Joint Sparse Form of Window Three for Koblitz Curve

Yong Ding[1,2], Kwok-Wo Wong[2], and Yu-Min Wang[3]
(Corresponding author: Kwok-Wo Wong)

Department of Computer Science and Mathematics, Guilin University of Electronic Technology[1]
Guilin, Guangxi 541004, P.R. China
Department of Electronic Engineering, City University of Hong Kong[2]
83 Tat Chee Avenue, Kowloon Tong, Hong Kong (Email: itkwwong@cityu.edu.hk)
National Laboratory on ISN, Xidian University[3]
Xi'an Shanxi 710071, P.R. China

(Received July 18, 2005; revised and accepted Aug. 19, 2005)

Abstract

The joint sparse form (JSF) for the non-adjacent form (NAF) representation of two large integers a and b, was proposed by Solinas. Then Ciet extended it to the ϕ-JSF for the ϕ-NAF representations of a and b using the endomorphism ϕ when computing $aP+bQ$, where P and Q are two points on the elliptic curve, in elliptic curve cryptography (ECC). It can be observed that τ-JSF is a special case of ϕ-JSF. In this paper, we will extend the τ-JSF idea to window 3 (RTNAF_3), referred to as window three τ - joint sparse form (WTT-JSF). Mathematical analysis shows that a number of additions can be eliminated with this representation. Moreover, a detail derivation of the length and density of this form is given. The density is 11/27 which is lower than 7/16 when RTNAF_3 is applied directly.

Keywords: ϕ-JSF, JSF, RTNAF$_w$, WTT-JSF

1 Introduction

Elliptic curve cryptosystem (ECC) was first proposed by Koblitz [5] and Miller [7] independently in 1985, and has been widely studied in recent years due to its higher security strength per key bit over existing public key cryptographic algorithms such as RSA [8]. The best known algorithm for solving the underlying mathematical problem of ECC, referred to as the elliptic curve discrete logarithm problem, takes full exponential time [11]. On the contrary, sub-exponential-time algorithms are known for tackling the integer factorization and the discrete logarithm problems that RSA and DSA are relied on [3, 4]. This implies that the algorithms for solving the elliptic curve discrete logarithm problem become infeasible much more rapidly as the problem size increases than those algorithms for the integer factorization and the discrete logarithm problems. For this reason, ECC offers a security level equivalent to RSA and DSA while using a far smaller key size [11].

In ECC applications such as signature verification in Elliptic Curve Digital Signature Algorithm (ECDSA) [1], there is frequently a need to compute the $aP + bQ$ operation, where a are b are large integers, P and Q are two points on the elliptic curve. In order to perform this computation efficiently, the joint sparse form (JSF) was proposed [10]. It is based on the non-adjacent form (NAF) representation of a and b, with a little change. Furthermore, the ϕ-JSF was given by Ciet in [2] after defined the endomorphism ϕ. To obtain the ϕ-JSF, the integers a and b are first decomposed to ϕ-NAF, with a little modification. In [9], the RTNAF representation of k is utilized to speed up the computation of kP in Koblitz curve with base of endomorphism τ, where τ is a special case of ϕ. Thus, the τ-JSF can be obtained by applying τ to ϕ in ϕ-JSF. The window technique can be applied to RTNAF representation [9] and the corresponding representation is denoted as RTNAF_w where w is the window size. In this paper, we will give a JSF of RTNAF_3, in which a and b are represented by RTNAF_3 with a little modification to obtain as many double zero positions as possible. We call this form the window three τ - joint sparse form (WTT-JSF).

The rest of the paper is organized as follows. In Section 2, the JSF is briefly introduced while the RTNAF_w is described in Section 3. In Section 4, the algorithm for obtaining the proposed WTT-JSF is given, together with a detail analysis of the length and the density. Finally, a conclusion is drawn in Section 5.

2 JSF Representation

In order to speedup the computation of $aP + bQ$, Algorithm 1 is given to obtain the JSF of a given pair of integers a and b. The notation $c = a$ mods b means that $c \equiv a \bmod b$ and $-b/2 \leq c < b/2$.

Algorithm 1 *(**JSF**)*
Input *Nonnegative integers a and b*

Output *JSF of a and b in the form of* $(u_{0,m-1}, u_{0,m-2}, \cdots, u_{0,1}, u_{0,0})$ *and* $(u_{1,m-1}, u_{1,m-2}, \cdots, u_{1,1}, u_{1,0})$

Process

Set $j = 0$, $k_0 = a$ *and* $k_1 = b$
While $k_0 > 0$ *or* $k_1 > 0$ *do*
  For $i = 0$ *to 1 do*
    If k_i *is even*
      then $u = 0$
    else
      $u = k_i$ *mods 4*
      If $k_i \equiv \pm 3 (\bmod 8)$ *and* $k_{1-i} \equiv 2 (\bmod 4)$
        then $u = -u$
  Set $u_{i,j} = u$
  Next i
  For $i = 0$ *to 1 do*
    $k_i = (k_i - u_{i,j})/2$
  Next i
  $j = j + 1$
End while

The JSF possesses the following two properties, namely, JSF-1 and JSF-2, as proved in [10].

(JSF-1) Of any three consecutive positions, at least one is a double zero. In other words, for any positions i and j, we have $u_{i,j+k} = u_{1-i,j+k} = 0$ for $k = 0$ and ± 1.

(JSF-2) The probability of occurrence of double zero, which satisfies $u_{i,j} = u_{1-i,j} = 0$ for any position j, is $1/2$.

After obtained the JSF of a and b, we have

$$\begin{aligned} aP + bQ \quad = \quad & 2(\cdots(2(2(u_{0,m-1}P + u_{1,m-1}Q) + \\ & (u_{0,m-2}P + u_{1,m-2}Q)) + (u_{0,m-3}P + \\ & u_{1,m-3}Q)) + \cdots) + (u_{0,0}P + u_{1,0}Q). \end{aligned} \tag{1}$$

If $P \pm Q$ are precomputed and stored, $(m-1)$ doublings and $(m-1)/2$ additions are required in Equation (1). If aP and bQ are calculated separately, $2(m-1)$ doublings and $2(m-1)/3$ additions are required. On the other hand, $(m-1)$ doublings and $(m-1)5/9$ additions are needed if the NAF representations of a and b are applied to Equation (1). From the above analyzes, it can be observed that JSF leads to a reduction in the computational complexity of $aP + bQ$.

3 Window RTNAF

Koblitz curve was first suggested in [6]. Its equation is

$$E_a : y^2 + xy = x^3 + ax^2 + 1.$$

It is defined over the finite field $GF(2^m)$, with $a = 0$ or 1. There is an endomorphism τ defined on the elliptic curve group $E_a(GF(2^m))$. For a given point $P = (x, y)$ belonging to $E_a(GF(2^m))$, we have

$$\tau(x, y) = (x^2, y^2).$$

As $(\tau^2 + 2)P = \mu\tau P$, where $\mu = (-1)^{(1-a)}$, τ can be regarded as a complex number satisfying $(\tau^2 + 2) = \mu\tau$ and so we have $\tau = \frac{\mu + \sqrt{-7}}{2}$. The ring $Z(\tau)$ is a set of $a + b\tau$ for all $a, b \in Z$. In order to compute kP, the value of $\gamma = x_0 + x_1\tau = k \bmod \delta$ is first calculated by the partial reduction modulo δ method [9], where $\delta = (\tau^m - 1)/(\tau - 1)$. Then it is decomposed into a polynomial of τ with coefficient 0 or ± 1. This polynomial is denoted as RTNAF(k). Finally, kP can be obtained by using RTNAF(k).

The window technique can be applied to RTNAF(k) and the corresponding representation is denoted as RTNAF$_w(k)$. It is described as follows.

Let $t_w = 2U_{w-1}U_w^{-1} \bmod 2^w$, where $\{U_k\}$ is the Lucas sequence defined by $U_0 = 0, U_1 = 1, U_{k+1} = \mu U_k - 2U_{k-1}$ for $k \geq 1$. Define $\alpha_i \equiv i \bmod \tau^w$ for $i \in \{1, 3, \ldots, (2^{w-1} - 1)\}$. The RTNAF$_w$ method works as follows. Firstly, the partial reduction modulo method is used to obtain $\theta = x_0 + x_1\tau = k \bmod \delta$, then RTNAF$_w(k)$ = TNAF$_w(\theta) = \sum_{i=0}^{l-1} u_i\tau^i$ is obtained by Algorithm 2, where $u_i \in \{0, \pm\alpha_1, \pm\alpha_2, \ldots, \pm\alpha_{2^{w-1}-1}\}$.

Algorithm 2 *(**TNAF**$_w$)*
Input $\theta = x_0 + x_1\tau, w, t_w, \alpha_i = \beta_i + \gamma_i\tau$ *for all* $i \in \{1, 3, \ldots, (2^{w-1} - 1)\}$

Output *TNAF*$_w(\theta)$

Process

Set $i = 0$;
While $x_0 \neq 0$ *or* $x_1 \neq 0$ *do*
  If x_0 *is odd*
    then $u = x_0 + x_1 t_w$ *mods* 2^w
    If $u > 0$
      then $s = 1$
    else $s = -1, u = -u$
      $x_0 = x_0 - s\beta_u, x_1 = x_1 - s\gamma_u, u_i = s\alpha_u$
  else $u_i = 0$
    $t = x_0, x_0 = x_1 + \mu x_0/2, x_1 = -t/2, i = i + 1$
End while

4 WTT-JSF

As the JSF is for the NAF representation of integers, it is natural to extend it to the RTNAF representation. However, Ciet has presented a ϕ-JSF form in [2], to which

RTNAF-JSF is a special case. Here we extend the JSF idea to RTNAF_w. We will study the case of $w = 3$, which is called window three τ - joint sparse form (WTT-JSF). The algorithm for generating this form will be described and the length and density of this form will be analyzed.

For a given element $\alpha \in Z(\tau)$, let $(\text{TNAF}_3(\alpha))_5$ denote the five least significant bits of $\text{TNAF}_3(\alpha)$. Let $\times$ denote the nonzero bit of $(\text{TNAF}_3(\alpha))_5$. In the computation of $aP + bQ$, we first obtain the WTT-JSF of a and b by Algorithm 3. Then we compute

$$\begin{aligned} aP + bQ &= \tau(\cdots(\tau(\tau(u_{0,l-1}P + u_{1,l-1}Q) \\ &\quad +(u_{0,l-2}P + u_{1,l-2}Q)) + (u_{0,l-3}P \\ &\quad +u_{1,l-3}Q)) + \cdots) + (u_{0,0}P + u_{1,0}Q). \end{aligned}$$

where the WTT-JSF of a and b is $(u_{0,l-1}, u_{0,l-2}, \ldots, u_{0,1}, u_{0,0})$ and $(u_{1,l-1}, u_{1,l-2}, \ldots, u_{1,1}, u_{1,0})$, respectively.

Algorithm 3 *(**WTT-JSF**)*
Input *Integer a and b,* t_3, $\alpha_i = \beta_i + \gamma_i\tau = i \bmod \tau^3$ *for* $i \in \{1, 3\}$

Output *WTT-JSF of a and b*

Process

Compute $\beta = x_0 + y_0\tau = a \bmod \delta$
  and $\gamma = x_1 + y_1\tau = b \bmod \delta$
  $l_0 = \beta = x_0 + y_0\tau$, $l_1 = \gamma = x_1 + y_1\tau$, *j=0*
While $l_0 \neq 0$ *or* $l_1 \neq 0$ *do*
  For i = 0 to 1 do
    If x_i *is odd*
      then $u_{i,j} = 0$
    else $u = x_i + y_i t_3$ *mods 8*
    If$(TNAF_3(l_i))_5 = (\times, 0, 0, \times, 0)$ *and*
     $(TNAF_3(l_{1-i}))_5 = (0, \times, 0, 0, \times)$
      then u = (u-2) mods 8
    else
      If $u > 0$
        then $s = 1$
      else $s = -1$ *and* $u = -u$
      $u_{i,j} = s\alpha_u$, $x_i = x_i - s\beta_u$, $y_i = y_i - s\gamma_u$
  Next i
  For $i = 0$ *to 1 do*
    $t = x_i$, $x_i = y_i + \mu x_i/2$, $y_i = -t/2$
    $l_i = x_i + y_i\tau$
  Next i
  $j = j + 1$
End while

The WTT-JSF possesses the following three properties, namely, WTT-JSF-1, WTT-JSF-2 and WTT-JSF-3.

(WTT-JSF-1) Of any 5 consecutive positions, at least two of them contain double zero.
Proof:
For any element l_i, the possible value set of $(\text{TNAF}_3(l_i))_5$ is $S = \{(0,0,0,0,0), (\times,0,0,0,0), (0,\times,0,0,0), (0,0,\times,0,0), (0,0,0,\times,0), (0,0,0,0,\times), (\times,0,0,\times,0), (\times,0,0,0,\times), (0,\times,0,0,\times)\}$. Thus, the number of possible cases of $((\text{TNAF}_3(l_0))_5, (\text{TNAF}_3(l_1))_5)$ are 81.

If $((\text{TNAF}_3(l_0))_5, (\text{TNAF}_3(l_1))_5)$ is $((\times,0,0,\times,0), (0,\times,0,0,\times))$ or $((0,\times,0,0,\times), (\times,0,0,\times,0))$, they will be changed to $((\times,\times,0,0,*), (0,\times,0,0,\times))$ or $((0,\times,0,0,\times), (\times,\times,0,0,*))$ by Algorithm 3, where * stands for zero or nonzero bit. For the second and third rightmost positions, they are both double zero.

For the other cases, the algorithm will not modify them. However, it can be checked one by one that there are at least two double zero positions at a consecutive of five positions. **Q.E.D.**

Table 1: The states A, B, and C

S_j	$u_{0,j} = u_{1,j} = 0$	S_{j+1}
A	Yes	A, B, C
B	No	A
C	No	B

(WTT-JSF-2) The length of WTT-JSF is at most $m + 4$.
Proof:
The WTT-JSF length is the longer one of the WTT-JSF of the two integers. Moreover, the representation is based on RTNAF_3 with length at most $m + 1$. The possible modification occurs when the TNAF_3 is $(\times, 0, 0, \times, 0)$. By checking one by one on the selection of $u = \pm 1$ and whether "$\times$" is $\pm\alpha_1$ or $\pm\alpha_3$, it can be found that the change made by Algorithm 3 will only affect three bits after $(\times, 0, 0, \times, 0)$. Thus, this property is held. **Q.E.D.**

As the length is limited and the algorithm is deterministic, the WTT-JSF of an integer pair a and b exists and only exists in one form.

For the third property, WTT-JSF-3, the definition and lemmas listed below are needed. In the course of running Algorithm 3, every loop has a state $S_j = ((\text{TNAF}_3(l_0))_5, (\text{TNAF}_3(l_1))_5)$ and an output $(u_{0,j}, u_{1,j})$. For every iteration j, it can be regarded as generating $(u_{0,j}, u_{1,j})$ with input S_j and changing the state to S_{j+1}. For all the 81 possible cases, they can be classified as three different states, denoted by A, B and C. Their relations and outputs are listed in Table 1.

State A indicates that it will output double zero. For state B, it will not output any double zero in this iteration, but will do so in the next iteration. In other words, state S_{j+1} will generate double zero. However, state C will not give any double zero output in the first two loops, but will do so after two iterations. This means that the output of states S_j and S_{j+1} is not double zero, but that of S_{j+2} is.

Lemma 1 $P(u_{i,j+1} = 0 \mid u_{i,j} = 0) = 5/8$ *and* $P(u_{i,j+1} = \times \mid u_{i,j} = 0) = 3/8$ *for all* j.

Proof:

$$P(u_{i,j+1} = 0 \mid u_{i,j} = 0)$$

$$
\begin{aligned}
&= P(u_{i,j-1}=\times)P((u_{i,j+1}=0 \mid u_{i,j}=0 \mid u_{i,j-1}=\times)+\\
&\quad P(u_{i,j-1}=0)P((u_{i,j+1}=0 \mid u_{i,j}=0) \mid u_{i,j-1}=0)\\
&= P(u_{i,j-1}=\times)P(u_{i,j+1}=0 \mid u_{i,j}=0, u_{i,j-1}=\times)+\\
&\quad P(u_{i,j-1}=0)P(u_{i,j+1}=0 \mid u_{i,j}=0, u_{i,j-1}=0)\\
&= 1/4+(3/4)/2=5/8.
\end{aligned}
$$

Using the same method, it can be proved that $P(u_{i,j+1}=\times \mid u_{i,j}=0)=3/8$. **Q.E.D.**

Lemma 2 *$P(S_{j+1}=A \mid S_j=A)=25/64$, $P(S_{j+1}=B \mid S_j=A)=24/64$, $P(S_{j+1}=C|S_j=A)=15/64$ for all j.*

Proof:

$$
\begin{aligned}
&P(S_{j+1}=A \mid S_j=A)\\
&= P(u_{0,j+1}=0 \mid u_{0,j}=0)P(u_{1,j+1}=0 \mid u_{1,j}=0)\\
&= 25/64.
\end{aligned}
$$

$$
\begin{aligned}
&P(S_{j+1}=B \mid S_j=A)\\
&= P(u_{0,j+2}=0, u_{0,j+1}=0 \mid u_{0,j}=0)\\
&\quad P(u_{1,j+2}=0, u_{1,j+1}=\times \mid u_{1,j}=0)+\\
&\quad P(u_{1,j+2}=0, u_{1,j+1}=0 \mid u_{1,j}=0)\\
&\quad P(u_{0,j+2}=0, u_{0,j+1}=\times \mid u_{0,j}=0)+\\
&\quad P(u_{1,j+2}=0, u_{1,j+1}=\times \mid u_{1,j}=0)\\
&\quad P(u_{0,j+2}=0, u_{0,j+1}=\times \mid u_{0,j}=0)\\
&= 24/64
\end{aligned}
$$

$$
\begin{aligned}
&P(S_{j+1}=C \mid S_j=A)\\
&= 1-P(S_{j+1}=A \mid S_j=A)-P(S_{j+1}=B \mid S_j=A)\\
&= 15/64.
\end{aligned}
$$

Q.E.D.

After the preparation of state definition, Lemmas 1 and 2, we have the third property.

(WTT-JSF-3) The density of WTT-JSF is 11/27.
Proof:
Let

$$
\begin{aligned}
P(S_j=A) &= P(S_{j+1}=A)=P(A),\\
P(S_j=B) &= P(S_{j+1}=B)=P(B), \quad \text{and}\\
P(S_j=C) &= P(S_{j+1}=C)=P(C).
\end{aligned}
$$

Then

$$
\begin{aligned}
P(A) &= P(B)P(A|B)+P(A)P(A|A)\\
&= P(B)+P(A)(25/64).\\
P(B) &= P(C)P(B|C)+P(A)P(B|A)\\
&= P(C)+P(A)(24/64).\\
P(C) &= P(A)P(C|A)=P(A)(15/64).
\end{aligned}
$$

$$P(A)+P(B)+P(C)=1.$$

From above equations, it can be obtained that

$$
\begin{aligned}
P(A) &= 16/27 \text{ and}\\
P(u_{0,j}=u_{1,j}=0) &= P(A)\\
&= 16/27.
\end{aligned}
$$

Hence WTT-JSF-3 holds. **Q.E.D.**

5 Conclusion

Combining the ideas of Solinas and Ciet, a new representation form, WTT-JSF, is proposed in this paper. Mathematical analysis shows that the length of WTT-JSF is approximately the same as that of RTNAF_3 and its density is 11/27. In the computation of $aP+bQ$ in elliptic curve cryptography, if aP and bQ are calculated separately, about $2m$ times of τ operations and $m/2$ additions are needed. If the same position of $\text{RTNAF}_3(a)$ and $\text{RTNAF}_3(b)$ is regarded as one column, then m times of τ operations and $7m/16$ additions are required, together with 4 extra storages. However, with the same number of storages and approximately the same number of τ operations, there are only $11m/27$ additions involved when our method is used. This shows that the WTT-JSF representation leads to a reduction in the computational complexity over other forms. One can also extend WTT-JSF from window 3 to any window size. However, it is not beneficial as 2^{w-1} extra storages are required but only a very limited number of additions are saved.

Acknowledgements

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 121305).

References

[1] ANSI X9.62-1998, *Public key cryptography for the financial industry: the elliptic curve digital signature algorithm (ECDSA)*.

[2] M. Ciet, T. Lange, F. Sica, and J. Quisquater, "Improved algorithms for efficient arithmetic on elliptic curves using fast endomorphisms," in *EUROCRYPT'2003*, LNCS 2656, pp. 388–400, Springer-Verlag, 2003.

[3] J. Crowie, B. Dodson, R. Elkenbracht-Huizing, A. Lenstras, P. Montgomery, and J. Zayer, "A world wide number field sieve factoring record: on to 512 bits," in *ASIACRYPT'96*, LNCS 1163, pp. 382–394, Springer-Verlag, 1996.

[4] D. Gordon, "Discrete logarithms in $GF(p)$ using the number field sieve," *SIAM Journal on Discrete Mathematics*, vol. 6, pp. 124–138, 1993.

[5] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.
[6] N. Koblitz, "CM curves with good cryptographic properties," in *Crypto'91*, LNCS 576, pp. 279–287, Springer-Verlag, 1992.
[7] V. S. Miller, "Use of elliptic curves in cryptography," in *CRYPTO'85*, LNCS 218, pp. 417–426, Springer-Verlag, 1986.
[8] R. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120–126, 1978.
[9] J. Solinas, "Efficient arithmetic on Koblitz curves," *Designs, Codes, and Cryptography*, vol. 19, pp. 195–249, 2000.
[10] J. A. Solinas, *Low-weight Binary Representations for Pairs of Integers*, Technical Report CORR 2001-41, CACR, available at: www.cacr.math.uwaterloo.ca/techreports/2001/corr2001-41.ps, 2001.
[11] S. A. Vanstone, "Next generation security for wireless: elliptic curve cryptography," *Computers & Security*, vol. 22, no. 5, pp. 412–415, 2003.

Yong Ding was born in Chongqing in June 1975. He graduated with a B.E degree from Dept. of Mathematics, Sichuan University, China, in 1998. He received the M.S degree and the PhD degree in Xidian University, China, in 2003 and 2005, respectively. In 2004, he was a Research Associate in Dept. of Computer Engineering and Information Technology, City University of Hong Kong. He is currently an Assistant Professor in Dept. of Computer Science and Mathematics, Guilin Institute of Electronic Technology. His research interests are cryptography and network security.

Kwok-Wo Wong was born in Hong Kong. He graduated with a BSc(EE) degree from The Chinese University of Hong Kong and a PhD degree from City University of Hong Kong. He is currently an Associate Professor in Department of Electronic Engineering, City University of Hong Kong. His current research interests include chaos, cryptography and neural networks. He has published over 70 papers in 25 international mathematics, physics and engineering journals, in the fields of nonlinear dynamics, cryptography, neural networks and optics. Dr. Wong is a Senior Member of The Institute of Electrical and Electronic Engineers (IEEE). He is also a Chartered Engineer and a Member of The Institution of Electrical Engineers (IEE).

Yumin Wang was born in Beijing on Feb 18, 1936. He received the B.E degree from Dept. of Telecommunication Eng. in Xidian University, China, in 1959. Since 1959, he has been teaching in Xidian University, and is currently a Professor as well as a Doctoral Degree Supervisor. From 1979-1981, he was a visiting scholar in Dept. of EE, Hawaii University. He is a Fellow of the Chinese Institute of Communication and the Chinese Institute of Electronics. He serves in the Board of Governors (Preparatory Committee) of the Chinese Institute of Cryptography and also the Committee of Information Theory Society for the Chinese Institute of Electronics. Prof. Wang is a senior member of IEEE. His research interests are communication, information theory, coding and cryptography.