ELLIPTIC CURVES SCALAR MULTIPLICATION COMBINING MBNR WITH POINT HALVING

ABDULWAHED M. ISMAIL AND MOHAMAD RUSHDAN

Institute for Mathematical Research (INSPEM), University Putra Malaysia, Serdang,43300, Selangore, Malaysia
wahid963@yahoo.com, mrushdan@info.upm.edu.my

Abstract. Elliptic curves scalar multiplication over some finite fields, attractive research area, which paid much attention by researchers in the recent years. Researchs still in progress to improve elliptic curves cryptography implementation and reducing it's complexity. Elliptic curve point-halving algorithm proposed in [11] and later double-base chain [3] and step multi-base chain [19] are among efficient techniques offered in this field.Our paper proposes new algorithm combining step multi-base number representation and point halving. We extend the work done by [14], which combined double base chain with point halving technique. The expriment results show our contribution will enhance elliptic curves scalar multiplication.

Keywords: Elliptic curves cryptography, Integer representation, Multi-number rsystem, Point halving.

1. Introduction

The rapid advances in the information technology in the past few decades led to an intensive researches about information security. Many technologies and cryptographical systems are invented, all to secure information and keep it away from unuthorized invaders. Public-key cryptography widely studied and used since 1975 when R.L. Rivest, A. Shamir, and L. Adleman invented RSA public key cryptography. This system heavily depends on integer factorization problem IFP, using very big size key bits as 1024 bits and 2048 bit and more. Later Deffie-Hellman in [6] developed the public key exchange algorithm using discrete logarithm problem DLP. ElGamal also used DLP in encryption and digital signature scheme. In 1985 Neal Koblitz and Victor Miller independently used elliptic curves ECs in cryptography using Elliptic curves discrete logarithm problem ECDLP in their papers in [8] and [**?**]. In recent years researchers pay more attention to develop the proposed ECC algorithms and improve their efficiency. Improving the efficiency of scalar multiplication in elliptic curves one of the main interests of many researchers in the field of cryptology. The techniques proposed so far, use different tricks for representing the scalar k, which clearly show different level of computation speed and security. Binary representation is extended to signed binary representation , and it's Non-Adjacent Form NAF algorithm [10]. Other well known techniques such as window methods and Montgomery method bring about much improvement in term of the efficiency of elliptic curve arithmetic. When doubling one point P to get

Date: December 26, 1997.

2P=R as a new point on E, requires extra field squaring over prime fields but it is the same cost as in point adding if the curve been defined over the binary fields. As we see in the next section adding two points P and Q on the same elliptic curve E, requires solving three equations, that involve one field inversion, one field squaring and two field multiplications, which are costly operations in EC implementation. Some other operations involved in adding two points are additions, subtractions and multiplication by small integers, which are in most cases neglectible operations.

The proposed algorithms offered many different formula for point multiplication finding from the given point P. One can use many doublings 2(...(2(2P))) with one or more extra point addition. The reverse operation also lead to find points such as finding P from (2P) which is called *point halving* or involve triplings (3P), triple-and-add (3P)+P, quadruplings (4P), quadruple-and-add (4P)+P and recently proposed (5P) for example finding $127P$ such that $P = (x_p, y_p)$, the above options available to represent 127. Using each thechnique requires different amount of operations and different number of computer iteration.

Elliptic curve point-halving algorithm [11] and [12] and later double-base chain [3] and step multi-base chain [19] are among efficient techniques offered in this field. In section 2 we present more detail about the above three algorithms. In section 3 we show our work which extend the work done by [14], which is a combination of double base chain with point halving technique. Our contribution will be combining Multi-base representation with point halving. We expect that our contribution will enhance elliptic curves scalar multiplication.

2. Background

Elliptic curves are used for several kinds of cryptosystems, including key exchange protocols and digital signature algorithms. If q is a prime or prime power, we let F_q denote the field with q elements. When $gcd(q, b) = 1$, an elliptic curve over the field F_q is given by an equation of the form $E : y^2 = x^3 + ax + b$

with a, b in F_q and $4a^3 + 27b^2 \neq 0$. (See [9]).

The general curve equation subsumes the case

$E_2 : y^2 + xy = x^3 + ax^2 + b$

with a, b in F_q and $b \neq 0$, which is used over fields of characteristic 2.

In all cases, the group used for cryptosystem is the group of points on the curve over F_q. If represented in affine coordinates, the points have the form (x, y), where x and y are in F_q and they satisfy the equation of the curve, as well as a distinguished point O (called the point at infinity) which acts as the identity for the group law [4].

The addition of any two distinct points on the eliptic curve will be done through some formula as shown below. For the curves over finite fields of charactristic more than three we consider the simple form of the elliptic curve equation.

When P = (x_P,y_P) and Q = (x_Q,y_Q) are not negative of each other, then P+Q = R.

First step we need to find λ as the slope of the line through P and Q.

$$\lambda = \frac{y_P - y_Q}{x_P - x_Q}$$
$$x_R = \lambda^2 - x_P - x_Q \text{ and}$$
$$y_R = -y_P + \lambda(x_P - x_R)$$

The doubling operation for any point in the elliptic curve group, requires some steps (the result is same point when doubing $2P$, or adding $P + P$). Lets say we want to double the point $P = (x_p, y_p)$, when y_P is not 0,

$2P = D$, where

$$\lambda = \frac{3x_P^2+a}{2y_P}$$
$$x_D = \lambda^2 - 2x_P \text{ and}$$
$$y_D = -y_P + \lambda(x_P - x_D)$$

For each doubling of the point P we need the above procedure to get $2P, 2(2P), 2(2(2P))$, and so on. Here we note, to compute kP, for k any integer, we need a number of doublings and additions until we get $P + P + ... + P$ (k-times). The scalar k in elliptic curves cryptosystems is considered as a secrete key in the system, usually with very long bits. Such point multiplication arithmetic requires much memory and time for running and implementing necessary cryptographical operations.

Points are added using a geometric group law which can be expressed algebraically through rational functions involving x and y. Whenever two points are added, forming $P+Q$, or a point is doubled, forming $2P$, these formulae are evaluated at the cost of some number of multiplications, squarings, and divisions in the field. For example, to double a point in affine coordinates using the short Weierstrass form, costs 1 multiplication, 2 squarings, and 1 division in the field, not counting multiplication by 2 or 3 [2] . To add two distinct points in affine coordinates costs 1 multiplication, 1 squaring, and 1 division in the field. Performing a doubling and an addition $2P+Q$ costs 2 multiplications, 3 squarings and 2 divisions if the points are added as $(P + P) + Q$, i.e., first double P and then add Q [1].

2.1. **Field Operation.** Reducing the field operations over the elliptic curves could speed up the ECC systems. An algorithm proposed by [1] showing some improvement of speed of scalar multiplication on general curves, using affine coordinates. This achieved by eliminating a field multiplication when computing $2P + Q$ from given points P and Q on the curve.

Suppose $P = (x_p, y_p)$ and $Q = (x_Q, y_Q)$ are distinct points on E, and $x_p \neq x_Q$. The point $P + Q$ will have coordinates (x_R, y_R), where

$\lambda_1 = (y_Q - y_p)/(x_Q - x_p)$,

$x_R = \lambda_1 - x_p - x_Q$, and

$y_R = (x_p - x_R)\lambda_1 - y_p$.

Now suppose we want to add$(P + Q)$ to P. We must add (x_p, y_p) to (x_R, y_R) using the above rule. Assume $x_R \neq x_p$. The result has coordinates (x_4, y_4), where

$\lambda_2 = (y_R - y_p)/(x_R - x_p)$,

$x_4 = \lambda_2 - x_p - x_R$, and

$y_4 = (x_p - x_4)\lambda_2 - y_p$.

We can omit the y_R computation, because it is used only in the computation of λ_2, which can be computed without knowing y_R as follows:

$\lambda_2 = -1 - 2y_p/(x_R - x_p)$.

Omitting the y_R computation saves a field multiplication. Each λ formula requires a field division, so the overall saving is this field multiplication. This trick can also be applied to save one multiplication when computing $3P$, the triple of a point $P \neq O$, where the λ_2 computation will need the slope of a line through two distinct points $2P$ and P. This trick can be used twice to save 2 multiplications when computing $3P+Q = ((P+Q)+P)+P$. Thus $3P+Q$ can be computed using

1 multiplication, 3 squarings, and 3 divisions. Such a sequence of operations would be performed repeatedly if a multiplier were written in ternary form and left-to-right scalar multiplication were used. Ternary representation performs worse than binary representation for large random multipliers k, but the operation of triple and add might be useful in another context. A similar trick works for elliptic curve arithmetic in characteristic 2[1]. The field inversion is the most expensive operation over the computer system using affine coordinates (x, y), while we don't need any field inversion when working on projective coordinates (X, Y, Z). This work can be extended through saving more operations and applying the same trick on other eliptic curves. later Ciet et al.[15] proposed a variant to the previous work provided by [1] which is faster whenever a field inversion is more expensive than six field multiplications. The paper provides an improvement when doubling $(2P)$, tripling $(3P)$, and quadrupling $(4P)$ of a point P.

2.2. **Point Halving.** For the same purpose, there are algorithms dealing with the arithmetic of ECs, such as the scalar multiplication kP, which conducted through *double-and-add algorithm* is the origin algorithm used for conducting the operations in EC group law. The opposit operation for the previous algorithm is *halve-and-add algorithm,* which was proposed independently by Knudsen [11] and Schroeppel [12]. The method replacing all point doublings in the double-and-add algorithm with another operation called point halving. This method implemented for conducting scalar multiplication on a non-supersingular elliptic curves in charactristic two. Point halving applied to the curves with minimal two-torsion. For polynomial basis, the disadvantage is the amount of storage needed, while for normal basis, according to [11] there are no disadvantages. The point halving method is faster than doubling method if implemented using affine coordinates.

Let $P = (x, y)$ be a point on the elliptic curve defined over binary field using affine coordinates.

A point doubling requires to calculate the coordinates of the point $Q = 2P = (u, v)$ using the following equations:

$$\lambda = x + \frac{y}{x} \tag{2.1}$$

$$u = \lambda^2 + \lambda + a \tag{2.2}$$

$$v = x^2 + u(\lambda + 1) \tag{2.3}$$

Point halving is just the opposite, i.e., given $Q = (u, v)$, find $P = (x, y)$ such that $Q = 2P$. It is computed by solving equation 2.2 for λ, Eq. (3) for x, and finally, Eq. (1) for y. This means that we have to solve $\lambda^2 + \lambda = u + a$ for λ, $x^2 = v + u(\lambda + 1)$ for x, and finally obtain $y = \lambda x + x^2$.

A detail analysis of the computational complexity of point halving was made in [13].

It was reported that the point halving method is $15 - 24\%$ faster than point doubling. Moreover, this approach performs better when the point P is not known in advance and the inversion-to- multiplication ratio is small [14].

For E over F_{2^n}, the coefficients of the curve lie in a small subfield of F_{2^n}, one can use the Frobenius τ of the field extension to replace doublings . Since the cost of τ is negligible if normal bases are used, the scalar multiplication is written in "base τ" and the resulting "τ -and-add" algorithm gives very good performance . In [17], the authors combined the above two ideas of point halving and Frobenius endomorphism to have a new decomposition of the scalar. This method faster than the mentioned Frobenius method without any precomputation, if applied on Koblitz curves. The combined τ-NAF approach with a single point halving, reducing the amount of point additions from $n/3$ to $2n/7$, and providing a speed with about 14.29% saving. The idea is, using a single point halving, to replace some sequences of a τ-NAF having density 1/2 (and containing at least three non-zero coefficients) with sequences having weight 2.

2.3. **Double Base Chain.** An important contributation by [15] was a new ternary/binary method to perform efficient scalar multiplication. This method evaluate expressions of the form $6P \pm Q$, that can be computed as $2(3P) \pm Q$ or $3(2P) \pm Q$. When using the short Weierstrass form $y^2 = x^3 + ax + b$, the latter takes an extra inversion but saves five (field) multiplications. For binary curves, the costs are $3I + 4S + 11M$ and $2I + 6S + 16M$, so the trade-off is $1I$ for $2S + 5M$..

A similar idea was suggested in [16] when an integer k is represented in double-base number system as the sum or difference of mixed powers of two and three, as given by following definition.

Definition 1. *(Double-Base Chain) Given $k > 0$, a sequence $(K_i)_{i>0}$, of positive integers satisfying:*

$$K_1 = 1,\ K_{i+1} = 2^u 3^v K_i + s_i \ \text{ with } i \geq 2,\ s_i \in \{1, -1\},$$

for some $u, v \geq 0$, and such that $K_m = k$ for some $m > 0$, is called a *double-base chain* for k. The length, m, of a double-base chain is equal to the number of 2-integers in DBNS equation, used to represent k.

Let $P \in E(F_q)$ and $k > 0$ is represented in DBNS as

$$k = \sum_{i=1}^{m} s_i 2^{b_i} 3^{t_i} \text{ with } s_i \in \{1, -1\} \text{ and } b_i, t_i \geq 0.$$

If the sequences of binary and ternary exponents decrease monotonically, i.e.

$b_1 \geq b_2 \geq ... \geq b_m \geq 0$ and $t_1 \geq t_2 \geq ... \geq t_m \geq 0$, a double-base chain is formed.

Algorithm 1. *The proposed DB Chain algorithm is shown below*

Algorithm1: Conversion to DBNS with restricted exponents

Input k, a n-bit positive integer; $b_{max}, t_{max} > 0$, the largest allowed binary and ternary exponents
Output The sequence $(s_i, b_i, t_i)_{i>0}$ such that $k = \sum_{i=1}^{m} s_i 2^{b_i} 3^{t_i}$, with $b_1 \geq \cdots \geq b_m \geq 0$ and $t_1 \geq \cdots \geq t_m \geq 0$
1: $s \leftarrow 1$
2: **while** $k > 0$ **do**
3: define $z = 2^b 3^t$, the best approximation of k with $0 \leq b \leq b_{max}$ and $0 \leq t \leq t_{max}$
4: **print** (s, b, t)
5: $b_{max} \leftarrow b, \quad t_{max} \leftarrow t$
6: **if** $k < z$ **then**
7: $s \leftarrow -s$
8: $k \leftarrow |k - z|$

Algorithm 2. *Point Multiplication in Even Characteristic In even characteristic, i.e., with P ∈ E(F2n) and k defined as above, Algorithm 2 below, computes the new point kP.*

Algorithm2. Double-Base Scalar Multiplication in even characteristic

Input An integer $k = \sum_{i=1}^{m} s_i 2^{b_i} 3^{t_i}$, with $s_i \in \{-1, 1\}$, and such that $b_1 \geq b_2 \geq \cdots \geq b_m \geq 0$, and $t_1 \geq t_2 \geq \cdots \geq t_m \geq 0$; and a point $P \in E(K)$
Output the point $kP \in E(K)$
1: $Z \leftarrow s_1 P$
2: **for** $i = 1, \ldots, m-1$ **do**
3: $u \leftarrow b_i - b_{i+1}$
4: $v \leftarrow t_i - t_{i+1}$
5: **if** $u = 0$ **then**
6: $Z \leftarrow 3(3^{v-1} Z) + s_{i+1} P$
7: **else**
8: $Z \leftarrow 3^v Z$
9: $Z \leftarrow 4^{\lfloor (u-1)/2 \rfloor} Z$
10: **if** $u \equiv 0 \pmod 2$ **then**
11: $Z \leftarrow 4Z + s_{i+1} P$
12: **else**
13: $Z \leftarrow 2Z + s_{i+1} P$
14: **Return** Z

We remark that although m-1 additions are required to compute kP, we never actually use the addition operation (ADD); simply because we combine each addition with either a doubling (Step13), a tripling (Step 6) or a quadrupling (Step 11), using the DA, TA and QA primitives.

As a result, fast computation of scalar multiplication is achieved by the following recursive calculations. For example 314159 which used in [16]. Its double-base chain representation is $314159 = 2^{12}3^4 - 2^{11}3^2 + 2^8 3^1 + 2^4 3^1 - 2^0 3^0$

The calculation successively computes 17P, 409P, 6545P and finally 314159P. If prime field is chosen, it needs to calculate 13 inversions, 55 squarings and 95 multiplications[14]. The analyses made in [16] show that this method is faster than

the classical binary, 2-NAF, 4-NAF and the ternary/binary approach proposed in [15], over both binary and prime fields, without requiring any precomputation.

Advantage taken by [3] in using sparseness and the ternary nature of the double-base number system (DBNS) to propose new point multiplication algorithm based on double-base chains. The speed-ups attained through reducing point additions and improving formulae for point triplings and quadruplings in both even and odd characteristic. Dimitrov et al. show that this algorithm faster than windowing methods and protected against simple and differential side-channel analysis by using side-channel atomicity and classical randomization techniques.

In another work [18] produce a practical algorithm to find a DBNS decomposition, and refine the decomposition into an effective scalar multiplication algorithm to compute nP on some supersingular elliptic curves of characteristic 3. The proposed algorithm can be used for cryptographic protocols based on supersingular curves, such as identity based schemes.

The previous two papers extended by [17] It examine the double-base decompositions of integers n, namely expansions loosely of the form

$$n = \sum_{i,j} A^i B^j$$

for some base $\{A, B\}$ in the case when A, B lie in N. It shows how to extend the results of [18] to Koblitz curves over binary fields, obtaining a sublinear scalar algorithm to compute, given a generic positive integer n and an elliptic curve point P, the point nP in time $O\left(\frac{\log n}{\log \log n}\right)$ elliptic curve operations with essentially no storage, claiming the method faster than any known scalar multiplication algorithm on Koblitz curves. Scalar multiplication using double base numbers been analyzed and shown that on a generic elliptic curve over a finite field, one cannot expect a sublinear algorithm.

Combination of two or more different methods of scalar multiplication is one of the techniques used by researchers for proposing new faster algorithms. In [14] instead of using powers of 2 and 3 in double-base chain representation of scalar, a new representation with decreasing powers of 1/2 and 3 been presented. The point halving operation incorporated in the new double-base chain to achieve fast scalar multiplication. The paper shows that the advantage of this representation is that all point doublings required in the original chain point doubling and quadrupling can be replaced by faster point halving while keeping the tripling operations. For binary fields, the approach requires only about half the number of inversions, one-third the number of squarings, and a fewer number of multiplications if compared with the scalar multiplication using the original double base chain. The idea is to multiply k with a large power of 2, say, 2^q. From the experimental results, can choose 2^q to be a value around the field size. Then we find the remainder k' after modulo the field size p, as given by $k' = 2^q k \bmod p$, then need to obtain the double-base chain of k' with powers of 2 and 3 in the form of increasing binary exponents but decreasing ternary exponents. Some more steps lead to have the following forms of representation k as given by

$$k = \frac{k'}{2^q} = \frac{\sum_{i=1}^{m} s_i 2^{b_i'} 3^{t_i}}{2^q} = \sum_{i=1}^{m} s_i \left(\frac{1}{2}\right)^{(q-b_i')} 3^{t_i} \bmod p \quad \text{with} \quad k' = 2^q k \bmod p$$

where $s_i \in \{1, -1\}$, $0 \le b_1' < b_2' < ... < b_m', t_1 \ge t_2 \ge ... \ge t_m \ge 0, q \ge b_i' \forall i$.

As we see in the following table, we can evaluate proposed algorithm by checking the number of field operations required. Most part of this table offered by [16] and the rest by other mentioned sourses, showing the number of inversions [i], squarings

[s] and multiplications [m] we need for different curve operations over F_p and F_{2^m} with using affine coordinates.

Curveoperation	Binaryfields
P+Q	1[i]+1[s]+2[m]
2P	1[i]+1[s]+2[m]
2P+Q	1[i]+2[s]+9[m]
3P	1[i]+4[s]+7[m]
3P+Q	2[i]+3[s]+9[m]
4P	1[i]+5[s]+8[m]
4P+Q	2[i]+6[s]+10[m]
5P[19]	1[i]+5[s]+13[m]
(1/2)P [14]	– – 1[m]
(1/2)P +Q [14]	1[i] – 5[m]

Table1: The costs of different curve operations

2.4. **Step Multi-Base Number Representation.** The above so called double-base number representation has been generalized to multi-base number representation by it's authors in their recent paper [19]. They proposed two efficient formulas for computing (5P), when P is an elliptic curve point over prime and binary finite fields. This work led to a new scalar multiplication algorithm, which represent the scalar using three bases 2, 3 and 5and computes the scalar multiplication very efficiently.

Definition 2. *a multiple representation* $n = s_i 2^{bi} 3^{ti} 5^{qi}$ *using the bases {2,3,5} is called a step multi-base number representation SMBR, which each exponent* $\{b_i\}$, $\{ti\}$ *and* $\{qi\}$ *are seperate monotonic decresing sequences.*

The length of MBNR are shorter than DBNR and also more redundant, as the number of representations of n grows very fast based of the number of base elements. The example shown in [19] the number 50 has 72 DBN representations using the bases (2 and 3), while has 489 MBN representations using the bases (2, 3 and 5). The special MBNR is more suitable for scalar multiplication algorithms than general MBNR.

Algorithm 3. *An integer can be converted to multi-base representationusing greedy algorithm, which produces shortest representation.*

Algorithm3: mGreedy Algorithm for Conversion into SMBR

Input: k a positive integer; $max2, max3, max5 > 0$, the largest allowed binary, ternary and quinary exponents and the array $T[0..max2;\ 0..max3;\ 0..max5]$.

Output: The sequence $(s_i, b_i, t_i, p_i)_{i>0}$ such that $k = \sum_{i=1}^{m} s_i\, 2^{b_i}\, 3^{t_i}\, 5^{p_i}$, with $b_1 \geq \ldots \geq b_m \geq 0$, $t_1 \geq \ldots \geq t_m \geq 0$, $p_1 \geq \ldots \geq p_m \geq 0$.

$s \leftarrow 1$
while $k > 0$ **do**
  for(b=0 to max2, t=0 to max3, p=0 to max5)
    $z = T[b, t, p]$, the best approximation of k
  print (s, b, t, p)
  $max2 \leftarrow b, \quad max3 \leftarrow t, \quad max5 \leftarrow p$
  if $k < z$ **then**
    $s \leftarrow -s$
  $k \leftarrow |k - z|$

Algorithm 4. *The scalar multiplication using SMBR is a generalization of the algorithm of scalar multiplication using DBNR.*

Algorithm4: Scalar Multiplication for Curves over Binary Finite Fields

Input: An integer $k = \sum_{i=1}^{m} s_i 2^{b_i} 3^{t_i} 5^{p_i}$, with $s_i \in \{-1, 1\}$, and such that $b_1 \geq b_2 \geq \ldots \geq b_m \geq 0$, $t_1 \geq t_2 \geq \ldots \geq t_m \geq 0$ and $p_1 \geq p_2 \geq \ldots \geq p_m \geq 0$ and a point $P \in E(F_q)$
Output: the point $kP \in E(F_q)$

$Z \leftarrow s_1 P$
for $i = 1, \ldots, m-1$ **do**
  $u \leftarrow b_i - b_{i+1}$
  $v \leftarrow t_i - t_{i+1}$
  $x \leftarrow p_i - p_{i+1}$
  if $u = 0$ **then**
    $Z \leftarrow (5^x Z)$
    if $v \neq 0$ **then**
      $Z \leftarrow 3(3^{v-1} Z) + s_{i+1} P$ //(TA used here)
    else
      $Z \leftarrow Z + s_{i+1} P$
  else
    $Z \leftarrow 5^x Z$
    $Z \leftarrow 3^v Z$
    $Z \leftarrow 2^{u-1} Z$
    $Z \leftarrow 2Z + s_{i+1} P$ //(DA used here)
Return Z

According to the experiments conducted by the authors of [19] the multi-base algorithm perform faster and more competitive comparing to the other sequential scalar multiplication algorithms. In algorithm 4 for computng the scalar multiplications over binary finite fields, the required curve operations can be calculated as b_i doublings, t_i triplings and q_i quintauplings. The number of curve additions is as same as the number of terms in the chain. Whenever the components of the binary and ternary are not zero, we can use double-and-add and triple-and-add operations instead of curve addition.

The authors of SMBR showing unability to provide theoritical proofs of efficiency of this scalar multiplication algorihms. The average performance shown in applying them to the huge numbers of randomly chosen scalars.

3. The Proposed Method

In order to have faster ECC scalar multiplication, we propose a new multi-base chain representation for scalars. We modify mixed powers of 2, 3 and 5, proposed in [19] by representing the scalar by a new multi-base chain with monotonic decreasing powers of $1/2, 3$ and 5. With this method we remove point doubling and quadrupling to use point halving instead, with keeping the tripling and quintupling points operations. Besides of a number of point additions, which equal to the number of the terms in the chain. Halve-and-add operation can be used instead of normal point addition as long as the power of $(1/2)$ is not zero. In case the power of $(1/2)$ equal to zero we need to use triple-and-add operation if the power of the base (3) not zero. In the worst case when dont have the above choices we have to use the normal point additin since we dont have any (quintaple-and-add) formula.

We take the same technique used by [14] to apply it in our proposed new multi-base number representation.

Some more steps lead to have the following forms of representation k as given by

(3.1)
$$k = \frac{k'}{2^q} = \frac{\sum_{i=1}^{m} s_i 2^{b_i'} 3^{t_i} 5^{l_i}}{2^q} = \sum_{i=1}^{m} s_i \left(\frac{1}{2}\right)^{(q-b_i')} 3^{t_i} 5^{l_i} \bmod p \quad with \quad k' = 2^q k \bmod p$$

where $s_i \in \{1, -1\}$, $0 \leq b_1' < b_2' < ... < b_m', t_1 \geq t_2 \geq ... \geq t_m \geq 0, l_1 \geq l_2 \geq ... \geq l_m \geq 0, q \geq b_i' \forall i$.

The first step in our algorithm starts with multiplying the origin scalar k by $2^q \bmod p$,which is preferable to find 2^q with a value around the field size . Then we find the remainder $k\prime$ as given by next equation.

(3.2)
$$k\prime = 2^q k \bmod p$$

The next step is to obtain the multi-base chain of $k\prime$ with powers of 2, 3 and 5 in the form of increasing binary exponents but decreasing ternary and quinary exponents. We achieve this by an iterative approach. First, we find n such that $k = 0 \bmod n$, with the trial of n in the order of $\{30, 27, 25, 24, 20, 18, 16, 15, 12, 10, 9, 8, 6, 5, 4, 3, 2\}$ respectively. The formula can be as: $k = 0 \bmod n$,return $2^{n_1} \cdot 3^{n_2} \cdot 5^{n_3}(\frac{k}{n})$ when always $2^{n_1} \cdot 3^{n_2} \cdot 5^{n_3} = n$, for example

If $k = 0 \bmod 30$, return $2 \cdot 3 \cdot 5(\frac{k}{30})$

If $k = 0 \bmod 27$, return $3^3(\frac{k}{27})$

$\vdots$

If $k = 0 \bmod 3$, return $3(\frac{k}{3})$

If $k = 0 \bmod 2$, return $2(\frac{k}{2})$

In any round if such modular not found; the algorithm will find k_1-a power of 2, which should be the closest to k, then finds $|k - k_1|$. The power of 2 is chosen as an approximation to k since doubling (which becomes halving later) is cheaper than tripling. As the returned value is getting smaller and smaller, it can always be approximated by a lower power of 2 in next round.

Thus, the binary exponents in this multi-base chain keep strictly decreasing and so triple-and-add and quintuple-and-add operations are not required in this scalar multiplication. The recursion never stops until k is equal to 1, a power of 2 or a power of 3, i.e., a positive number that can be represented by $2^b 3^t$ for any non-negative integers b and t. This iterative algorithm will return the terms in an order from the highest power of 2 times the lowest power of 3 times the lowest power of 5 to the lowest power of 2 times the highest power of 3 times the highest power of 5. If we reverse the order of the terms, i.e., the last term becomes the first term, the expression becomes the desired multi-base chain with increasing binary exponents but decreasing ternary and quinary exponents. Finally, we divide the double-base chain by 2^q to make all the binary exponents negative but with decreasing magnitude. The ternary and quinary exponents are unaffected and are

k=31415 k'=k(2^{18}) mod314161	
multi-base chain of k'	$2^1 3^3 5^1 + 2^4 3^3 5^1 + 2^8 3^1 5^1 + 2^{15} 3^1 5^0$
The new multi-base chain of k=k'/2^{18}	$(1/2)^{17}\, 3^3\, 5^1 + (1/2)^{14}\, 3^3\, 5^1 + (1/2)^{10}\, 3^1\, 5^1 + (1/2)^3\, 3^1\, 5^0$

chains representing k=314159	[i]	[s]	[m]
Step multi-base number representation:			
$2^8\, 3^5\, 5^1 - 2^3\, 3^4\, 5^1 + 2^3\, 3^1\, 5^1 + 2^0\, 3^0\, 5^0$	15	36	80
Our Proposed method			
$(1/2)^{17}\, 3^3\, 5^1 + (1/2)^{14}\, 3^3\, 5^1 + (1/2)^{10}\, 3^1\, 5^1 + (1/2)^3\, 3^1\, 5^0$	7	17	77

all positive or zero with decreasing magnitude. This is actually a new multi-base chain with decreasing powers of 1/2,3 and 5 with value equal to k.

Example 1. *Here we try to use the same example used in prevoius papers* [14]

if the field size p is chosen as 314161, the scalar k=314159 becomes 104034 after multiplied 2^{18} *and mod 314161. The steps in finding the new multi–base chain representation of 314159 are shown in the following table. For this example in both algoeithm the number of terms are 4 terms. While the cost elliptic curve arithmetic and field operations are not same in the two algorithms. As shown in the table using our algorithm costs less field inversions [i],less field squarings [s] and less field multiplications [m]*

4. Experimental Results

To get better evaluation of the new representation we compare step multi-base number representation SMBR with the new combination by testing some recommended big size numbers. The algorithms programmed on C and C++ with running them on the LINUX platform. Our experiment based on testing both algorithms in same platform and invironment. Same hardware and software have been used during entire experiment. Three binary field sizes have been selected, 163 bits, 233 bits and 283 bits. For each field size above we tested 100 randomely selected integers, for each random number both algorithms applied and been run, such that at the end the average costs of that 100 numbers been considered. The number of terms of each scalar found, the curve operations and field arithmetic costs calculated, including the number of inversions, squarings and multiplications required. In the tables shown below [D] refers to number of doublings required, [DA] used for double-and-add, [T] for triplings, [TA] for triple-and-add, [Q] for quintuplings, [A] for additions, [H] for halvings, [HA] for halve-and-add, [i] for field inversions,

[s] for squarings and [m] for multiplications. As shown in the table below the average savings of field inversions is 45.5%, squarings about 53% and multiplications is about 8.6%.

For computng the scalar multiplications over binary finite fields, the required curve operations can be calculated as b_i doublings, t_i triplings and q_i quintauplings, which is the same formula offered by [19]. The number of curve additions is as same as the number of terms in the chain. Whenever the components of the binary and ternary are not zero, double-and-add and triple-and-add operations been used instead of curve addition.

Number of elliptic curve and field operations over binary fields using Step multi-base representation

No of Bits	**[Terms]**	**[D]**	**[DA]**	**[T]**	**[TA]**	**[Q]**	**[A]**	**[i]**	**[s]**	**[m]**
163 bit	31	38	25	34	2	19	1	124	331	824
233 bit	44	56	38	47	2	26	2	176	465	1163
283 bit	53	68	46	57	2	32	3	213	567	1412

Number of elliptic curve and field operations over binary fields using our proposed algorithm

No of Bits	**[Terms]**	**[H]**	**[HA]**	**[T]**	**[Q]**	**[i]**	**[s]**	**[m]**
163 bit	34	129	32	24	10	67	149	729
233 bit	47	185	46	35	15	97	218	1051
283 bit	58	224	57	42	18	117	259	1267

In some cases the average number of terms in our algorithm is more than the terms in the original multi-base algorithm, but still our method costs less than the original algorithm.

5. Conclusion

Some scalar multiplication algorithms such as point halving and double base are among the well known algorithms in speeding up elliptic curve arithmetic. Recently, Mishra and Dimitrov proposed a generalization of their previous work DB chain In our work we proposed a new multi-base number representation algorithm combining two scalar multiplication algorithms. The experiment shows great improvement of step multi-base number represention algorithm after it is combined with the point halving algorithm. The main procedure in the new algorithm using elliptic curve point halving and halve-and-add operations instead of using elliptic curve point doubling and double-and-add operations, which are less cost as we showed in

table1. The results show that this new algorithm will enhance elliptic curves scalar multiplication, which can applied to any other related applications.

References

[1] K. Eisenträger, K. Lauter and P. L. Montgomery, *Fast elliptic curve arithmetic and improved Weil pairing evaluation.* In (M. Joye ed.), Topics in Cryptology – CT-RSA 2003, Lecture Notes in Computer Science, 2612 Springer-Verlag, Berlin (2003) pp. 343-354.

[2] I. F. Blake, G. Seroussi and N. P. Smart, *Elliptic Curves in Cryptography*, London Mathematical Society Lecture Note Series, 265, Cambridge University Press, Cambridge (2000).

[3] V. S. Dimitrov, L. Imbert, and P. K. Mishra. *Efficient and secure elliptic curve point multiplication using double-base chains.* In Advances in Cryptology - ASI-ACRYPT 2005, volume 3788 of Lecture Notes in Computer Science, pages 59-78. Springer, 2005.

[4] L. C. Washington, *Elliptic Curves: Number Theory and Cryptography*, Chapman & Hall/CRC, 2003.

[5] V.S. Miller, *The Weil Pairing, and its efficient calculation.* Journal of Cryptology, 17, 235–261, 2004.

[6] W. Diffie and M. E. Hellman, *New directions in cryptograpgy*, IEEE Trans.Inform. Theory, IT-22(6), November 1976.

[7] T. ElGamal, *A public-key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Trans. Inform. Theory, IT-31(4):469-472, July 1985.

[8] N. *Koblitz, Elliptic curve cryptosystems*, Math. Comp., 48(177): 203-209, January 1987.

[9] J. H. Silverman, *The Arithmetic of Elliptic Curves*, Graduate Texts in Mathematics, 106, Springer-Verlag, 1986.

[10] D. Hankerson, A. J. Menezes, and S. A. Vanstone. *Guide to elliptic curve cryptography.* Springer-Verlag, Berlin, 2003.

[11] E. Knudsen, *Elliptic scalar multiplication using point halving.* Advances in Cryptology—ASIACRYPT 99, Lecture Notes in Computer Science 1716:135–149, 1999.

[12] R. Schroeppel, *Elliptic Curve Point Ambiguity Resolution Apparatus and Method*, International Patent Application Number PCT/US00/31014 filed 9 November (2000).

[13] K. Fong, D. Hankerson, J. Lopez, A. Menezes, *Field inversion and point halving revisited*, IEEE Transactions on Computers 53 (8)(2004) 1047–1059.

[14] K.W. Wong, Edward C.W. Lee, L.M. Cheng, Xiaofeng Liao, *Fast elliptic scalar multiplication using new double-base chain and point halving,* Applied Mathematics and Computation, 2006.

[15] M. Ciet, M. Joye, K. Lauter, P.L. Montgomery, *Trading Inversions for Multiplications in Elliptic Curve Cryptography*, Cryptology ePrint Archive, Report 2003/257 (2003). Also to appear in Design, Codes and Cryptography.

[16] V.S. Dimitrov, L. Imbert, P.K. Mishra, *Fast Elliptic Curve Point Multiplication using Double-Base Chains*, Cryptology ePrint Archive, Report 2005/069 (2005).

[17] R. Avanzi, M. Ciet, and F. Sica, *Faster scalar multiplication on Koblitz curves combining point halving with the Frobenius endomorphism*, Proceedings of Public Key Cryptography 2004, Singapore, March 1-4, 2004, Lecture Notes in Comput. Sci., vol. 2947, Springer, 2004, pp. 28-40.

[18] M. Ciet and F. Sica. *An Analysis of Double Base Number Systems and a Sublinear Scalar Multiplication Algorithm.* In E. Dawson and S. Vaudenay, editors, Progress in Cryptology - Proceedings of Mycrypt 2005, volume 3715 of Lecture Notes in Computer Science, pages 171-182. Springer, 2005.

[19] P. K. Mishra, V. S. Dimitrov. *Efficient Quintuple Formulas for Elliptic Curves and Efficient Scalar Multiplication Using Multibase Number Representation.* Springer-Verlag, 2007, volume 4779, pages 390-406. Springer, 2005.