Malaysian Journal of Mathematical Sciences 9(S) June: 115-137 (2015)
Special Issue: The 4th International Cryptology and Information Security Conference 2014 (Cryptology 2014)

MALAYSIAN JOURNAL OF MATHEMATICAL SCIENCES

Journal homepage: http://einspem.upm.edu.my/journal

Accelerating Integer Sub-Decomposition for Elliptic Scalar Multiplication using the Generalized w_{j-}NAF Expansions Method

[*] **Ruma Kareem K. Ajeena and Hailiza Kamarulhaili**

School of Mathematics, Babylon University, Babil city, Iraq

School of Mathematical Sciences, Universiti Sains Malaysia,
11800 Minden, Penang, Malaysia

E-mail: ruma.usm@gmail.com

*Corresponding author

ABSTRACT

In this paper, wNAF expansion method is used to compute a scalar multiplication on classes of elliptic curve over a prime field that have efficiently-computable endomorphisms. This scalar multiplication is called integer sub-decomposition (ISD) method, which is based on the GLV method of Gallant, Lambert and Vanstone that was initially proposed in the year 2001. In this work the ISD implementation uses speed parallel computation of endomorphisms ψ_i for i = 1,2 to compute the multiple kP of a point P of order n lying on an elliptic curve. The decomposition of a scalar k according to GLV method produces two integers k_1 and k_2 lie inside the range of $\pm\sqrt{n}$. This decomposition also gives significant number of integers k_1 and k_2 lie outside the given range of $\pm\sqrt{n}$. These outliers are not considered in the GLV method. Therefore, the ISD approach is proposed to bridge the gap and to complement the GLV method. Besides that, the ISD method helps increase the percentage of successful computation of kP. In this paper, the main idea is to present the parallel computation of ISD elliptic scalar multiplication which is defined by the following decompositions.

$$kP= k_{11}P+ k_{12}\psi_1(P)+ k_{21}P+ k_{22}\psi_2(P), \text{ with } |k_{11}|, |k_{12}|, k_{21}|, |k_{22}| < \sqrt{n}.$$

This computation employs two models using the interleaving methods based on parallel computation of the generalized $w_{\text{j-NAF}}$ expansions for j=1,2,3,4. It is known that the parallel processing on two proposed interleaving methods produce more computation speed in comparison with the computations that were performed individually.

Keywords: Elliptic curve cryptography, Scalar multiplication, Parallel interleaving method, Parallel w_{j}-NAF expansions, Efficiently-computable endomorphisms, Integer sub-decomposition.

1. INTRODUCTION

Elliptic curve cryptography was proposed by Miller, 1986 and Neil Koblitz, 1987 in the year 1985, and it has attracted increasing attention in these recent years because of their shorter key length requirement when compared to other public-key cryptosystems like RSA. Elliptic curve cryptography (*ECC*) is derived from the hardness of the discrete logarithm problem over the additive group of points on an elliptic curve defined on finite fields. Among the benefits of *ECC* are shorter key length, higher speed and lower power consumption. These advantages are useful for some devices like mobile and wireless which, typically, have limited computational resources and bandwidth.

Scalar multiplication, in general, represented by kP is considered as the central time-consuming operation in *ECC*. In order to compute this operation, it is necessary to perform iterative addition (*ECADD*) and doubling (*ECDBL*) of points, which we referred to as *ECC* point operations, and their efficient performance is essential to speed up the computation of scalar multiplication Hao, et al., 2008.

Elliptic curves have a well-known facts and distinct theoretical aspects for the algebraic structures and also the endomorphism applications which can be applied to improve performance fast in elliptic curve scalar multiplication. The extension idea of using Frobenius endomorphism $\psi \in End(E)$ on elliptic curves of arbitrary characteristic $p \geq 3$ splits a large computation into a sequence of cheaper ones so that the overall computational cost is lowered Hankerson et al., 2004. Such a technique, which contrary to previous ones, also applied to curves defined over large prime fields, was used, recently, by Gallant et al., 2001. Their method uses an efficiently computable endomorphism $\psi \in End(E)$ to rewrite kP as

$$kP = k_1P + k_2\psi(P), \text{ with } \max\{|k_1|,|k_2|\} = O\left(\sqrt{n}\right).$$

In Gallant, Lambert and Vanstone (GLV) method, the value k is decomposed into the values k_1 and k_2 with the condition that both values are bounded by $\pm\sqrt{n}.$ There are some failing points in GLV method, among them, the main weakness point is, it does not determine the case when the values of k_1 and k_2 are not within the range $\pm\sqrt{n}.$ So, the GLV method will not work with this case. As result, we have proposed new method is called Integer Sub-Decomposition (ISD) (Ajeena and Kamarulhaili, 2013; Ajeena and Kamarulhaili, 2014a; 2014b) to increase the percentage of a successful computation of kP. The basic idea of ISD method is the sub-decomposition of the values k_1 and k_2 into the values k_{11}, k_{12}, k_{21} and k_{22}. The sub-decomposition from

$$k = k_1 + k_2\lambda \,(\mathrm{mod}\, n) \tag{1}$$

is elucidated as the following:

$$k_1 = k_{11} + k_{12}\lambda_1 \text{ (mod } n) \text{ and } k_2 = k_{21} + k_{22}\lambda_2 \text{ (mod } n). \tag{2}$$

The meaningful role of the method lies in the definition of the group homomorphism (the ISD reduction map)

$$\begin{aligned} &T : Z \times Z \rightarrow Z / n \\ &(i, j) \rightarrow i + \lambda_m j \,(\mathrm{mod}\, n),\ m = 1, 2. \end{aligned} \tag{3}$$

In particular, we compute the sub-decomposition as follows:

$$\begin{aligned} kP &= k_{11}P + k_{12}\lambda_1 P + k_{21}P + k_{22}\lambda_2 P \\ &= k_{11}P + k_{12}(\lambda_1 P) + k_{21}P + k_{22}(\lambda_2 P) \\ &= k_{11}P + k_{12}\psi_1(P) + k_{21}P + k_{22}\psi_2(P). \end{aligned} \tag{4}$$

The computation of kP that is defined in equation (4) can be carried out through two proposed methods which uses the computation of the interleaving based on the generalized w_j-NAF expansions in parallel. Computing interleavings on the first proposed method is performed in two parallel lines (threads) . The sum of two outputs which form as two elliptic curve points gives the final result of the ISD elliptic scalar multiplication kP. Whereas, the computation with the second proposed method is done in one parallel line (thread) to find the final result of kP directly.

2. MATHEMATICAL PRELIMINARIES

In this section, some mathematical basics such as definitions, group laws of points on elliptic curves and related known results are shown to give more insight of our work.

2.1. Elliptic Curve over Prime Field

Definition 1. (Hankerson et al., 2004 and Washington, 2008). An elliptic curve E over a field K is defined by an equation

$$E: y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{5}$$

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. (Hankerson et al., 2004). A Weierstrass equation defined over K in equation (5) can be simplified considerably by applying admissible changes of variables. If $Char(K) \neq 2$ 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

$$E': y^2 = x^3 + ax + b, \tag{6}$$

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 (6) is expressed as

$$E': y^2 = x^3 + ax + b (\text{mod} \;\; p), \tag{7}$$

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. (Hankerson et al., 2004 and Washington, 2008). Let an elliptic curve be defined as $E: y^2 = x^3 + ax + b (\text{mod} \;\; p)$ 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. Negatives. 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. (Gallant et al., 2001; and Hankerson et al., 2004). 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 $\psi:O_E\to O_E$, 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 $n \geq 1, \psi$ is a group homomorphism of $E(F_p)$ and $E(F_q)$.

Definition 5. (Hankerson et al., 2004). The endomorphism of elliptic curve *E* defined over F_q is the *m*- multiplication map $[m]: E \to E$ defined by

$$P \to mP \tag{8}$$

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.

Lemma 6. Let *E* be an elliptic curve over prime field F_p and let P be a point lies on E has large prime order *n*. Assume that $\psi(P)$ is a non trivial endomorphism of E. Then $\psi(P) = \lambda P$, where λ is a root of its characteristic polynomial.

Definition 7. (Kim and Lim, 2003). ISD generators are two sets $\{v_3, v_4\}$ and $\{v_5, v_6\}$ of the linearly independent vectors v_3, v_4 and v_5, v_6 in the kernel of the homomorphism

$$T : Z \times Z \to Z/n \text{ defined by } (i, j) \to (i + j\lambda_m)(\text{mod } n), \tag{9}$$

where m=1,2. It is called ISD generators if each component of v_3, v_4 and v_5, v_6 are bounded by √*n*.

Definition 8. (Karypis et al., 1994; Jones, 1999; Barney et al., 2010). Parallel computing is a mode of computation which carries out many of repeated calculations simultaneously, or it is operating on the basic dividing the large problems into smaller ones to solve them in parallel. On the other words, parallel computing is the employ of two or more processors in combination to solve computational problem such that this problem can be broken into discrete portions, and can be solved concurrently.

3. WINDOW METHODS

With some extra memory, the implementation to represent the scalar *k* can be done using the window method. The method takes an input of *w* digits of k at the same time. Therefore, the executing time to represent k can be decreased.

Definition 9. (Hankerson et al., 2004). Suppose $w \geq 2$ is a positive integer. A width-*w* NAF of a positive integer *k* is an expression

$$z = \sum_{i=0}^{l-1} G_i 2^i \qquad (10)$$

where each nonzero coefficient G_i is odd, $|G_i| < 2^{w-1}$, $G_{l-1} \neq 0$, and at most one of any w consecutive digits is nonzero. The length of the width-*w* NAF is l.

Theorem 10. (Properties of width-*w* NAFs). (Hankerson et al., 2004)
Let *k* be a positive integer, then

(i) k has a unique width-*w* NAF denoted $NAF_w(k)$.

(ii) $NAF_2(k) = NAF(k)$.

(iii) The length of $NAF_w(k)$ is at most one more than the length of the binary representation of k.

(iv) The average density of nonzero digits among all width-*w* NAFs of length *l* is approximately $1/(w+1)$.

3.1. Interleaving Method

To increase efficiency, one needs to speed up the computation of $kP + lQ$ that was used in some parts of elliptic curve cryptosystems such as in digital signature scheme. This acceleration can be achieved through using a simultaneous multiple point multiplication that is also named Shamir trick. The simultaneous method depends on combinations of the points $iP + jQ, i, j = 0,1,2,3,\ldots,$ that was pre-computed in the pre-computation stage. When the pre-computed value has only a single point like iP_j then the simultaneous method is called interleaving.

The interleaving, in computation $\sum z_j P_j$ for points P_j and integers $z_j, j = 1,2,$ permits to apply various methods for each $z_j P_j$ such that the doubling step can be carried out simultaneously. For instance, using width wNAF method with different width values *w*. The computational cost of doubling can be determined through the determination of the maximum number of required doublings for each computing $z_j P_j$. The interleaving method to compute

$$\sum_{j=1,2} z_j P_j \tag{11}$$

is given by algorithm (3.51) of Hankerson et al., 2004. The representation of integers z_j that written in equation (11) can be done by using w_jNAF expansions that is illustrated in algorithm (3.35) of Hankerson et al., 2004. In algorithm (3.51), computing the points iP_j for odd $i < 2^{w_j - 1}$ was performed in precomputation stage. The processing through expansions NAF $w_j(z_j)$ can be carried out simultaneously from left to right with a single doubling of the accumulator at each phase Hankerson et al., 2004.

4. PARALLEL COMPUTATION OF W$_j$-NAF EXPANSIONS AND EFFICIENTLY COMPUTABLE ENDOMORPHISMS

In this section, parallel computation of $w_{j\text{-}}$NAF expansion and efficiently computable endomorphisms are discussed. Algorithms for both models are shown and proposed here.

4.1. Proposed Model of The Parallel Computation of *w*NAF Expansions in Two Parallel Lines

In this work, it is possible to generalize the computation of wNAF expansion that was introduced in Definition (9) and Theorem (10) and enhanced the implementations by Algorithm (3.51) in Hankerson et al., 2004. The generalization produces through finding the representation of $w_{j\text{-}}$NAF expansions for four integers simultaneously through the parallel computation which consists of two parallel lines. Each parallel line works through two parallel sub-lines. First parallel sub-line takes (z_1, w_1) as an input to compute w_1NAF(k_{11}) expansion, second parallel sub-line takes input (z_2, w_2) to output w_2NAF(k_{12}) expansion. Whereas, on the second parallel line, the first parallel sub-line takes (z_3, w_3) as an input to obtain w_3 NAF(k_{21}) expansion and second parallel sub-line takes input (z_4, w_4) and resulted in w_4 NAF(k_{22}) expansion.

In the context of the Definition (9) that presented to explain wNAF expansion, it can be generalized for parallel computation concept to represent more than one of integers and compute the $w_{j\text{-}}$NAF expansions of them as follows:

Definition 11. Let $w_j > 2,\ j = 1,2,3,4$ be positive integers. The width w_j *NAFs* of positive integers z_j are expansions

$$z_j = \sum_{i=0}^{l_j - 1} G_{i,j} 2^i, \text{ with } j = 1,2,3,4. \tag{12}$$

such that each nonzero coefficient $G_{i,j}$ is odd and satisfies $\left|G_{i,j}\right| < 2^{w_j - 1}$. The leftmost significant bit $G_{l_j - 1, j} \neq 0$. For each expansion, any w_j consecutive bits, at most one of them is nonzero. The length of each w_j NAF expansion is l_j for $j = 1,2,3,4$. The implementation of algorithm (1) gives results of the generalized w_j NAF expansions in two parallel lines. Figure (1) depicts the parallel computation of w_j-NAF expansions on two parallel lines.

4.2. Parallel Computation of the Width- w_j-NAFs of Four Positive Integers in Two Parallel Lines

The following algorithm 1 is the algorithm for parallel computation of width-wNAFs with four positive integers in two parallel lines.

Algorithm 1:
Input: Window width w_j, positive integer z_j, j=1,2,3,4.

Output: The $w_j NAF$ expansions of positive integers $z_j, j = 1,2,3,4.$

```
Computation:
$i \leftarrow 0$
First parallel Line:
While $(z_j \geq 1, j = 1,2)$ do
    If ( $z_j$ is odd ) then
        $G_{i,j} \leftarrow z_j \bmod 2^{w_j}, z_j \leftarrow z_j - G_{i,j}$
    Else $G_{i,j} \leftarrow 0$
    Endif
    $z_j \leftarrow z_j/2,\ i \leftarrow i+1.$
Endwhile
Second parallel Line:
While $(z_j \geq 1, j = 3,4)$ do
```

```
13. If ( $z_j$ is odd ) then
14.   $G_{i,j} \leftarrow z_j \bmod 2^w,\ z_j \leftarrow z_j - G_{i,j}$
15. Else
16.   $G_{i,j} \leftarrow 0$
17. Endif
18. $z_j \leftarrow z_j/2,\ i \leftarrow i+1.$
19. Endwhile
20. Return: From first parallel line:
    $\{G_{i-1,1}, G_{i-2,1}, \ldots, G_{1,1}, G_{0,1}\}$ and $\{G_{i-1,2}, G_{i-2,2}, \ldots, G_{1,2}, G_{0,2}\}$
21. From second parallel line:
    $\{G_{i-1,3}, G_{i-2,3}, \ldots, G_{1,3}, G_{0,3}\}$ and $\{G_{i-1,4}, G_{i-2,4}, \ldots, G_{1,4}, G_{0,4}\}$
```

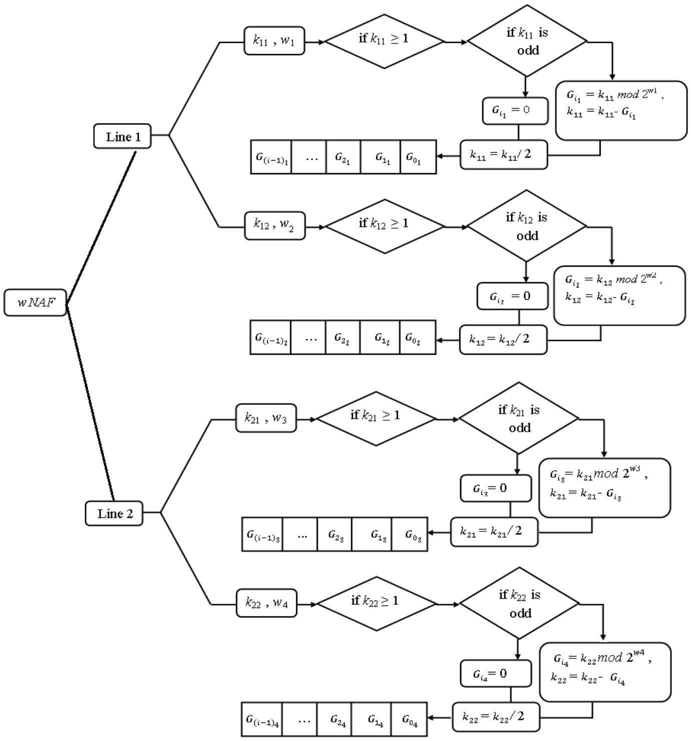


Figure 1: shows the parallel computations of w_{j}-NAF expansions on two lines.

Remark 12. To make the computation of w_{j}-NAF expansions of the integers k_{11}, k_{12}, k_{21} and k_{22} and later on computing the interleavings $k_{11}P + k_{12}\psi_1(P)$ and $k_{21}P + k_{22}\psi_2(P)$ or $k_{11}P + k_{12}\psi_1(P) + k_{21}P + k_{22}\psi_2(P)$ more clearly, it is possible to write z_1, z_2, z_3 and z_4 rather than k_{11}, k_{12}, k_{21} and k_{22} respectively.

4.3. The proposed Parallel Computation of w_{j}-NAF Expansions in One Parallel Line

On the other side, the parallel computations to find w_{j}-NAF expansions can take another proposed model. This model is formed through another type of the parallel designs. It consists of one parallel line contains of four parallel sub-lines. On each parallel sub-line take place processing each value z_j for $j = 1, 2, 3, 4$ that based on the generalization of the original *w*NAF idea which presented in the Definition (11). For instance, the first parallel sub-line takes z_1 and w_1 as inputs and the processing performs to output the result of $w_1 \text{NAF}(z_1)$ expansion. For all others three parallel sub-lines, the processing occurs in the same way through computing same operations in each parallel sub line. All the computations carries out simultaneously to output results. So, such these computations save a lot of execution time. The parallel implementation results can be obtained by applying the following Algorithm (2). Figure (2) illustrates the parallel computation of w_{j}-NAF expansions on one parallel line.

4.4. Parallel Computation of the Width- w_{j}-NAFs Positive Integers in One Parallel Line

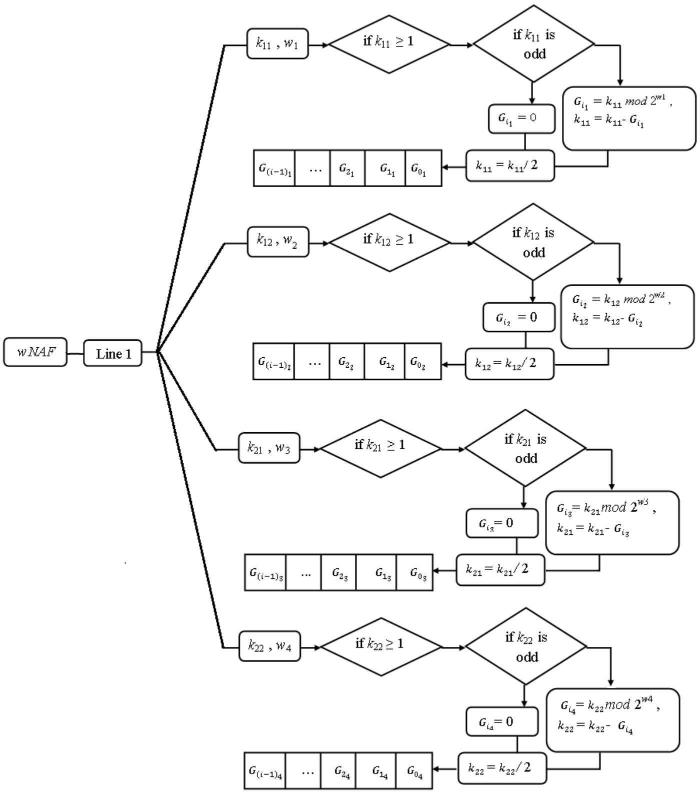
Next is the algorithm of parallel computation of the width- w_{j}-NAFs with positive integers in one line.

Algorithm 2:
Input: Window width w_j, positive integer z_j, $j = 1, 2, 3, 4$.

Output: The $w_j NAF$ expansions of positive integer z_j.

```
Computation:
1.   $i \leftarrow 0$
2.   While $(z_j \geq 1, j = 1, 2, 3, 4)$ do
3.   If ( $z_j$ is odd ) then
4.   $G_{i,j} \leftarrow z_j \bmod 2^{w_j}, z_j \leftarrow z_j - G_{i,j}$
5.   Else
```

```
        $G_{i,j} \leftarrow 0$
    Endif
        $z_j \leftarrow z_j/2,\ i \leftarrow i+1.$
    Endwhile
    Return: On one line: from parallel sub-line1: $\{G_{i-1,4}, G_{i-2,4}, \ldots, G_{1,4}, G_{0,4}\}$,
    parallel sub-line 2: $\{G_{i-1,2}, G_{i-2,2}, \ldots, G_{1,2}, G_{0,2}\}$,
    parallel sub-line 3: $\{G_{i-1,3}, G_{i-2,3}, \ldots, G_{1,31}, G_{0,3}\}$,
    parallel sub-line 4: $\{G_{i-1,4}, G_{i-2,4}, \ldots, G_{1,4}, G_{0,4}\}$
```



$wNAF$
Line 1
k_{11} , w_1
if $k_{11} \geq 1$
if k_{11} is odd
$G_{i_1} = 0$
$G_{i_1} = k_{11}$ mod 2^{w1} , $k_{11} = k_{11}$ - G_{i_1}
$k_{11} = k_{11}/2$
$G_{(i-1)_1}$ … G_{2_1} G_{1_1} G_{0_1}
k_{12} , w_2
if $k_{12} \geq 1$
if k_{12} is odd
$G_{i_2} = 0$
$G_{i_2} = k_{12}$ mod 2^{w2} , $k_{12} = k_{12}$ - G_{i_2}
$k_{12} = k_{12}/2$
$G_{(i-1)_2}$ … G_{2_2} G_{1_2} G_{0_2}
k_{21} , w_3
if $k_{21} \geq 1$
if k_{21} is odd
$G_{i_3} = 0$
$G_{i_3} = k_{21}$ mod 2^{w3} , $k_{21} = k_{21}$ - G_{i_3}
$k_{21} = k_{21}/2$
$G_{(i-1)_3}$ … G_{2_3} G_{1_3} G_{0_3}
k_{22} , w_4
if $k_{22} \geq 1$
if k_{22} is odd
$G_{i_4} = 0$
$G_{i_4} = k_{22}$ mod 2^{w4} , $k_{22} = k_{22}$ - G_{i_4}
$k_{22} = k_{22}/2$
$G_{(i-1)_4}$ … G_{2_4} G_{1_4} G_{0_4}

Figure 2: shows the parallel computations of w_j-NAF expansions on one line.

4.5. Parallel Computing For Efficiently Computable Endomorphisms

Recall the definition of endomorphism (4) and how to compute it though Definition (5) and Lemma (6). Such endomorphism has been computed as a multiplication by λ where $\lambda \in [1, n-1]$. For $\lambda_1, \lambda_2 \in [1, n-1]$, it is possible using Lemma (6) to compute two endomorphisms ψ_1 and ψ_2 such that $\psi_1(P) = \lambda_1 P$ and $\psi_2(P) = \lambda_2 P$, where $\lambda_1 \neq \pm\lambda_2$ and P is a point lies on *E* over prime field F_p. Since the computation of endomorphisms form as multiplication by λ's then, it is easy to use any algorithm that computes the doubling $\lambda_1 P$ and $\lambda_2 P$. The parallel computation of these doublings is more efficient, because it saves (min $(\lambda_1, \lambda_2) - 1)\, I + 2\,(\min(\lambda_1, \lambda_2) - 1)$ $M + (\min(\lambda_1, \lambda_2)\text{-}1)\, S$ from the executing time in comparison with the time that needs for computing two endomorphisms separately, where *I*, *M* and *S* are field operations, inversion, multiplication and squaring respectively.

5. THE PROPOSED INTERLEAVING METHODS TO COMPUTE ISD ELLIPTIC SCALAR MULTIPLICATION

In this section, we proposed two interleaving methods to compute ISD elliptic scalar multiplication kP.

5.1. The Interleaving Method to Compute $k_{11}P+k_{12}\psi_1(P)$ and $k_{21}P+k_{22}\psi_2(P)$

Let $E: y^2 = x^3 + ax + b$ be an elliptic curve defined over a prime field F_p. And let *P* be a point on *E* which has prime order *n*. The curve *E* has two efficiently computable endomorphisms $\psi_1 P$ and $\psi_2 P$ that computed as shown in Definition (5) and Lemma (6). The computation of $k_{11}P + k_{12}\psi_1(P)$ and $k_{21}P + k_{22}\psi_2(P)$ can be carried out through the applying of the conception of interleaving method. But, it is not reasonable to compute these interleavings one by one because this procedure needs further of the executing time. So, it requires to propose new model based on the concept of the parallel computation of these interleavings $k_{11}P + k_{12}\psi_1(P)$ and $k_{21}P + k_{22}\psi_2(P)$ simultaneously.

The basic idea of this model depends on the parallel implementation of the generalized $w_{\text{j-}}$NAF expansions that proposed and implemented by Algorithm (1). This model consists of two parallel lines, each parallel line contains on two parallel sub-lines. On these parallel sub-lines take place the processing of

$w_{j\text{-}}$NAF expansions to represent four sub-scalars k_{11}, k_{12}, k_{21} and k_{22} on the same time.

The parallel computation in two lines of interleavings is performed in two stages. The pre-computation stage and evaluation stage. The pre-computation stage implements to compute the points iP_j for $i \in \{1, 3, \ldots, 2^{w_j - 1} - 1\}$ for $j = 1, 2, 3, 4$. There are two cases of computing iP_j when $j = 1, 3$. Since P_1=P_3=P, so when w_1= w_3 then the computation of iP_1 and iP_3 carries out for one of them. Whereas, if $w_1 \neq w_3$ the computation performs also for one of them that has maximum value between w_1 and w_3. In evaluation stage takes place the computing of w_j NAF for $j = 1, 2$ on the first parallel line and the computing of w_j NAF for $j = 3, 4$ on the second parallel line. Furthermore, on each parallel line, the applying of the interleaving method can be performed. Figure (3) shows the parallel computations of $w_{j\text{-}}$NAF expansions in two lines. The implementation of the proposed model is given by Algorithm (3).

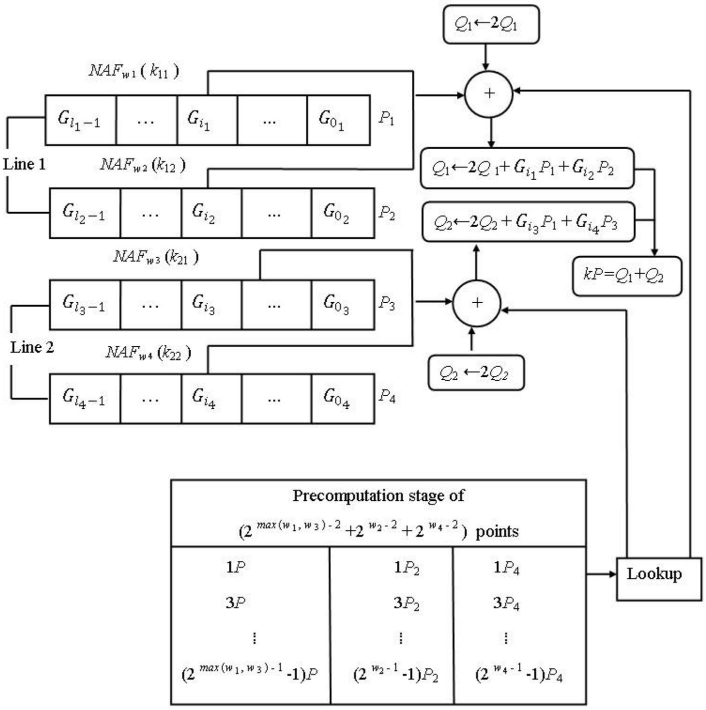

$Q_1 \leftarrow 2Q_1$
$NAF_{w1}(k_{11})$
G_{l_1-1} … G_{i_1} … G_{0_1} P_1
Line 1
$NAF_{w2}(k_{12})$
G_{l_2-1} … G_{i_2} … G_{0_2} P_2
$NAF_{w3}(k_{21})$
G_{l_3-1} … G_{i_3} … G_{0_3} P_3
Line 2
$NAF_{w4}(k_{22})$
G_{l_4-1} … G_{i_4} … G_{0_4} P_4
+
$Q_1 \leftarrow 2Q_1 + G_{i_1}P_1 + G_{i_2}P_2$
$Q_2 \leftarrow 2Q_2 + G_{i_3}P_1 + G_{i_4}P_3$
$kP = Q_1 + Q_2$
+
$Q_2 \leftarrow 2Q_2$
Precomputation stage of
$(2^{max(w_1, w_3)-2} + 2^{w_2-2} + 2^{w_4-2})$ points
$1P$ $1P_2$ $1P_4$
$3P$ $3P_2$ $3P_4$
⋮ ⋮ ⋮
$(2^{max(w_1, w_3)-1}-1)P$ $(2^{w_2-1}-1)P_2$ $(2^{w_4-1}-1)P_4$
Lookup

Figure 3: shows the parallel computation of two interleavings $k_{11}P$+$k_{12}\psi_1(P)$ and $k_{21}P$+$k_{22}\psi_2(P)$.

5.2. Algorithm 3 of the interleaving method based on w_{j}-NAF expansions in two parallel lines.

Input: Integers z_j, widths w_j and points P_j for $j=1,2,3,4$.

Output: An interleavings points $\sum_{j=1,2} z_j P_j$ and $\sum_{j=3,4} z_j P_j$.

```
Precomputation stage:
    Compute $iP_j$ for $i \in \{1,3,...,2^{w_j-1}-1\}$ where $j=1,2,3,4$.
    Computation of the endomorphisms $\psi_1(P)$ and $\psi_2(P)$.
Parallel computation stage:
First parallel line:
    Set $P_1 \leftarrow P, P_2 \leftarrow \psi_1(P)$.
    Run parallel computations width $w_j$-NAF of positive integers Algorithm
    (1) to compute $NAF_{w_j}\left(\left|z_j\right|\right)=\sum_{i=1}^{l_j-1} G_{i,j}2^i$ for $j$ from 1 to 2 do.
    Set $l=\max\{l_j, j=1,2\}$.
    Define $G_{i,j}=0$ for $i$ from $l_j$ to $l$-1, and for $j$ from 1 to 2 do.
    If $(z_j<0)$ then,
        Set $G_{i,j} \leftarrow -G_{i,j}, i=0:l, j=1,2$.
    Else $G_{i,j} \leftarrow G_{i,j}, i=0:l, j=1,2$.
    Endif
    $Q \leftarrow \infty$.
    For $i$ from $l$-1 down to 0 do
        $Q \leftarrow 2Q$.
        For $j$ from 1 to 2 do.
            If $(G_{i,j} \neq 0)$ then,
                If $(G_{i,j} > 0)$ then,
                    $Q \leftarrow Q + G_{i,j}P_j$
                Else $Q \leftarrow Q - \left|G_{i,j}\right| P_j$
                Endif
            Else $Q \leftarrow Q$.
            Endif
    Endfor
Second parallel line:
    Set $P_3 \leftarrow P, P_4 \leftarrow \psi_2(P)$.
```

```
Run parallel computations width-w NAF of positive integers algorithm (2) to compute $NAF_{w_j}\left(\left|z_j\right|\right)=\sum_{i=1}^{l_j-1} G_{i,j}2^i$ for $j$ from 3 to 4 do.
Set $l' = \max\{l_j, j = 3,4\}$.
Define $G_{i,j} = 0)$ for $i$ from $l_j$ to $l'$-1, and for $j$ from 3 to 4 do.
If $(z_j < 0)$ then,
    Set $G_{i,j} \leftarrow -G_{i,j}, i = 0:l', j = 3,4$.
Else $G_{i,j} \leftarrow G_{i,j}, i = 0:l', j = 3,4$.
Endif
$Q \leftarrow \infty$.
For $i$ from $l'$-1 down to 0 do.
    $Q \leftarrow 2Q$.
    For $j$ from 3 to 4 do.
        If $(G_{i,j} \neq 0)$ then,
            If $(G_{i,j} > 0)$ then,
                $Q \leftarrow Q + G_{i,j}P_j$.
            Else
                $Q \leftarrow Q - \left|G_{i,j}\right|P_j$.
            Endif
        Else
            $Q \leftarrow Q$.
        Endif
    Endfor
Endfor
Return $Q$.
```

5.3. Interleaving Method to Compute $k_{11}P+k_{12}\psi_1(P)+k_{21}P+k_{22}\psi_2(P)$

For speeding up computation of elliptic scalar multiplication kP, another model of the interleaving method which consists of four sub-scalar multiplications can be used. The idea is to modify the computation proposed that has been accomplished on two parallel lines to compute interleavings $k_{11}P + k_{12}\psi_1(P)$ and $k_{21}P + k_{22}\psi_2(P)$ into calculation of the interleaving that defined in equation (4) on one line that comprises of four parallel sub-lines. For each term in equation (4), the interleaving permits to use different methods. For instance, using width w_{j}-NAF with various window widths or other methods. The doubling at each term can be carried out simultaneously. So the cost of the doubling is given by the maximum number of doublings required to compute equation (4).

Computation of the interleaving that was defined in equation (4), the processing takes place firstly for all four parallel sub-lines simultaneously to determine the w_{j}-NAF expansions for integers k_{11}, k_{12}, k_{21}, k_{22} through w_j with j=1,2,3,4. Follows these the determinations, the computation of interleaving that also based on the saved points iP_j for odd $i < 2^{w_j - 1}$, j=1,2,3,4 which are obtained from the pre-computation stage. Since P_1=P_3=P, in the pre-computation stage, then sets of the points iP_1 and iP_3 are equal when $w_1 = w_3$, so the pre-computation does for one of them. On the other hand, it is possible determining the maximum number between w_1 and w_3, $\max(w_j)$, j =1,3, so the pre-computation is performed for $i < 2^{\max(w_j)-1}$, j=1,3.

Figure (4) shows the parallel computation of one interleaving $k_{11}P + k_{12}\psi_1(P) + k_{21}P + k_{22}\psi_2(P)$. Algorithm (4) can be used to compute the interleaving that is defined in equation (4). On this algorithm, the computation of the steps (15-28) carries out from the left to the right and at each phase, there is one doubling operation 2Q of the accumulator.

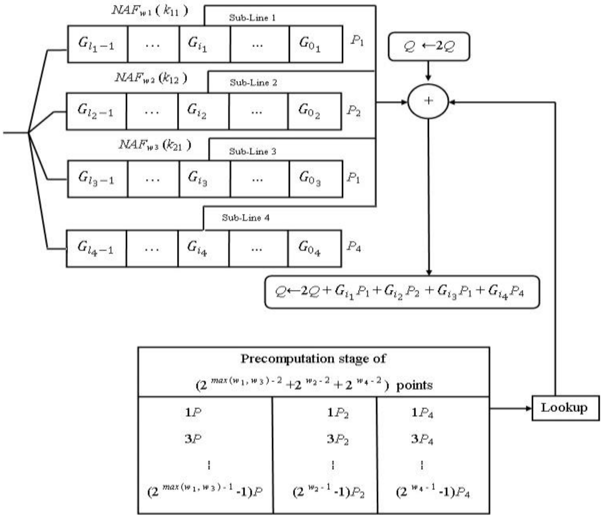


Figure 4: shows the parallel computation of one interleaving $k_{11}P$+$k_{12}\psi_1(P)$+ $k_{21}P$+$k_{22}\psi_2(P)$.

5.4. Algorithm 4 of the interleaving method based on w_j-NAF expansions in one parallel lines. Part 1 (Pre-computation stage)

Input: Integers z_j, widths w_j and points P_j, $j=1,2,3,4.$

Output: The interleaving point $\sum_{j=1:2} z_j P_j$.

```
Set $P_1 = P_3 = P.$
If $(j=2,4)$ then
    For $i=1,3,5:2^{w_j-1}-1$ then
        Compute $iP_j$
    Endfor
Endif
If $(j=1,3)$ then
    If ($w_1$=$w_3$) then
        For $i=1,3,5:2^{w_j-1}-1$ then
            Compute $iP_1$.
        Endfor
    Else ($w_1 \neq w_3$)
        For $i=1,3:2^{\max(w_j)-1}-1$ then
            Compute $iP_j$ .
        Endfor
    Endif
Return $iP_j$ .
Computation of the endomorphism $\psi_1(P)$ and $\psi_2(P)$.
```

5.5. Algorithm 4 of the interleaving method based on w_j-NAF expansions in one parallel line. Part 2 (Parallel computation stage)

```
Set $P_1 = P_3 = P$, $P_2 = \psi_1(P)$ and $P_4 = \psi_2(P)$.
Run parallel computations width-w NAF of positive integers Algorithm
    (2) to compute $NAF_{w_j}\left(\left|z_j\right|\right) = \sum_{i=1}^{l_j-1} G_{i,j} 2^i$ for $j$ from 1 to 4 do.
Set $l = \max\{l_j, j=1,2,3,4\}$.
Define $G_{i,j}=0$ for $i$ from $l_j$ to $l$-1 and for $j$ from 1 to 4 do.
For $i$ from 0 to l do.
    For $j$ from 1 to 4 do.
        If $(z_j < 0)$ then,
            Set $G_{i,j} \leftarrow -G_{i,j}$.
        Else
            Set $G_{i,j} \leftarrow G_{i,j}$.
        Endif
    Endfor
Endfor
$Q \leftarrow \infty$.
For $i$ from $l$-1 down to 0 do.
    $Q \leftarrow 2Q$.
    For $j$ from 1 to 4 do.
        If $(G_{i,j} \neq 0)$ then
            If $(G_{i,j} > 0)$ then
                $Q \leftarrow Q + G_{i,j}P_j$.
            Else
                $Q = Q - \left|G_{i,j}\right|P_j$.
            Endif
        Else
            $Q \leftarrow Q$.
        Endif
    Endfor
Endfor
Return $Q$.
```

6. INTERLEAVING METHOD TO COMPUTE PROPOSED ISD SCALAR MULTIPLICATION BASED ON W$_j$-NAF EXPANSIONS

The idea of GLV method Gallant et al., 2001, is the main source on which the ISD method depends to obtain a faster scalar multiplication on an ordinary elliptic curve E defined in equation (7).

This method primarily aims to sub-decompose the values k_1 and k_2 when one or both values are not bounded by $\sqrt{n}$. The sub-decomposition from equation (1) is expressed by these formulas that defined in equation (2).

To accomplish sub-decomposition, one should first find a GLV generator $\{v_1,v_2\}$ by using a GLV generator algorithm in Kim and Lim, 2003, for a given n and λ, where n is a large prime order of elliptic curve point P

and λ is a root of the characteristic polynomial of endomorphism ψ of E. Consequently, $k \in [1, n\text{-}1]$ is decomposed into k_1 and k_2. This decomposition can be performed using the balanced length-two representation of a multiplier k algorithm in Hankerson et al., 2004. Our modified algorithm can then be used to generate the ISD generators $\{v_3,v_4\}$ and $\{v_5,v_6\}$ such that each component of v_3,v_4,v_5 and v_6 is bounded by $\sqrt{n}$ and relatively prime to each other. These generators can be easily computed by solving the closest vector problem in a lattice that is involved in using an extended Euclidean algorithm in Gallant et al., 2001; Hankerson et al., 2004, k_1 and k_2 can be decomposed again into integers k_{11} k_{12}, k_{21} and k_{22} which means that the sub-decomposition of k as follows:

$$k \equiv k_{11} + k_{12}\lambda_1 + k_{21} + k_{22}\lambda_2 (\text{mod} \;\; n) \tag{13}$$

with $-\sqrt{n} < k_{11}, k_{12}, k_{21}, k_{22} < \sqrt{n}$ from any ISD generators $\{v_3,v_4\}$ and $\{v_5,v_6\}$. Finally, the scalar multiplication kP can be computed by formula (4).

The formula of ISD elliptic scalar multiplication kP defined in equation (4) can be carried out through the applying of two proposed models to compute the interleaving based on wNAF expansions. The processing, on the first model that consists from two parallel lines, each line contains on two parallel sub-lines, needs computing $k_{11}P+k_{12}\psi_1(P)$ and $k_{21}P+k_{22}\psi_2(P)$ separately in two parallel lines but in the same time. The final result of kP comes through the sum of two elliptic points resulting from the computing $k_{11}P+k_{12}\psi_1(P)$ and $k_{21}P+k_{22}\psi_2(P)$. Whereas, the second proposed interleaving model to compute kP consists of one parallel line contains on four parallel sub-lines. The processing takes place to compute one interleaving $k_{11}P+k_{12}\psi_1(P) + k_{21}P+k_{22}\psi_2(P)$ to find the final result of kP . The computation with this model is more efficient in comparison with the first one because it gives more speeding up in computation. The ISD elliptic scalar multiplication kP can be computed by using Algorithm (5).

6.1. Algorithm 5 of ISD Elliptic Scalar Multiplication

Input: The integers $p, n, \lambda, P, w_j, \quad j = 1, 2, 3, 4.$

Output: kP.

```
1. Precomputation stage:
2. Compute two endomorphisms $\psi_1(P) = \lambda_1 P$ and $\psi_2(P) = \lambda_2 P$.
3. Computation stage:
```

4. Run GLV generator Algorithm (1) of Kim and Lim, 2003, to find the generator $\{v_1, v_2\}$ such that $v_1 \leftarrow (r_{m+1}, -t_{m+1})$ and $v_1 \leftarrow (r_m, -t_m)$ or $v_1 \leftarrow (r_{m+2}, -t_{m+2})$.
5. Run balanced length-two representation of a multiplier Algorithm (3.74) of Hankerson et al., 2004, to decompose k into k_1 and k_2.
6. Choose randomly λ_1, $\lambda_2 \in [1, n\text{-}1]$ such that $\lambda_1 \neq \pm \lambda_2$.
7. Run ISD generators Algorithm (1) of Ajeena and Kamarulhaili, 2014a; 2014b,to find $\{v_3, v_4\}$ and $\{v_5, v_6\}$ such that

$$v_3 \leftarrow (r_{m_1+1}, -t_{m_1+1}), v_4 \leftarrow (r_{m_1+2}, -t_{m_1+2}) \text{ or } (r_{m_1}, -t_{m_1}),$$

$$v_5 \leftarrow (r_{m_2+1}, -t_{m_2+1}) \text{ and } v_6 \leftarrow (r_{m_2+2}, -t_{m_2+2}) \text{ or } (r_{m_2}, -t_{m_2}).$$

8. Use Algorithm (2) of Ajeena and Kamarulhaili, 2014, to sub-decompose k_1 and k_2 into $k_1 \equiv k_{11} + k_{12}\lambda_1 (\text{mod } n)$ and $k_2 \equiv k_{21} + k_{22}\lambda_2 (\text{mod } n)$ such that $k \equiv k_{11} + k_{12}\lambda_1 + k_{21} + k_{22}\lambda_2 (\text{mod } n)$.
9. Set P_1=P_3=P, P_2= $\psi_1(P)$ and P_4= $\psi_2(P)$.
10. Use parallel computing w_j-NAF Algorithm (1) or (2) to compute w_jNAF expansions for j=1,2,3,4 of integers k_{11}, k_{12}, k_{21} and k_{22}.
11. Use interleaving Algorithm (3) or (4) to compute kP= $k_{11}P$+ k_{12} $\psi_1(P)$ + $k_{21}P$+ k_{22} $\psi_2(P)$.
12. Return kP.

7. CONCLUSION

Computing the ISD elliptic scalar multiplication kP requires computation of the terms $k_{11}P$, $k_{12}\psi_1(P)$, $k_{21}P$ and $k_{22}\psi_2(P)$. This computation comprises several parts and terms. To compute those terms and parts individually needs more execution time. Thus, it is wise to find ways to compute these terms simultaneously to reduce computation times. In this work, we proposed two new algorithms to compute these terms jointly through the concept of parallel computations. For the first proposed model, the parallel computation of two interleavings $k_{11}P$+$k_{12}\psi_1(P)$ and $k_{21}P$+$k_{22}\psi_2(P)$ can be carried out on two parallel lines. The implementation on this model firstly takes place on the proposed parallel computation of w_j-NAF expansions that also consists of two parallel lines. Each parallel line contains two parallel sub-lines. Each parallel sub-line performs the computations of w_jNAF expansion for j=1,2,3,4. Thereafter, the computation of interleavings $k_{11}P$+$k_{12}\psi_1(P)$ and $k_{21}P$+$k_{22}\psi_2(P)$ are determined by two parallel lines. The sum of the final results from the interleavings that were represented as two elliptic points, give the final result of kP.

However, it is also possible to use another model to compute the terms in ISD elliptic scalar multiplication kP simultaneously. This model uses the parallel computation to compute the interleaving $k_{11}P+k_{12}\psi_1(P)$ + $k_{21}P+k_{22}\psi_2(P)$ on one parallel line. The performance was based on the computation of the generalized w_{j-}NAF expansions in one parallel line. This line consists of four parallel sub-lines that on them are determined the w_{j-}NAF expansions. The processing on the one parallel line model that consists of four parallel sub-lines is more efficient because it implements with less executing time. It provides four times the time used for the implementation. The cost of doubling here determines on the basic the maximum number among k_{11}, k_{12}, k_{21} and k_{22} in comparison with the computation that performed of the terms $k_{11}P$, $k_{12}\psi_1(P)$, $k_{21}P$ and $k_{22}\psi_2(P)$ individually.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to the Fundamental Research Grant (FRGS), funded by the Ministry of Higher Education and the School of Mathematical Sciences, Universiti Sains Malaysia.

REFERENCES

Ajeena, R. K. K. and Kamarulhaili, H. (2013). Analysis on the Elliptic Scalar Multiplication Using Integer Sub-Decomposition Method. *International Journal of Pure and Applied Mathematics*. **87**(1): 95-114.

Ajeena, R. K. K. and Kamarulhaili, H. (2014). Point Multiplication using Integer Sub-Decomposition for Elliptic Curve Cryptography. *Applied Mathematics and Information Sciences*. **8**(2).

Ajeena, R. K. K. and Kamarulhaili, H. (2014). Comparison Studies on Integer Decomposition Method for Elliptic Scalar Multiplication. *Advanced Science Letters*. **20**(2): 526-530.

Barney, B. (2010). Introduction to parallel computing. *Lawrence Livermore National Laboratory*. **6**(13):10.

Gallant, R. P., Lambert, R. J. and Vanstone, S. A. (2001). Faster point multiplication on elliptic curves with efficient endomorphisms. *In Advances in Cryptology—CRYPTO 2001*: 190-200.

Hankerson, D., Vanstone, S. and Menezes, A. J. (2004). *Guide to elliptic curve cryptography*.

Hao, Y., Ma, S., Chen, G., Zhang, X., Chen, H. and Zeng, W. (2008). Optimization Algorithm for Scalar Multiplication in the Elliptic Curve Cryptography over Prime Field. *In Advanced Intelligent Computing Theories and Applications*: 904-911.

Jones, J. E. (1999). A parallel multigrid tutorial. *In Proceedings of the Ninth Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO*: 11-16.

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

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*. **48**(177): 203-209.

Kumar, V., Grama, A., Gupta, A. and Karypis, G. (1994). *Introduction to parallel computing*. Redwood City: Benjamin/Cummings.

Longa, P. and Miri, A. (2011). *U.S. Patent No. 7,991,162*. Washington, DC: U.S. Patent and Trademark Office.

Miller, V. S. (1986). Use of elliptic curves in cryptography. *In Advances in Cryptology—CRYPTO'85 Proceedings*:417-426.

Washington, L. C. (2008). *Elliptic curves: number theory and cryptography*. CRC press.