An Improvment of the Elliptic Net Algorithm

Binglong Chen and Chang-An Zhao*

Abstract

In this paper we propose a modified Elliptic Net algorithm to compute pairings. By reducing the number of the intermediate variables which should be updated in the iteration loop of the Elliptic Net algorithm, we speed up the computation of pairings. Experimental results show that the proposed method is about 14% faster than the original Elliptic Net algorithm on certain supersingular elliptic curves with embedding degree two.

Index Terms

Pairing based cryptography, Elliptic curve, Tate pairing, Elliptic Net algorithm.

1 Introduction

Pairings on elliptic curves have become one of the most popular cryptographic primitives in public key cryptography. There are a lot of interesting cryptographic schemes and signatures based on pairings. An excellent survey of pairing-based applications can be found in [1]. Since the computation of the pairings are the most expensive operation in pairing-based cryptosystems, reducing the computational complexity of pairing computations has been an important issue in pairing-based cryptography. Since then, there are much research devoted to the efficient computation of pairings. We refer the reader to [2], [3] for more information about pairings. To date, there are many variants that have been proposed for efficiency reasons based on the Tate pairing, such as Duursma-Lee [4], η_T [5], (optimized) ate [6], [7], ate$_i$ [8], R-ate [9], pairing lattice [10] and optimal [11] pairings. On the other hand, there are also a few pairing variants constructed via the Weil pairing, such as the eil pairing [10], the omega pairing [12] and the improved self-pairings [13], [14], [15].

There only exist two polynomial-time algorithms to compute the pairings till now. One is Miller's algorithm [16], [17] and the other is the Elliptic Net algorithm [18]. A lot of avenues of optimizing Miller's algorithm have been investigated, and now it takes only a few milliseconds to compute pairings by using the optimized Miller's algorithm.

However, all of optimizations of Miller's algorithm can be seldom applied directly in speeding up the Elliptic Net algorithm except simplifying the final exponentiation. There is few work to improve the efficiency of the Elliptic Net algorithm compared with Miller's algorithm. How to compute the ate-like pairings based on the Elliptic Net algorithm has been investigated in [19], [20]. The concept of elliptic nets has been generalized to hyper-elliptic curves for computing hyper-pairings in [21], [22].

Our goal in this paper is to shorten the theoretical gap in the studying of the Elliptic Net algorithm, and to this end we propose several efficient techniques for the arithmetic operations involved as follows.

- Observe that in many practical setting, when computing the evaluation of rational functions $f_{r,P}(Q)$, the concrete parameter r will be chosen reasonably sparse(a good survey of pairing-friendly curves, refer to [23]). This means that the $Double$ function will be more frequent than the $DoubleAdd$ one in the whole iteration loop of the Elliptic Net algorithm. Inspired by the work of Rachel Shipsey [24], we can reduce the number of the intermediate

- *B. Chen and C.-A. Zhao are with the School of Mathematics and Computational Science, School of Mathematics and Computational Science, Sun Yat-sen University, Guangzhou 510275, P.R.China.*
  ** Corresponding author.*
  E-mail: {mcscbl, zhaochan3}@mail.sysu.edu.cn

*Manuscript received *********; revised *******.*

variables in the original Elliptic Net algorithm proposed by Stange [18]. This saves four multiplications and one squaring in the each doubling step compared to the original Elliptic Net algorithm. Although the $DoubleAdd($ or $DoubleSubtraction)$ step is somewhat slower than the previous one, this disadvantage is amply mitigated by the proposed doubling because of the sparse form of r.

- It is known that an integer has a non-adjacent form (NAF) representation. In this representation, the density of the number of non-zero digits in NAF representation will be approximately $1/3$ on average, while it is $1/2$ in the case of binary representation. This leads us to give a modified Elliptic Net algorithm in NAF form.
- When computing the pairing evaluation $f_{r,P}(Q)$ where P is a rational point on an elliptic curve over a finite field $\mathbb{F}_q$, we can choose the prime power q satisfying $\gcd(q-1,3)=1$. Then we can change the value $W(2,0)$ to 1 by using the equivalence of elliptic nets as discussed in [18]. This makes the saving of four multiplications in each iteration loop practically possible.

Efficiency analysis indicates that all of the above optimizations can speed up the Elliptic Net algorithm indeed. It is shown that the presented method can be more efficient than the previously known Elliptic Net algorithm under the condition the density of non-zero digits of r is less than 0.44 and the cost of one inverse is about equal to that of ten multiplications. In addition, experimental results show that the proposed algorithm is about 14% faster than the original Elliptic Net algorithm [18] on certain supersingular elliptic curves with $k=2$ at 128-bit security level. To the best of our knowledge, such a significant improvement of the Elliptic Net algorithm has not been appeared earlier, though it is slower than the state-of-the-art Miller's algorithm.

The rest of the paper is organized as follows. In Section 2, we give an overview of the Tate pairing and the Elliptic Net algorithm. In Section 3, we derive the new $Double$ and $DoubleAdd$(or $DoubleSubtraction$) steps in the Elliptic Net algorithm, and analyze operation counts for one typical iteration. Section 4 provides the efficiency analysis and some detailed suggestions in pratical implementations. Finally, we draw our conclusion in Section 5.

2 PRELIMINARIES

In this section we first recall the definition of the Tate pairing, and the Elliptic Net algorithm to compute it.

2.1 Pairings

Let $\mathbb{F}_q$ be a finite field with $q=p^m$ elements, where p is a prime. Let E be an elliptic curve defined over $\mathbb{F}_q$, and let ∞ be the point at infinity. Let r be a prime such that r divides $\#E(\mathbb{F}_q)$, where $\#E(\mathbb{F}_q)$ denotes the order of $E(\mathbb{F}_q)$. Assume that r^2 does not divide q^k-1 and k is greater than 1, where k is the embedding degree. We denote by $E[r]$ the r-torsion group of E. Let $P\in E[r]$ and $Q\in E(\mathbb{F}_{q^k})$. Let D_P be a degree zero divisor which is equivalent to $(P)-(\infty)$. For every integer i and point P, let $f_{i,P}$ be a rational function such that $(f_{i,P})=i(P)-(iP)-(i-1)(\infty)$. In particular, $(f_{r,P})=rD_P$. Let μ_r be the r-th roots of unity in $\mathbb{F}_{q^k}$. Then the reduced Tate pairing [25] is defined as follows

$$e: E[r]\times E(\mathbb{F}_{q^k})\to\mu_r,$$
$$e(P,Q)=f_{r,P}(Q)^{\frac{q^k-1}{r}}.$$

Note that most of the pairing variants are constructed from the Tate pairing for efficiency reasons, such as the eta [5], ate [6], R-ate [9] and optimal [11] pairings. All of Tate-like and ate-like pairings can be calculated by the Elliptic Net algorithm [18], [19], [20].

2.2 Elliptic net algorithm

Stange first gave the definition of elliptic nets and proposed a polynomial time algorithm to compute the pairings in 2007 [18]. In fact, elliptic nets are a generalization of elliptic divisibility sequences first studied by Ward [26]. An elliptic net

is a map W from a finitely generated free abelian group A to an integral domain such that the following recurrence holds for all p, q, r, $s \in A$:

$$\begin{aligned} &W(p+q+s)W(p-q)W(r+s)W(r) \\ &+W(q+r+s)W(q-r)W(p+s)W(p) \\ &+W(r+p+s)W(r-p)W(q+s)W(q)=0. \end{aligned} \tag{1}$$

Theorem 1. [18] Let E be an elliptic curve over a finite field K, r a positive integer, $P \in E(K)[r]$ and $Q \in E(K)$. Denote by D_Q the divisor equivalent to $(Q)-(\infty)$. If $W_{P,Q}$ is the elliptic net associated to $E,\ P,\ Q$, then we have

$$f_{r,P}(D_Q)=\frac{W_{P,Q}(r+1,1)W_{P,Q}(1,0)}{W_{P,Q}(r+1,0)W_{P,Q}(1,1)}.$$

From the properties of elliptic nets, one can obtain the $Double$ and $DoubleAdd$ formulas which can be applied in the Elliptic Net algorithm. For simplicity, we abbreviate $W_{P,Q}(m,n)$ as $W(m,n)$ from now on. According to the results of [18], it is essential to update *a block centered on i* consisting of a first vector of $eight$ consecutive terms centered on term $W(i,0)$ and a second vector of three consecutive terms centered on term $W(i,1)$.

Assume that $W(1,0)=W(0,1)=1$. Then all of $W(n,0)$ can be updated by the following two formulas,

$$W(2i-1,0)=W(i+1,0)W(i-1,0)^3-W(i-2,0)W(i,0)^3, \tag{2}$$

and

$$W(2i,0)=(W(i,0)W(i+2,0)W(i-1,0)^2-W(i,0)W(i-2,0)W(i+1,0)^2)/W(2,0). \tag{3}$$

For obtaining the pairing values, one also requires the following important formula which can be used for computing the $W(i,1)$ terms,

$$W(2i-1,1)=(W(i+1,1)W(i-1,1)W(i-1,0)^2-W(i,0)W(i-2,0)W(i,1)^2)/W(1,1), \tag{4}$$

$$W(2i,1)=W(i-1,1)W(i+1,1)W(i,0)^2-W(i-1,0)W(i+1,0)W(i,1)^2, \tag{5}$$

$$W(2i+1,1)=(W(i-1,1)W(i+1,1)W(i+1,0)^2-W(i,0)W(i+2,0)W(i,1)^2)/W(-1,1), \tag{6}$$

and

$$W(2i+2,1)=(W(i+1,0)W(i+3,0)W(i,1)^2-W(i-1,1)W(i+1,1)W(i+2,0)^2)/W(2,-1). \tag{7}$$

On the basis of the above formulas, one can use the Elliptic Net algorithm to compute the pairings. We remark that the values of $W(2,0)$, $W(1,1)$, $W(-1,1)$ and $W(2,-1)$ heavily affect the efficiency of the Elliptic Net algorithm. Under certain circumstances, some of these values can be changed to one for accelerating pairing computations by using the equivalence of elliptic nets [18].

3 DOUBLE AND DOUBLEADD (OR DOUBLESUBTRACTION) STEPS IN THE ELLIPTIC NET ALGORITHM

In this section, we investigate how to modify the basic $Double$ and $DoubleAdd$ steps in the Elliptic Net algorithm by Stange [18], which leads to a further improvement in certain special cases.

In the practical implementations of pairing-based protocols and schemes, one often chooses the parameter r with a low Hamming weight for efficiency, which defines the rational function $f_{r,P}$. Also, many families of pairing-friendly curves can be constructed flexibly according to the requirements of the concrete applications indeed [23]. For example, one can generate a supersingular curve over large prime fields with embedding degree two easily. Note that this curve has been recommended in the IEEE Standard draft for Identity-Based cryptographic techniques using pairings [27]. A resulting curve defined over $\mathbb{F}_q$ has its order divisible by a large prime r which has a low Hamming weight. In addition, the popular B-N curves can also be constructed with fixed coefficients with a sparse form [28], [29]. When the parameter r that is used for defining the corresponding Miller function $f_{r,P}$ has a low Hamming weight, we will show that the complexity of the elliptic net algorithm can be reduced by using the proposed method.

3.1 New double steps

In the original Elliptic Net algorithm proposed by Stange [18], one needs a block centered on i to consist of a first vector of $eight$ consecutive terms. There are two functions $Double(V)$ and $DoubleAdd(V)$ for updating the block. Observe that if the integer r has a low Hamming weight this indicates that the $Double(V)$ function can be used more frequently than the $DoubleAdd(V)$ one in the whole iteration loop. This means that faster $Double$ steps may lead to the improvement of pairing computations although slower $DoubleAdd$ or $DoubleSubtraction$ ones are employed.

In the thesis of Rachel Shipsey [24], it has been pointed out that the first vector centered at i with $seven$ consecutive terms is sufficient for the doubling steps by the duplication formulas in the computation of elliptic divisibility sequences. This indicates that the first vector with $seven$ terms may be sufficient for designing the whole $Double(V)$ function in pairing computation.

The remaining task is to update the second vector with $three$ consecutive terms $W(\lambda,1)_{2i-1\leq\lambda\leq 2i+1}$ given the values of $W(\lambda,0)_{i-3\leq\lambda\leq i+3}$ and $W(\lambda,1)_{i-1\leq\lambda\leq i+1}$. Observe that updating the second vector only involves the values of $W(\lambda,0)_{i-2\leq\lambda\leq i+2}$ and $W(\lambda,1)_{i-1\leq\lambda\leq i+1}$ by using Equations (2)-(7). Therefore, we can design a new $Double$ function with 10 terms.

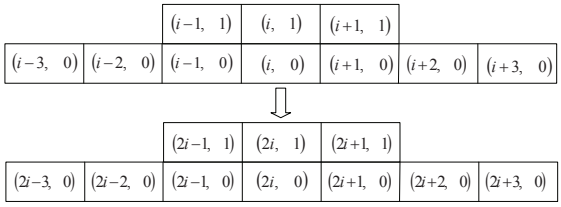

Fig. 1. Updating the block centered at i to $2i$

Consequently, we can discard the 8-th term $W(i+4,0)$ of the first vector in the original Elliptic Net algorithm [18] and then save the cost of updating it. For $P\in E(\mathbb{F}_q)$ and $Q\in E(\mathbb{F}_{q^k})$(This corresponds to the Miller-lite or Tate-like case), we can give a new $Double(V)$ alogrithm as shown in Algorithm 3.1. When the points P and Q are swapped for the ate-like case, it is not hard to modify Algorithm 3.1 for computing the pairings. For simplicity, we omit this case here.

When counting field operations, we use S and M to respectively represent the cost of a squaring and a multiplication in the ground field $\mathbb{F}_q$. Let S_s and M_s denote the costs of a squaring and a multiplication in some extension field $\mathbb{F}_{q^s}$ respectively. Assume that the cost of multiplying an element of $\mathbb{F}_q$ by one of $\mathbb{F}_{q^s}$ takes sM. Then we give an estimate of each step in the new $Double(V)$ algorithm as shown in Algorithm 3.1. According to the discussion of [18], the value $W(-1,1)$ can be adjusted to 1 but $W(1,1)$ will be equal to x_P-x_Q, where x_P and x_Q denote the x-coordinates of point P and Q respectively. In summary, the total cost for the new $Double(V)$ algorithm is $5S+(22+6k)M+S_k+2M_k$ in general.

Note that the embedding degree k could be chosen to be even in most cases. This implies that x_P-x_Q can be contained in the subfield $\mathbb{F}_{q^{k/2}}$ exactly and then the total cost can be reduced to $5S+(22+6k)M+1S_k+\frac{3}{2}M_k$ as discussed in [18], [30]. Finally, we summarize all discussions into Table 1. We see that the proposed $Double(V)$ algorithm will save $1S+4M$ compared to the previous results from Table 1. This gain is from neglecting the computation of the term $W(2i+4,0)$ actually.

TABLE 1
Cost of the $Double(V)$ algorithm for the different methods

Method	**Operation Count**
Elliptic Net algorithm [18]	$6S+(26+6k)M+S_k+\frac{3}{2}M_k$
This work	$5S+(22+6k)M+S_k+\frac{3}{2}M_k$

Algorithm 3.1 Double and DoubleAdd/DoubleSubtraction Algorithm

Input: Block V centered at i in which the first vector has $seven$ terms and the second vector has $three$ terms. Array S and P with 5 elements. $W(1,0) = W(0,1) = 1$, $\alpha = W(2,0)^{-1}$, $\beta = W(-1,1)^{-1}$, $\gamma_1 = W(2,-1)^{-1}$, $\gamma_2 = W(2,1)^{-1}$, $\delta = W(1,1)^{-1}$, $w_2 = W(2,0)^2$, $w_{13} = W(1,0) \cdot W(3,0)$ and $flag \in \{-1,0,1\}$

Output: Block centered at $2i$ if $flag = 0$, centered at $2i+1$ if $flag = 1$ and centered at $2i-1$ if $flag = -1$

1. $S_0 \leftarrow V[2,2]^2$; $P_0 \leftarrow V[2,1] * V[2,3]$; $//1S_k + 1M_k$

2. for i := 1 to 5 do

   $S[i] \leftarrow V[1,i+1]^2$;

   $P[i] \leftarrow V[1,i] * V[1,i+2]$; $// \ 5 \cdot (1S + 1M)$

   end for;

3. if $flag \ eq \ 0$ then

   for $i \leftarrow 1$ to 3 do

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

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

   end for;

   $V[1,7] \leftarrow S[4] * P[5] - S[5] * P[4]$; $//2M$

   $V[2,1] \leftarrow (S[2] * P_0 - P[2] * S_0) * \delta$; $//2kM + M_k$

   $V[2,2] \leftarrow S[3] * P_0 - P[3] * S_0$; $//2kM$

   $V[2,3] \leftarrow (S[4] * P_0 - P[4] * S_0) * \beta$; $//2kM$

   else if $flag \ eq \ 1$ then

   for $i \leftarrow 1$ to 3 do

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

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

   end for;

   $t_1 \leftarrow V[1,4] * V[1,6]$; $\quad t_2 \leftarrow V[1,5]^2$;

   $V[1,7] \leftarrow (t_1 * w_2 - t_2 * w_{13}) / V[1,3]$; $//1I + 3M$

   $V[2,1] \leftarrow S[3] * P_0 - P[3] * S_0$; $//2kM$

   $V[2,2] \leftarrow (S[4] * P_0 - P[4] * S_0) * \beta$; $//2kM$

   $V[2,3] \leftarrow (P[5] * S_0 - S[5] * P_0) * \gamma_1$; $//2kM + 1M_k$

   else

   for $i \leftarrow 1$ to 3 do

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

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

   end for;

   $t_1 \leftarrow V[1,2] * V[1,4]$; $\quad t_2 \leftarrow V[1,3]^2$;

   $V[1,1] \leftarrow (t_1 * w_2 - t_2 * w_{13}) / V[1,5]$; $//1I + 3M$

   $V[2,1] \leftarrow (S[1] * P_0 - P[1] * S_0) * \gamma_2$; $//2kM + 1M_k$

   $V[2,2] \leftarrow (S[2] * P_0 - P[2] * S_0) * \delta$; $//2kM + M_k$

   $V[2,3] \leftarrow S[3] * P_0 - P[3] * S_0$; $//2kM$

   end if;

4. return V

3.2 New DoubleAdd/DoubleSubtraction steps

It is well-known that an integer has a non-adjacent form (NAF) representation. In this representation, the density of the number of non-zero digits will be approximately one-third on average. Therefore, the non-adjacent form (NAF) representation of an integer has been suggested for computing the pairings or scalar multiplication in efficiency. This leads us to consider a modified Elliptic Net algorithm when the parameter r is represented in NAF form. The remaining task is to design two $DoubleAdd(V)$ and $DoubleSubtraction(V)$ functions if the first vector only has 7 consecutive terms centered at i.

3.2.1 DoubleAdd function

Firstly, we consider how to update the next block V centered at $2i+1$ from the current block centered at i which has the first vector with 7 terms. In this case, the non-zero digit is 1.

We now restrict that the first vector only has 7 terms $W(\lambda,0)_{i-3\leq\lambda\leq i+3}$ centered at i. It follows from the duplication formula that $W(\lambda,0)_{2i-3\leq\lambda\leq 2i+3}$ can be updated under such an assumption. However, we hope that the updated first vector should be centered at $2i+1$. This indicates that $W(2i-3,0)$ should be omitted but $W(2i+4)$ must be calculated in the new $DoubleAdd(V)$ algorithm.

It is known that the sequence $\{W(i,0)\}$ is an elliptic net of rank one [18](or equivalently, an elliptic divisibility sequence [24], [26]) and then satisfies the following well-known recursive formula,

$$W(h+2,0)W(h-2,0)=W(h+1,0)W(h-1,0)W(2,0)^2-W(3,0)W(1,0)W(h,0)^2. \tag{8}$$

Replacing h by $2i+2$ yields the following equality for computing $W(2i+4,0)$,

$$W(2i+4,0)=\frac{W(2i+3,0)W(2i+1,0)W(2,0)^2-W(3,0)W(1,0)W(2i+2,0)^2}{W(2i,0)}. \tag{9}$$

By Equation (9), we see that $W(2i+4,0)$ can be computed provided that the values $W(\lambda,0)_{2i\leq\lambda\leq 2i+3}$ are known. However, all of $W(\lambda,0)_{2i\leq\lambda\leq 2i+3}$ can be obtained given the values $W(\lambda,0)_{i-3\leq\lambda\leq i+3}$ by Equations (2) and (3).

Since the second vector centered at $2i+1$ can be updated provided that all of $W(\lambda,0)_{i-1\leq\lambda\leq i+3}$ and $W(\lambda,1)_{i-1\leq\lambda\leq i+1}$ are known, it is reasonable that the first vector of the block only has $seven$ terms for constructing the $DoubleAdd(V)$ algorithm as shown Algorithm 3.1.

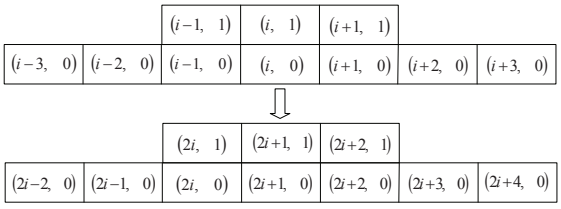

Fig. 2. Updating the block centered at i to $2i+1$

It is obvious that the proposed $DoubleAdd(V)$ step will be more costly than the original $DoubleAdd$ one(see Algorithm 2 of [18]) since the inverse involves in the updating process. However, the new algorithm seems to be attractive possibly under the condition that the parameter r has a low Hamming weight. We will give a detailed efficiency consideration in the later section.

We now estimate the cost of the new $DoubleAdd(V)$ algorithm in the Tate-like case. Compared to the $Double(V)$ algorithm in the previous section, discarding the computation of $W(2i-3,0)$ will save $2M$. Now we consider the cost of computing $W(2i+4,0)$. Note that $W(2,0)^2$ and $W(3,0)W(1,0)$ can be precomputed. Therefore we neglect the cost of computing them. We also remark that the cost of computing $W(2i+3,0)W(2i+1,0)$ and $W(2i+2,0)^2$ can be ignored since both of them must be calculated in the next iteration loop for updating the first vector. This means that if $t1$ and $t2$ in Algorithm 3.1 can be cached then both can be updated directly to $S[4]$ and $P[4]$ in the next loop. Therefore, the computational cost of computing $W(2i+4,0)$ will be $1I+3M$. This gives that the total cost of the presented $DoubleAdd(V)$ step will be $5S+(23+6k)M+I+S_k+2M_k$ in general. We list the above efficiency analysis and the previous results for comparison in Table 2.

TABLE 2
Cost of the DoubleAdd(V) algorithm for the different cases

Method	Operation count
Elliptic Net algorithm [18]	$6S + (26 + 6k)M + S_k + 2M_k$
This work	$5S + (23 + 6k)M + I + S_k + 2M_k$

3.2.2 DoubleSubtraction function

We now consider the case of the non-zero digit -1. This means that we need consider how to move the center of the next block to $2i - 1$ provided that the current block centered at i with the first vector having $seven$ terms is known.

According to the above discussions about $DoubleAdd$ steps, we can discard the term $W(2i + 3, 0)$ and calculate the term $W(2i - 4, 0)$ by symmetry. In a similar manner, replacing h by $2i - 2$ in Equation (8) yields the following equality,

$$W(2i-4,0) = \frac{W(2i-3,0)W(2i-1,0)W(2,0)^2 - W(2i-2,0)^2W(3,0)W(1,0)}{W(2i,0)}.$$

In a similar manner, we can update all the terms $W(\lambda, 0)_{2i-4\leq\lambda\leq 2i+2}$ in the first vector.

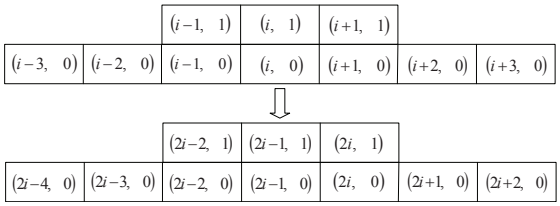

Fig. 3. Updating the block centered at i to $2i - 1$

We now consider how to update the terms $W(\lambda, 1)_{2i-2\leq\lambda\leq 2i}$ in the second vector if the terms $W(\lambda, 1)_{i-1\leq\lambda\leq i+1}$ are known. Observe that the term $W(2i - 1, 1)$ and $W(2i, 1)$ can be computed by Equations (4) and (5) in this case. The remaining task is to compute the term $W(2i-2, 1)$ by using the recurrence relation (1). Replacing p, q, r, and s by $(i-2, 0)$, $(i, 1)$, $(1, 0)$ and $(0, 0)$ gives the following formula,

$$W(2i-2,1) = (W(i-1,1)W(i+1,1)W(i-2,0)^2 - W(i-1,0)W(i-3,0)W(i,1)^2)/W(2,1).$$

Therefore, it is reasonable to design a $DoubleSubtraction$ function on the basis of a block which has the first vector with $seven$ terms and the second vector with $three$ terms. We summarize the results into Algorithm 3.1. In addition, we remark that the cost of the $DoubleSubtraction$ step is the same as that of the $DoubleAdd$ one. On the basis of Algorithm 3.1 which consists of the basic $Double$ and $DoubleAdd/DoubleSubtraction$, it is easy to give a modified Elliptic Net algorithm shown as Algorithm 3.2. Note that the initial index of every vector in a block will be one, not $zero$ in our algorithm.

4 Efficiency comparisons and implementation results

In this section, we will analyze the whole operation count of computing the pairings by using the proposed Double and DoubleAdd steps, and compare it with the original Elliptic Net algorithm given by Stange in [18].

4.1 Efficiency comparison

Assume that $P \in E(\mathbb{F}_q)$ and $Q \in E(\mathbb{F}_{q^k})$. Now we consider the efficiency of computing $f_{r,P}(Q)$ by using the proposed method and the previous one [18], respectively. Let N be the bit length of r. Denote by ρ be the density of non-zero digits of the integer r in NAF representation. On the basis of the above assumption, we can estimate the cost of of computing the

Algorithm 3.2 Elliptic Net Algorithm in NAF form

Input: Points $P \in E(\mathbb{F}_q)$ and $Q \in E(\mathbb{F}_{q^k})$. Initial terms $a = W(2,0)$, $b = W(3,0)$, $c = W(4,0)$, $d = W(2,1)$, $e = W(-1,1)$, $f = W(2,-1)$, $g = W(1,1)$, $h = W(2,1)$ of an elliptic net satisfying $W(1,0) = W(0,1) = 1$ and an integer $m = (d_l d_{l-1} \cdots d_0)_2$ is represented in NAF form with $d_l = 1$ and $d_i \in \{0, \pm 1\}$ for $0 \leq i \leq l-2$.

Output: $W(m,0)$ and $W(m,1)$

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

Tate-like pairing $f_{r,P}(Q)$ by using the two different methods and determine the critical value of the density ρ. The total cost of the modified Elliptic Net algorithm is

$$\begin{aligned}&\rho N(5S + (23+6k)M + I + S_k + 2M_k) + (1-\rho)N(5S + (22+6k)M + S_k + \frac{3}{2}M_k)\\ =&N(5S + (22+6k)M + S_k + \frac{3}{2}M_k) + \rho N(M + I + \frac{1}{2}M_k).\end{aligned}$$

Similarly, the total cost of Stange's algorithm [18] is

$$\begin{aligned}&\rho N(6S + (26+6k)M + S_k + 2M_k) + (1-\rho)N(6S + (26+6k)M + S_k + \frac{3}{2}M_k)\\ =&N(6S + (26+6k)M + S_k + \frac{3}{2}M_k) + \frac{1}{2}\rho N M_k.\end{aligned}$$

Therefore, the proposed algorithm will be faster than the original Elliptic Net algorithm by Stange [18] under the condition that

$$\begin{aligned}&N(6S + (26+6k)M + S_k + \frac{3}{2}M_k) + \frac{1}{2}\rho N M_k\\ -&N(5S + (22+6k)M + S_k + \frac{3}{2}M_k) - \rho N(M + I + \frac{1}{2}M_k)\\ =&N(S + 4M) - \rho N(M + I) > 0.\end{aligned}$$

i.e.,

$$\rho < \frac{S + 4M}{M + I}.$$

Assume that 1S = 0.8M. In table 3, we present the maximal value of the density ρ such that the proposed algorithm will be more efficient than the previous results for computing the Tate-like(or Miller-lite) pairings. It is well-known that for a randomly chosen n-bit integer the density of non-zero digits will be $1/3$ on average by using NAF representation. This indicates that the proposed method will be most probably faster than the original Elliptic Net algorithm under the assumption of $I = 10M$.

4.2 Implementation results

In this section, experimental results are given for computing the Tate pairing using the the different techniques. There are many choices of pairing friendly curves for implementing the pairing. The chosen curve here will be supersingular elliptic curves over large prime fields with embedding degree $k = 2$, which are recommended in some industry standards(or

TABLE 3
Maximal value of the density ρ for the proposed method

Density	I = 10M	I = 20M	I = 30M
ρ	0.44	0.23	0.15

drafts), such as IEEE P1363.3 draft [27] and RFC 5091 [31]. Curves of this form adopt many advantages that have been clarified in [32]. In particular, the order of the subgroup of points defined over the base field can be chosen flexibly.

Our running environment specifications are listed as follows: Ubuntu Kylin 14.04 64bits, Core i5-4670 CPU 3.40GHz$\times$ 4, and memory, 8GB. The code was written in Magma [33]. The concrete parameters for the supersingular elliptic curve with embedding degree 2 are given as follows.

- $r = 2^{255} + 2^{41} + 1$; (256 bits)
- $p = 12 \cdot (2^{1280} + 2^{31} + 2^{15}) \cdot r - 1$; (1539 bits)
- $\mathbb{F}_{p^2} = \mathbb{F}_p[i]/(i^2 + 1)$;
- $E : y^2 = x^3 - 3x$ over $\mathbb{F}_p$.

Notice that the prime p satisfies $\gcd(p-1, 3) = 1$. This implies that every nonzero element in $\mathbb{F}_p$ has a cube root. In particular, one can find an element η in $\mathbb{F}_p$ such that $\eta^3 = W(2,0)^{-1}$. By using the equivalence of the elliptic nets, we can change $W(2,0)$ to 1 and then save four $\mathbb{F}_p$ multiplications in each iteration as discussed in [18]. Assume that $1M_2 = 3M$ and $1S_2 = 2M$. Here we assume that $1I = 10M$ and $1S = 0.8M$. In the whole comparison, we neglect the pre-computations since the point P can be fixed for many pairing-based cryptographic protocols.

The overall cost of the iteration loop by using the proposed method on the above curve will be

$$\begin{aligned} &253 \cdot (5S + (18 + 6k)M + S_k + \frac{3}{2}M_k) + 2 \cdot (5S + (19 + 6k)M + I + S_k + 2M_k) \\ =&253 \cdot 40.5M + 2 \cdot 53M = 10352.5M. \end{aligned}$$

In a similar manner, the cost of the whole iteration loop in Stange's algorithm [18] will be

$$\begin{aligned} &253 \cdot (6S + (22 + 6k)M + S_k + \frac{3}{2}M_k) + 2 \cdot (6S + (22 + 6k)M + S_k + 2M_k) \\ =&253 \cdot 45.3M + 2 \cdot 46.8M = 11554.5M. \end{aligned}$$

From a performance point of view it is essential to compare the running time of the iteration loop with the orginal Elliptic Net algorithm given by Stange in [18]. We also give the cost of computing $f_{r,P}(Q)$ using Miller's algorithm in projective coordinates for benchmarks. In Miller's algorithm, the computational cost of each doubling step is $8M + 4S + M_k + S_k$ and that of each addition step is $11M + 3S + M_k$, respectively [30], [34]. The overall operation count of Miller's algorithm will be about $4164M$. Table 4 summarizes the above estimation of the computational cost and some running times for calculating the pairings at 128-bit security level. All of the timings are shown in milliseconds. Our experimental results indicate that the presented method in this paper is about 14% faster than the previous Elliptic Net algorithm [18]. This shows that the proposed method will efficiently increase performance under the condition that the corresponding parameter r has a low Hamming weight. However, it is still substantially slower than Miller's algorithm in projective coordinate systems. This indicates that Miller's algorithm is still a valid candidate for practical pairing-based implementations and more optimizations about the Elliptic Net algorithm should be required.

5 Conclusion

In this paper, we showed how using the property of elliptic nets to design new doubling and addition steps in the Elliptic Net algorithm can reduce the cost of the running time when computing the pairings. Our experimental results indicated that under the condition that the loop parameter r has low Hamming weight, one can attain higher performance using the proposed method than using the previous Elliptic Net algorithm. While the modified Elliptic Net algorithm is somewhat

TABLE 4
Cost of computing $f_{r,P}(Q)$ by the different methods-128 security level

Method	Operation Count	Time(ms)
Stange's algorithm [18]	$11554.5M$	37.8
This work	$10352.5M$	33.2
Miller's algorithm	$4164M$	14.9

expensive compared to Miller's algorithm in projective coordinate systems, it can be still implemented quite efficiently on modern personal computers. We believe that this work will lead to more developments of the Elliptic Net algorithm in future.

ACKNOWLEDGMENTS

The work of Binglong Chen is partially supported by the National Natural Science Foundation of China under Grant No. 11025107. The work of Chang-An Zhao is partially supported by the National Natural Science Foundation of China under Grant No. 61472457.

REFERENCES

[1] K. G. Paterson, *Cryptography from Pairings - Advances in Elliptic Curve Cryptography*, I. F. Blake, G. Seroussi, and N. P. Smart, Eds. Cambridge University Press, 2005.
[2] H. Cohen and G. Frey, Eds., *Handbook of elliptic and hyperelliptic curve cryptography*. CRC Press, 2005.
[3] S. D. Galbraith, *Pairings-Advances in Elliptic Curve Cryptography*, I. F. Blake, G. Seroussi, and N. P. Smart, Eds. Cambridge University Press, 2005.
[4] I. Duursma and H.-S. Lee, "Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$," in *Advances in Cryptology - ASIACRYPT 2003*, ser. Lecture Notes in Computer Science, C.-S. Laih, Ed. Springer Berlin Heidelberg, 2003, vol. 2894, pp. 111–123.
[5] P. S. L. M. Barreto, S. D. Galbraith, C. O'Eigeartaigh, and M. Scott, "Efficient pairing computation on supersingular abelian varieties," *Des. Codes Cryptography*, vol. 42, no. 3, pp. 239–271., 2007.
[6] F. Hess, N. Smart, and F. Vercauteren, "The eta pairing revisited," *IEEE Transactions on Information Theory*, vol. 52, pp. 4595–4602, 2006.
[7] S. Matsuda, N. Kanayama, F. Hess, and E. Okamoto, "Optimised versions of the ate and twisted ate pairings," in *Cryptography and Coding*, ser. Lecture Notes in Computer Science, S. Galbraith, Ed. Springer Berlin / Heidelberg, 2007, vol. 4887, pp. 302–312.
[8] C.-A. Zhao, F. Zhang, and J. Huang, "A note on the ate pairing," *Int. J. Inf. Sec*, vol. 7, no. 6, pp. 379–382, 2008.
[9] E. Lee, H.-S. Lee, and C.-M. Park, "Efficient and generalized pairing computation on abelian varieties," *IEEE Transactions on Information Theory*, vol. 55, no. 4, pp. 1793–1803, 2009.
[10] F. Hess, "Pairing lattices," in *Pairing-Based Cryptography C Pairing 2008*, ser. Lecture Notes in Computer Science, S. Galbraith and K. Paterson, Eds. Springer Berlin Heidelberg, 2008, vol. 5209, pp. 18–38.
[11] F. Vercauteren, "Optimal pairings," *IEEE Transactions on Information Theory*, vol. 56, no. 1, pp. 455–461, 2010.
[12] C.-A. Zhao, D. Xie, F. Zhang, J. Zhang, and B.-L. Chen, "Computing bilinear pairings on elliptic curves with automorphisms," *Des. Codes Cryptography*, vol. 58, no. 1, pp. 35–44, 2011.
[13] C.-A. Zhao, F. Zhang, and D. Xie, "Faster computation of self-pairings," *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 3266 –3272, 2012.
[14] S. D. Galbraith and C.-A. Zhao, "Self-pairings on hyperelliptic curves," *J. Math. Crypt.*, vol. 7, pp. 31 – 42, 2013.
[15] B. Chen and C.-A. Zhao, "Self-pairings on supersingular elliptic curves with embedding degree three," *Finite Fields and Their Applications*, vol. 28, no. 0, pp. 79 – 93, 2014.
[16] V. S. Miller, "Short programs for functions on curves," Aug 26, 1986. [Online]. Available: http://crypto.stanford.edu/miller/miller.ps.
[17] ——, "The Weil pairing, and its efficient calculation," *J. Cryptology*, vol. 17, no. 4, pp. 235–261, 2004.
[18] K. E. Stange, "The Tate pairing via elliptic nets," in *Pairing-Based Cryptography - Pairing 2007*, ser. Lecture Notes in Computer Science, T. Takagi, T. Okamoto, E. Okamoto, and T. Okamoto, Eds. Springer Berlin Heidelberg, 2007, vol. 4575, pp. 329–348.
[19] N. Ogura, N. Kanayama, S. Uchiyama, and E. Okamoto, "Cryptographic pairings based on elliptic nets," in *Advances in Information and Computer Security*, ser. Lecture Notes in Computer Science, T. Iwata and M. Nishigaki, Eds. Springer Berlin Heidelberg, 2011, vol. 7038, pp. 65–78.
[20] C. Tang, D. Ni, M. Xu, B. Guo, and Y. Qi, "Implementing optimized pairings with elliptic nets," *Science China Information Sciences*, vol. 57, no. 5, pp. 1–10, 2014.
[21] Y. Uchida and S. Uchiyama, "The Tate-Lichtenbaum pairing on a hyperelliptic curve via hyperelliptic nets," in *Pairing-Based Cryptography Pairing 2012*, ser. Lecture Notes in Computer Science, M. Abdalla and T. Lange, Eds. Springer Berlin Heidelberg, 2013, vol. 7708, pp. 218–233.

[22] C. Tran, "Formulae for computation of Tate pairing on hyperelliptic curve using hyperelliptic nets," in *Progress in Cryptology AFRICACRYPT 2014*, ser. Lecture Notes in Computer Science, D. Pointcheval and D. Vergnaud, Eds. Springer International Publishing, 2014, vol. 8469, pp. 199–214.

[23] D. Freeman, M. Scott, and E. Teske, "A taxonomy of pairing-friendly elliptic curves," *Journal of Cryptology*, vol. 23, no. 2, pp. 224–280, 2010.

[24] R. Shipsey, "Elliptic divisibility sequences," Ph.D. dissertation, University of London, 2000.

[25] P. Barreto, H. Kim, B. Lynn, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," in *Advances in Cryptology - CRYPTO 2002*, ser. Lecture Notes in Computer Science, M. Yung, Ed. Springer Berlin Heidelberg, 2002, vol. 2442, pp. 354–369.

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

[27] I. P. Committee, "IEEE P1363.3: Identity-based public key cryptography," http://grouper.ieee.org/groups/1363/IBC/index.html, 2013.

[28] P. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *Selected Areas in Cryptography*, ser. Lecture Notes in Computer Science, B. Preneel and S. Tavares, Eds. Springer Berlin Heidelberg, 2006, vol. 3897, pp. 319–331.

[29] M. Shirase, "Barreto-Naehrig curve with fixed coefficient - efficiently constructing pairing-friendly curves -," Cryptology ePrint Archive, Report 2010/134, 2010.

[30] N. Koblitz and A. Menezes, "Pairing-based cryptography at high security levels," in *Cryptography and Coding*, ser. Lecture Notes in Computer Science, N. Smart, Ed. Springer Berlin Heidelberg, 2005, vol. 3796, pp. 13–36.

[31] M. L. Boyen X, "Identity-based cryptography standard (ibcs)(version 1), request for comments (rfc) 5091," 2007.

[32] M. Scott, "Computing the Tate pairing," in *Topics in Cryptology CT-RSA 2005*, ser. Lecture Notes in Computer Science, A. Menezes, Ed. Springer Berlin Heidelberg, 2005, vol. 3376, pp. 293–304.

[33] W. Bosma, J. Cannon, and C. Playoust, "The Magma algebra system. I. The user language," *J. Symbolic Comput.*, vol. 24, no. 3-4, pp. 235–265, 1997.

[34] S. Chatterjee, P. Sarkar, and R. Barua, "Efficient computation of Tate pairing in projective coordinate over general characteristic fields," in *Information Security and Cryptology ICISC 2004*, ser. Lecture Notes in Computer Science, C.-s. Park and S. Chee, Eds. Springer Berlin Heidelberg, 2005, vol. 3506, pp. 168–181.