + Q = (x_3, y_3)$, where

3.1. If $x_1 \neq x_2$, then

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2,$$

and

$$y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1.$$

3.2. If $x_1 = x_2$ but $y_1 \neq y_2$, then $P + Q = \infty$.

4. Point doubling. Let $P = (x_1, y_1) \in E(K)$, where

4.1. If $P = Q$ and $y_1 \neq 0$. Then $2P = (x_3, y_3)$,

where

$$x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1,$$

and

$$y_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3) - y_1.$$

4.2. If $P = Q$ and $y_1 = 0$, then $P + Q = \infty$.

Definition 4 ([6, 10])**.** Assume that E is an elliptic curve defined over the finite field F_p. The point at infinity is denoted by O_E. The set of F_p-rational points on E forms the group $E(F_p)$. A rational map $\psi : E \to E$ satisfies $\psi(O_E) = O_E$, which is called an endomorphism of E. The endomorphism ψ is defined over F_q, where $q = p^n$ if the rational map is defined over F_q. Thus, for any $n \geq 1$, ψ is a group homomorphism of $E(F_p)$ and $E(F_q)$.

Definition 5 ([1])**.** The endomorphism of elliptic curve E defined over F_q is the m-multiplication map $[m] : E \to E$ defined by

$$P \to mP, \tag{4}$$

for each $m \in Z$. The negation map $[-1] : E \to E$ defined by $P \to -P$ is a special case from m-multiplication map.

Definition 6. A GLV generator is a set $\{\nu_1, \nu_2\}$ of two linearly independent vectors ν_1 and ν_2 in the kernel of the homomorphism

$$T : Z \times Z \to Z / n \text{ defined by } (i, j) \to (i + j\lambda)(\text{mod } n). \tag{5}$$

It is called so if each component of ν_1 and ν_2 is bounded by $\sqrt{n}$ [2].

3. The Gallant-Lambert-Vanstone (GLV) Decomposition Method

In this section, we summarized the details of GLV computation method obtained in [2, 5, 7, 8]. Supposedly E is an elliptic curve defined over a finite field F_q and P is a point on E has a large prime order n such that $\# E(F_q) = n \leq 4.$ Let ψ be a non-trivial endomorphism of E defined over F_q and $X^2 + rX + s$ its characteristic polynomial. According to Hasse bound, as long as, n is large, then $\psi(P) = \lambda P$ for some $\lambda \in [1, n-1],$ λ as a root of $X^2 + rX + s$ modulo n. The GLV algorithm decomposes k as

$$k \equiv k_1 + k_2\lambda (\bmod n), \tag{6}$$

where $k_i = C\sqrt{n}$ for $i = 1, 2.$ In the GLV implementation C is regarded as 1.

Now, suppose that T refers to the homomorphism defined in Equation (5). We are aiming to find a small vector u such that $T(u) = k.$ As $T((k, 0)) = k,$ the problem is reduced to finding two linearly independent vectors ν_1 and ν_2 of small length $O(\sqrt{n})$ such that $T(\nu_1) = T(\nu_2) = 0$ and to decompose $(k, 0)$ in this basis with coefficients in Q and then rounding off $(k, 0)$ to the nearest vector $\nu,$ which is a linear combination of ν_1 and ν_2 with coefficient in Z. Eventually, u is a vector selected as $u = (k, 0) - \nu.$ The problem of finding ν_1 and ν_2 is solved in [8] using the extended Euclidean algorithm. Finally, the GLV computation of kP is more efficiently in comparison with the previous existing methods by calculating first $\psi(P),$ decomposing in Equation (6) with $\max(k_1, k_2) = O(\sqrt{n}),$ and using elliptic Straus-Shamir multiplication to compute

$$kP = k_1 P + k_2 \psi(P), \tag{7}$$

as defined in [3].

4. The Computation Complexity of GLV Method

The computational cost of GLV method is computed based on the computation cost of all algorithms employed in this method. The computational complexity can be computed through two types of operations that include elliptic curve operations, namely, A denotes elliptic curve addition point and D denotes elliptic curve doubling point on finite field operations; and I is a field inversion, M is a field multiplication, and S is a field squaring. The GLV approach depends on computing kP, in four sub-algorithms that are discussed with their costs as follows:

(1) The extended Euclidean algorithm to find two linearly independent vectors ν_1 and ν_2 that form the GLV generator that is $\{\nu_1, \nu_2\}$. The computational cost of this algorithm is given by

$$C_{Euclidean} = ZI + 3ZM, \tag{8}$$

where Z is the dimension of the returned vectors r and t as the output from the extended Euclidean algorithm.

(2) The balanced length two-representation of multiplier k to compute k_1 and k_2 as shown in Equation (6). The computational cost of this algorithm is

$$C_{Balanced} = 4S + 6M + 2I. \tag{9}$$

(3) Compute the efficient computable endomorphism $\psi(P) = \lambda P$ that acts on the subgroup $\langle P \rangle$ as a multiplication by λ for some $\lambda \in [1, n-1]$. The computational cost to compute it, is

$$C_{\psi} = \lambda D = \lambda(I + 2S + 2M), \tag{10}$$

where for all point doubling $D = I + 2S + 2M$.

(4) Computing width- $wNAF$ of k_1 and k_2. The cost of this algorithm is

$$\left[D + \sum_{j=1}^{2} \frac{1}{w_j - 1} A\right] \frac{l}{2}, \tag{11}$$

where w_j is the width windows and l is the bit-length of the $\max\{l_1, l_2\}$.

(5) Computing the $k_1 P + k_2 \psi(P)$ from left-to-right steps (8)-(9) in an algorithm (3.77) in [1]. These steps have computational cost

$$\left|\{j : w_j > 2\}\right| D + \sum_{j=1}^{2} (2^{w_j - 2} - 1) A. \tag{12}$$

Since the computation of iP_j, $j = 1, 2$ is performed as a pre-computation stage, so the term is not affected by the running time.

(6) Finally, the computational cost of the GLV method in one cycle is as follows:

$$C_{GLV\ Method} = C_{Euclidean} + C_{Balanced} + C_{\psi} + C_{w\text{-}NAF} + \sum_{j=1}^{2} (2^{w_j - 2} - 1) A. \tag{13}$$

Take note that if the decomposition values do not fall within the required range, then choosing a new value of k will require repeating the whole loop until both the decomposition values fall within the range. To anticipate this situation, we proposed to further sub-decompose the decomposition value/s. The following section will give the details of our proposed method.

5. The Proposed Sub-Decomposition Method for Computing *kP*

The idea of GLV method is the main source and reference on which the integer sub-decomposition (ISD) method depends on for obtaining a speedier scalar multiplication on an ordinary elliptic curve E defined in Equation (3). Concerning this method, its main idea is to sub-decompose the values k_1 and k_2 when both values or one of them is not bounded by $\pm\sqrt{n}$ and it is possible to apply it when both k's values are bounded by $\pm\sqrt{n}$.

The sub-decomposition from Equation (6) gives us the following equations:

$$k_1 = k_{11} + k_{12}\lambda_1 (\text{mod } n) \quad \text{and} \quad k_2 = k_{21} + k_{22}\lambda_2 (\text{mod } n). \tag{14}$$

In order to accomplish that, one should firstly find the GLV generator $\{\nu_1, \nu_2\}$ by using the GLV generator algorithm in [8], for a given n and λ, where n is a large prime order of elliptic curve point P and λ is a root of characteristic polynomial of endomorphism ψ of E. As a result, $k \in [1, n-1]$ will be decomposed into k_1 and k_2. The decomposition is performed by using the balanced length-two representation of a multiplier k algorithm in [6]. In this work, we have modified the existing algorithm to generate the ISD generators $\{\nu_3, \nu_4\}$ and $\{\nu_5, \nu_6\}$ such that each component of $\nu_3, \nu_4, \nu_5,$ and ν_6 is bounded by $\sqrt{n}$, and for each vector, the components are relatively prime to each other. These generators are computed by solving the closest vector problem in a lattice and it involved using an extended Euclidean algorithm. The values k_1 and k_2 are decomposed once again into the integers $k_{11}, k_{12}, k_{21},$ and k_{22} which means, that the sub-decomposition of k as follows:

$$k = k_{11} + k_{12}\lambda_1 + k_{21} + k_{22}\lambda_2 (\text{mod } n), \tag{15}$$

with $-\sqrt{n} < k_{11}, k_{12}, k_{21}, k_{22} < \sqrt{n}$ from any ISD generators $\{\nu_3, \nu_4\}$ and $\{\nu_5, \nu_6\}$. Finally, we compute the scalar multiplication kP as follows:

$$kP = k_{11}P + k_{12}[\lambda_1]P + k_{21}P + k_{22}[\lambda_2]P = k_{11}P + k_{12}\psi_1(P) + k_{21}P + k_{22}\psi_2(P). \tag{16}$$

The formula in Equation (16) demonstrates our modification, which includes computing two endomorphisms $\psi_1(P) = [\lambda_1]P$ and $\psi_2(P) = [\lambda_2]P$, where $P \in E(F_q)$, $\lambda_1, \lambda_2 \in [1, n-1]$, and $\lambda_1 \neq \pm\lambda_2$.

6. The Computation Complexity of ISD Method

The computation cost of ISD method depends on the cost of computations of all the algorithms used in this method. We summarize the main points in the following steps:

(1) We apply the extended Euclidean algorithm to find the first generator $\{\nu_1, \nu_2\}$, where ν_1 and ν_2 are linearly independent vectors in kernel T and they are considered as integer lattice points in two dimensions. The computational cost of this algorithm is given in Equation (8).

(2) Thereafter, use the balanced length-two representation a multiplier algorithm to find the decomposition k into k_1 and k_2. The computational cost of this algorithm is shown in Equation (9).

(3) Then, use the extended Euclidean algorithm once again to find the second and the third generators (ISD generators) $\{\nu_3, \nu_4\}$ and $\{\nu_5, \nu_6\}$, where $\nu_3, \nu_4, \nu_5,$ and ν_6 are linearly independent vectors in kernel T and they are also considered as integer lattice points in two dimensions. The computational cost of ISD generators is

$$C_{ISD\ generators} = (Z_1 I + 3Z_1 M) + (Z_2 I + 3Z_2 M). \tag{17}$$

(4) The complexity cost to find the sub-decomposing integers k_{11}, k_{12}, k_{21}, and k_{22} is

$$C_{Sub\text{-}Decomposing\ Integers} = 2(4S + 6M + 2I). \tag{18}$$

(5) Calculate efficiently computable endomorphisms $\psi_1(P) = [\lambda_1]P$ and $\psi_2(P) = [\lambda_2]P$. The cost of two endomorphisms is

$$C_{\psi 1 \text{ and } \psi 2} = (\lambda_1 + \lambda_2)(I + 2M + 2S). \tag{19}$$

(6) Compute width *w-NAF* of positive integers k_{11}, k_{12}, k_{21}, and k_{22} as NAF_{wj} and NAF_{ws}, where $j = 1, 2$ and $s = 1, 3$. The cost of this algorithm is

$$C_{NAF_{wj}+NAF_{ws}} = \left[D + \sum_{j=1,2} \frac{1}{w_j + 1} A\right]\frac{l}{2} + \left[D + \sum_{s=1,3} \frac{1}{w_s + 1} A\right]\frac{l}{2}.$$

But, when we using the simultaneous *w-NAF* computation, the computational cost is reduced to

$$C_{NAF_{wj}+NAF_{ws}} = \left[D + \sum_{j=1,\,2} \frac{1}{w_j + 1} A\right]\frac{l}{2}. \tag{20}$$

(7) To compute the scalar multiplication *kP* defined in Equation (16), we employ the Straus-Shamir trick for simultaneous point multiplication, the computational cost is reduced to

$$2\sum_{j=1}^{2}(2^{w_j-2} - 1)A. \tag{21}$$

7. Experimental Results

This section discussed the implementation results concerning the two methods, GLV and ISD. The experiments were done on several values of different digit of n and implemented by using the elliptic curve $E : y^2 = x^3 + ax + b$ over F_p with the base point P that has prime order

n. The complexity computation were based on 3 operations, multiplication, inversion, and subtraction. Several different values of n were considered, ranging from a small digit to a bigger digit. For each case, the computational cost was computed and it appears that for every value of n, the computational cost for GLV method was much higher compared to the ISD method. The minimum cost computed in the GLV method is $1485I + 3168M + 2574S$. Adding to this, the computational cost for the last loop which includes the values of k_1 and k_2 that lie inside the range gives us the total of GLV computational cost as follows:

$$C_{GLV} = 1538I + 3281M + 2663S. \tag{22}$$

In comparison with the computation complexity of the ISD method, by skipping the loops for finding a new value of k, the sub-decomposition requires less computations. The computational cost of the ISD method carried out on the same elliptic curve $E : y^2 = x^3 + ax + b$ over F_p with the base point P with exactly the same minimum prime order n is as follows:

$$C_{ISD} = 113I + 246M + 187S. \tag{23}$$

It also appeared that for each value of n, the percentage of successful computation of kP is much higher in the ISD method and it is proportionate to the value of n, in contrast with the GLV method, where the percentage decreased as the values of n increased.

In particular, calculating the percentage for the minimum and the maximum values of n, the GLV successful computation percentage are as follows:

$$GLV_{Percentage\ for\ min\ value\ n} = 4.50\% \text{ and } GLV_{Percentage\ for\ max\ value\ n} = 0.17\%. \tag{24}$$

Comparing the above, with the following ISD successful computation percentage, with exactly the same minimum and maximum values of n, which is as follows:

$$ISD_{Percentage\ for\ min\ value\ n} = 39.4\% \text{ and } ISD_{Percentage\ for\ max\ value\ n} = 49.66\%, \tag{25}$$

apparently, the percentage of success for computing multiplication operation kP in the ISD method has greater percentage than the GLV method. This is another advantage that makes the ISD method more efficient in application to compute kP.

8. Conclusion

The sub-decomposition method has drawn a new way to compute scalar multiplication kP. This method based its computations on the idea of GLV method. The application to this new procedure known as ISD (integer sub-decomposition) method came as a result from the sub-decomposition to values lying outside the range $\pm\sqrt{n}$, which were considered as an important issue as to increase the percentage of the success for the scalar multiplication kP. This paper presented a comparison on successful computation of kP and the complexity computations for both the GLV and the ISD methods in one cycle operation, which through it we observed that the computational cost of the ISD method in the Equation (23) is less than the computational cost of the GLV method in the Equation (22). In addition to this complexity computation, different experiments based on different values of n were carried out. According to Equations (24) and (25), it shows that the success percentage of ISD method is higher than that of GLV method. Moreover, one can deduce that as often as the value n is increased, the selected value of k from the interval $[1, n-1]$ will be increased and finally, the success percentage of ISD start to increase and on the contrary, the success percentage of GLV method begin to decrease. Therefore, the ISD approach increased the efficiency of scalar multiplication computation and which, in turn, increase the speed of computation in the elliptic curve cryptography.

Acknowledgement

This research work is supported by the Universiti Sains Malaysia Fundamental Research Grant, FRGS, 203/PMATHS/6711320.

References

[1] D. Hankerson, A. J. Menezes and S. Vanstone, Guide to Elliptic Curve Cryptography, Springer, 2004.

[2] D. Kim and S. Lim, Integer decomposition for fast scalar multiplication on elliptic curves, In Selected Areas in Cryptography (2003), 13-20.

[3] D. M. Gordon, A survey of fast exponentiation methods, Journal of Algorithms 27(1) (1998), 129-146.

[4] L. C. Washington, Elliptic Curves: Number Theory and Cryptography, 50: Chapman & Hall/CRC, 2008.

[5] M. Ciet, J.-J. Quisquater and F. Sica, Preventing differential analysis in GLV elliptic curve scalar multiplication, Cryptographic Hardware and Embedded Systems-CHES 2002 (2003), 1-13.

[6] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation 48 (1987), 203-209.

[7] P. Longa and F. Sica, Four-dimensional Gallant-Lambert-Vanstone scalar multiplication, In Advances in Cryptology-ASIACRYPT 2012, Springer, (2012), 718-739.

[8] R. Gallant, R. Lambert and S. Vanstone, Faster point multiplication on elliptic curves with efficient endomorphisms, In Advances in Cryptology-CRYPTO 2001 (2001), 190-200.

[9] R. Mahdavi and A. Saiadian, Efficient scalar multiplications for elliptic curve cryptosystems using mixed coordinates strategy and direct computations, In Cryptology and Network Security, Springer, (2010), 184-198.

[10] V. Miller, Use of elliptic curves in cryptography, In Advances in Cryptology-CRYPTO'85 Proceedings (1986), 417-426.

[11] Y. Hao, S. Ma, G. Chen, X. Zhang, H. Chen and W. Zeng, Optimization algorithm for scalar multiplication in the elliptic curve cryptography over prime field, In Advanced Intelligent Computing Theories and Applications, Springer, (2008), 904-911.

[12] Y. Sakai and K. Sakurai, Efficient scalar multiplications on elliptic curves without repeated doublings and their practical performance, In Information Security and Privacy (2000), 59-73.

■