Journal of Computational and Applied Mathematics 236 (2012) 4336–4343

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Improved Pollard rho method for computing discrete logarithms over finite extension fields☆

Ping Wang*, Fangguo Zhang*

School of Information Science and Technology, Sun Yat-sen University, Guangzhou 510006, China

ARTICLE INFO

Article history:
Received 11 May 2011

Keywords:
Pollard rho method
Normal basis representation
Discrete logarithm

ABSTRACT

It is clear that the distinctive feature of the normal basis representations, namely, the p-th power of an element is just the cyclic shift of its normal basis representation where p is the characteristic of the underlying field, can be used to speed up the computation of discrete logarithms over finite extension fields $\mathbb{F}_{p^m}$. We propose a variant of the Pollard rho method to take advantage of this feature, and achieve the speedup by a factor of $\sqrt{m}$, rather than $\frac{3p-3}{4p-3}\sqrt{m}$, the previous result reported in the literature. Besides the theoretical analysis, we also compare the performances of the new method with the previous algorithm in experiments, and the result confirms our analysis. Due to the MOV reduction, our method can be applied to paring-based cryptosystems over binary or ternary fields.

1. Introduction

One of the most important assumptions in modern cryptography is the hardness of the discrete logarithm problem (DLP). Many popular cryptosystems base their security on DLP. Such cryptosystems are, for example, the Diffie–Hellman key agreement protocol [1], the ElGamal signature and encryption schemes [2], the US governments Digital Signature Algorithm (DSA) [3], the Schnorr signature scheme [4]. Originally, they worked with multiplicative groups of finite prime fields. Once elliptic curve cryptosystems were proposed in [5,6], analogous practical systems based on the DLP in groups of points of elliptic curves over finite fields were designed [7]. Recall the following definition:

Definition 1 (*Discrete Logarithm Problem (DLP)*)**.** Let G be a cyclic group of prime order q and let $g \in G$ be generator of G. Given $g, h \in G$, determine the integer $0 \leq k < q$ such that $h = g^k$.

For DLP on a multiplicative subgroup G of prime order q of finite field $\mathbb{F}_{p^m}$, the index calculus method [8] determines the size of p^m, which is a subexponential time algorithm, while the size of q is set by Pollard's rho method [9]. Further, for pairing-based cryptography, binary and ternary fields are widely used as the base field over which Tate or Weil pairings are defined [10–12]. Due to the MOV reduction [13,14], the hard problems on which pairing-based cryptosystems are built can be transformed into the DLP on extended binary or ternary fields. This increases the significance of the study of DLP on finite extension fields.

For DLP on a subgroup of elliptic curve defined over finite field, Pollard's rho method and its modifications in [15,16] are to date known as the most efficient general algorithms. Oorschot and Wiener [17] showed that the modified Pollard's rho method can be parallelized with linear speedup.

☆ This work is supported by the National Natural Science Foundation of China (No. 61070168).

* Corresponding authors.
E-mail addresses: wangp62@mail2.sysu.edu.cn (P. Wang), isszhfg@mail.sysu.edu.cn (F. Zhang).

0377-0427/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2012.03.019

Pollard's rho method is a randomized algorithm for computing discrete logarithms. Generally, an iteration function $F : G \to G$ is used to define a pseudo-random sequence Y_i by $Y_{i+1} = F(Y_i)$ for $i = 0, 1, 2, \ldots$, with some initial value Y_0. The sequence $Y_0, Y_1, Y_2, \ldots$ represents a walk in the group G. Because the order of the group is finite, the sequence will ultimately reach an element that has occurred before. This is called a *collision* or a *match*. The advantage of this method is that the space requirements are small if one uses a clever method of detecting a collision.

The basic assumption for the analysis of the expected run time of the rho method is that the walk Y_i behaves as a random walk. By this we mean that the iteration function $F : G \to G$ is a random mapping in the sense that for any $Y_i \in G$ the function F map Y_i to each element in G with the same probability $\frac{1}{|G|}$. However, in practice the iteration function $F : G \to G$ is not a truly random mapping, which always results in more iteration requirements. The problem of efficient simulation of a random walk in the Pollard rho method is the central topic of this paper. Here, "efficient" means that the corresponding iteration function should require essentially no more than one group operation and use only constant or polynomial storage.

It is clear that the distinctive feature of the normal basis representations, namely, the p-th power of an element is just the cyclic shift of its normal basis representation where p is the characteristic of the underlying field, can be used to speed up the computation of discrete logarithms over finite extension fields $\mathbb{F}_{p^m}$. We propose a variant of the Pollard rho method by exploiting the distinctive feature of the normal basis representation, and achieve the speedup by a factor of $\sqrt{m}$, rather than $\frac{3p-3}{4p-3}\sqrt{m}$ [18], which is the previous result reported in the literature.

For computational and communication efficiency reasons, there are many cryptosystems [11,12] that suggest the use of bilinear maps over binary or ternary fields. For pairing-based cryptosystems, a bilinear map $e : G_1 \times G_2 \to G_3$ is used. When a Tate or Weil pairing is in use, G_1 is a subgroup of points on an elliptic curve $E(\mathbb{F}_{p^l})$ and G_3 is a cyclic subgroup of $\mathbb{F}^*_{p^{kl}}$, where k is the embedding degree. The MOV attack [13] transforms the DLP on G_1 or G_2 into a DLP on G_3. When our algorithm is applied to G_3, which is a subgroup of $\mathbb{F}^*_{p^{kl}}$, the complexity of the DLP on G_3 is reduced by a factor of $\sqrt{kl}$.

Besides the theoretical analysis, we also compare the performances of the new method with the previous algorithm in experiments, and the result confirms our analysis. The rest of this paper is structured as follows. In Section 2, we recall Pollard's rho method for discrete logarithm computation, and discuss the previous method for computing DLP over finite extension fields. We describe and analyze the new method in Section 3, and discuss its applications and our experiments in Section 4. Section 5 concludes this paper.

2. Preliminaries

In this section, we describe Pollard's rho method for discrete logarithm computation, and then discuss the previous method for computing DLP over finite extension fields, which takes advantage of the normal basis representation.

2.1. Pollard's rho method

Pollard [9] proposed an elegant algorithm for the discrete logarithms based on a Monte Carlo idea and called it the rho method. The rho method works by first defining a sequence of elements that will be periodically recurrent, then looking for a *match* in the sequence. The *match* will lead to a solution of the discrete logarithm problem with high probability. The two key ideas involved are the iteration function for generating the sequence and the cycle-finding algorithm for detecting a *match*.

If D is any finite set and $F : D \to D$ is a mapping and the sequence (X_i) in D is defined by the rule:

$$X_0 \in D, \qquad X_{i+1} = F(X_i)$$

this sequence is ultimately periodic. Hence, there exist unique integers $\mu \geq 0$ and $\lambda \geq 1$ such that $X_0, \ldots, X_{\mu+\lambda-1}$ are all distinct, but $X_i = X_{i+\lambda}$ for all $i \geq \mu$. A pair (X_i, X_j) of two elements of the sequence is called a *match* if $X_i = X_j$ where $i \neq j$. For the expected values of μ and λ, we have the following theorem:

Theorem 1 ([19]). *Under the assumption that an iteration function $F : D \to D$ behaves like a truly random mapping and the initial value X_0 is a randomly chosen group element, the expected values for μ and λ are $\sqrt{\pi |D|/8}$. The expected number of evaluations before a match appears is $E(\mu + \lambda) = \sqrt{\pi |D|/2} \approx 1.25\sqrt{|D|}$.*

2.1.1. Iteration function

Now we explain how the rho method for computing discrete logarithms works. Let G be a cyclic group of prime order q and let $g \in G$ be generator of G and $h \in G$. The discrete logarithm problem is to compute k satisfying $g^k \equiv h$. Pollard defined the iteration function $F : G \to G$ as follows:

$$Y_{i+1} = F(Y_i) = \begin{cases} g \cdot Y_i & Y_i \in S_1 \\ Y_i^2 & Y_i \in S_2 \\ h \cdot Y_i & Y_i \in S_3. \end{cases} \tag{1}$$

Let the initial value $Y_0 = 1$. In each iteration of $Y_{i+1} = F(Y_i)$, the function uses one of three rules depending on the value of Y_i. The group G is partitioned into three subsets S_1, S_2, S_3 of roughly equal size. Each Y_i has the form $g^{a_i} h^{b_i}$. The sequence

(a_i) (and similarly for (b_i)) can be computed as follows:

$$a_{i+1} = \begin{cases} a_i + 1 \pmod q & Y_i \in S_1 \\ 2a_i \pmod q & Y_i \in S_2 \\ a_i \pmod q & Y_i \in S_3. \end{cases} \tag{2}$$

As soon as we have a match (Y_i, Y_j), we have the equation $g^{a_i} * h^{b_i} = g^{a_j} * h^{b_j}$.

Since $h = g^k$, this gives,

$$a_i + b_i k \equiv a_j + b_j k \pmod q.$$

Now, if $\gcd(b_i - b_j, q) = 1$, we get that $k = (a_j - a_i)(b_i - b_j)^{-1} \bmod q$. Due to the method of Pohlig and Hellman [20], in practice applications the group order q is prime, so that it is very unlikely that $\gcd(b_i - b_j, q) > 1$ if q is large.

Theorem 1 makes the assumption of true randomness. However, it has been shown empirically that this assumption does not hold exactly for Pollard's iteration function [21]. The actual performance is worse than the expected value given in Theorem 1.

Teske [21] proposed better iteration functions by applying more arbitrary multipliers. Assume that we are using r partitions (multipliers). We generate $2r$ random numbers,

$$m_i, n_i \in_R \{0, 1, \ldots, q-1\}, \quad \text{for } i = 1, 2, \ldots, r.$$

Then we precompute r multipliers $M_1, M_2, \ldots, M_r$ where $M_i = g^{m_i} \cdot h^{n_i}$, for $i = 1, 2, \ldots, r$. Define a hash function,

$$v : G \to \{1, 2, \ldots, r\}.$$

Let $Y_0 = g^{a_0} h^{b_0}$, where a_0 and b_0 are two random integers in $[0, q-1]$. Then the iteration function $F : G \to G$ is defined as,

$$Y_{i+1} = F(Y_i) = Y_i \cdot M_j, \quad \text{where } j = v(Y_i).$$

The indices are updated by,

$$a_{i+1} = a_i + m_{v(Y_i)} \pmod q, \qquad b_{i+1} = b_i + n_{v(Y_i)} \pmod q.$$

The difference in performance between Pollard's original walk and Teske's r-adding walk has been studied in [22,21]. We summarize the results as follows. In prime order subgroups of $\mathbb{Z}_p^*$, the value of $E(\mu + \lambda)$ for Pollard's original walk and Teske's r-adding walk is $1.55\sqrt{|G|}$ and $1.27\sqrt{|G|}$, while in groups of points of elliptic curves over finite fields, the value is $1.60\sqrt{|G|}$ and $1.29\sqrt{|G|}$, respectively.

2.1.2. Collision detection

To find collisions in the pseudo-random walk, it always needs much storage. In order to minimize the storage requirements, a collision detection algorithm can be applied with a small penalty in the running time.

The idea of the distinguished point method is to search for a *match* not among all terms of the sequence, but only among a small subset of terms that satisfy a certain distinguishing property. It works as follows: One defines a set D, a subset of G, that consists of all group elements that satisfy a certain distinguishing property. During the pseudo-random walk, points that satisfy the distinguishing property are stored. Collision can be detected when a distinguished point is encountered a second time.

Currently, the distinguished point method is the most efficient algorithm to detect collisions in pseudo-random walk when $|G|$ is large. A popular way of defining D is to fix an integer k and to define that $w \in D$ if and only if the k least significant bits in the representation of w as a binary string are zero. Obviously, we have the following theorem.

Theorem 2 ([17]). *Let θ be the proportion of points in G which satisfy the distinguishing property, i.e., $\theta = |D|/|G|$. Under the assumption that $F : G \to G$ is a random mapping and D is a uniform distribution in G, the expected number of iterations before finding a match is $\sqrt{\pi |G|/2} + 1/\theta$.*

2.2. Previous method with normal basis representation

Let p be a prime and consider $\mathbb{F}_{p^m}$, the finite field with p^m elements. In this subsection, we briefly describe how to apply the previous Pollard rho method to cyclic subgroups of $\mathbb{F}_{p^m}^*$ to make use of the distinctive feature of the normal basis representation, which was proposed by Kim et al. [18].

We fix a normal basis $\{\alpha, \alpha^p, \ldots, \alpha^{p^{m-1}}\}$ for $\mathbb{F}_{p^m}$ and write elements of $\mathbb{F}_{p^m}$ using the coordinates in the normal basis, i.e., write $x = b_0\alpha + \cdots + b_{m-1}\alpha^{p^{m-1}}$ as $x = (b_0, \ldots, b_{m-1})$. Then our objective is to solve for $\log_g h$ in a cyclic group $G = \langle g \rangle \subset \mathbb{F}_{p^m}^*$ of prime order q.

There is a natural way to give $\mathbb{F}_p$ an ordering, and once a basis for $\mathbb{F}_{p^m}$ is fixed, we can give $\mathbb{F}_{p^m}$ the dictionary order using the ordering on $\mathbb{F}_p$. In particular, we have given an ordering to elements of $G \subset \mathbb{F}_{p^m}^*$. We define the map $\varphi : G \to G$ by

$$\varphi(x) = \min\{x^{p^i} | i = 0, \ldots, m-1\}. \tag{3}$$

When $\mathbb{F}_{p^m}$ is encoded using the normal basis, the function φ outputs the smallest of all cyclic shifts of its input. Notice that in any realization of φ that uses the normal basis, the number i producing the minimum will be known, in which case φ is naturally exponent traceable. The exponents are simply multiplied by p^i.

Therefore, one can define a variant r-adding walk as follows. Let $M_1, \ldots, M_r$ be the precomputed multipliers the same as the above. Let v be a hash function $v : G \to \{1, 2, \ldots, r\}$. Then the iteration function $F : G \to G$ defined as,

$$Y_{i+1} = \varphi(F(Y_i)) = \varphi(Y_i \cdot M_j), \quad \text{where } j = v(Y_i).$$

It is clear that F is a well-defined iteration function, and it can achieve a significant speed up. However, for each iteration one need to perform the function of φ, which slow the whole performance of F. More precisely, we have the following theorem.

Theorem 3 ([18]). *For a cyclic subgroup G of $\mathbb{F}^*_{p^m}$, by using φ as given by Eq. (3) and by working with the above variant of Pollard rho method, we can obtain a speed up by a factor of approximately $\frac{3p-3}{4p-3}\sqrt{m}$.*

3. New alternative algorithm

Iterative evaluations are the main operations of the Pollard rho method. We focus on how to design an efficient iteration function in this section. For efficiency, generally we have the following criteria: (a) for each iteration, the corresponding iteration function $F : G \to G$ should require essentially no more than one group operation, (b) the iteration function F behaves like or close to a truly random mapping and (c) the method uses only constant or polynomial storage.

3.1. The basic algorithm

As we have seen, the above variant r-adding walk needs more than one group operation, that is to perform the additional function φ in each iteration. In this subsection, we are expected to propose a more efficient variant random walk for the rho method. Generally, we first introduce an equivalence relation $\sim$ on the cyclic group $G = \langle g \rangle \subset \mathbb{F}^*_{p^m}$ of prime order q, and then define the random walk on the equivalence classes to reduce the search space, while keeping the iteration efficient.

More precisely, we set $\{\alpha, \alpha^p, \ldots, \alpha^{p^{m-1}}\}$ to be a normal basis of $\mathbb{F}_{p^m}$ and write elements of $\mathbb{F}_{p^m}$ using the coordinates in the normal basis. We define the equivalence relation $\sim$ on G as follows.

$$x \sim y : \quad \text{if } x = y^{p^i} \text{ for some } i \in \{0, 1, \ldots, m-1\}.$$

This relation partitions G into equivalence classes, and each class contains m elements. We denote the set of equivalence classes by $G/\sim$, and let $[x]$ denote the equivalence class containing x. Therefore,

$$[x] = \{x^{p^0}, x^{p^1}, \ldots, x^{p^{m-1}}\}.$$

Let v be a conventional hash function defined on $G/\sim$,

$$v : G/\sim \to \{0, 1, \ldots, m-1\}.$$

Let the map $\phi : G \to G$ be defined by,

$$\phi(x) = x^p.$$

Then, computing ϕ is trivial on $G \subset \mathbb{F}^*_{p^m}$, that is just the cyclic shift of its normal basis representation. Therefore, we can define the new iteration function F on $G/\sim$, instead of on G. Let the initial value $Y_0 = g^{a_0}h^{b_0}$, where a_0 and b_0 are two random integers in $\{0, 1, \ldots, q-1\}$. The new random mapping $F : G/\sim \to G/\sim$ is defined by,

$$Y_{i+1} = F(Y_i) = Y_i \cdot \phi^l(Y_i), \quad \text{where } l = v([Y_i]). \tag{4}$$

Then, each Y_i has the form $g^{a_i}h^{b_i}$, and the indices a_i and b_i can be updated correspondingly. Now, we explain why the new map is a well-defined map on $G/\sim$. More precisely, we have the following theorem.

Theorem 4. *Let $\sim$ be the equivalence relation defined as above, and let F be the random mapping on $G/\sim$ defined by Eq. (4). If $Y_i \sim Y_j$ for certain integers i and j, then $Y_{i+1} \sim Y_{j+1}$. Moreover, if $Y_i = \phi^l(Y_j)$ for certain integers i, j and l, then $Y_{i+1} = \phi^l(Y_{j+1})$.*

Proof. If $Y_i \sim Y_j$, then according to the definition of the equivalence relation, there exist certain integers i, j and l, such that $Y_i = \phi^l(Y_j)$. Let $k = v([Y_i]) = v([Y_j])$. Then we have

$$\begin{aligned} Y_{i+1} &= Y_i \cdot \phi^k(Y_i) \\ &= Y_i \cdot Y_i^{p^k} \\ &= \phi^l(Y_j) \cdot (\phi^l(Y_j))^{p^k} \\ &= Y_j^{p^l + p^{(l+k)}}. \end{aligned}$$

On the other hand, we know

$$\begin{aligned} Y_{j+1} &= Y_j \cdot \phi^k(Y_j) \\ &= Y_j \cdot Y_j^{p^k} \end{aligned}$$

then,

$$\begin{aligned} \phi^l(Y_{j+1}) &= (Y_j \cdot Y_j^{p^k})^{p^l} \\ &= Y_j^{p^l + p^{(l+k)}}. \end{aligned}$$

That is, $Y_{i+1} = \phi^l(Y_{j+1})$ and $Y_{i+1} \sim Y_{j+1}$. □

Theorem 4 shows that once the random walk defined by Eq. (4) falls into the same equivalence class with certain previous value, from that value on, the current walk and the previous walk will always fall into the same equivalence class for each step. This feature is the key point for the variant Pollard rho works. Then we can combine this feature with the distinguished point method to detect the collision. Therefore, the new iteration function is a well-defined map on $G/\sim$.

To make the new iteration function work, we need a partition function v defined on $G/\sim$. That is, for values coming from the same equivalence class, v outputs the same value. For $x = (b_0, \ldots, b_{m-1}) \in G$, let the Hamming weight $wt(x)$ of x be the number of its non-zero entries. One typical method is to define $v(x)$ as the function of its Hamming weight, that is $f(wt(x))$. Since in the normal basis representation, values from certain equivalence classes have the same Hamming weight. Then the partition function v is also a well-defined map on $G/\sim$.

Moreover, to make use of the distinguished point method to detect collisions, we can define the Hamming weight of x (also $[x]$) in a certain range as the distinguishing property. Once we find a distinguished point, generally we can normalize it by the function φ, and store the normalized distinguished point, and expect to find a collision among these stored values. Therefore, rather than the previous method, we only need to compute the Hamming weight of the current value and the iteration function, avoiding computing the function φ defined by Eq. (3) in each step. Thus, the new method is expected to achieve a speedup by a factor of $\sqrt{m}$, rather than $\frac{3p-3}{4p-3}\sqrt{m}$, the previous result reported in the literature.

3.2. Heuristic analysis of non-randomness

The new iteration function is defined via the Hamming weight of the normal basis representation of the group element. This reduces the randomness of the walk since the weight is not uniformly distributed. Certain heuristics for analyzing the randomness in the iteration function of the Pollard rho method have been discussed in [23,24]. Our analysis is a refinement of their heuristics.

Our iteration function is a multiplicative iteration function: every step maps an intermediate value Y_i to $Y_{i+1} = Y_i \cdot \phi^k(Y_i) = Y_i^{p^k+1}$, where $k \in \{0, 1, \ldots, m-1\}$. Let $s_k = p^k + 1$, assume that m different exponents s_k are used to define the random walk, and let pr_k be the probability that exponent s_k is used, and $pr_0 + pr_1 + \cdots + pr_{m-1} = 1$.

Let q be the cyclic group order, Z be a group element, Y and Y' be two independent uniform random group elements. Consider the event that Y and Y' both map to Z but $Y \neq Y'$. First, there is chance $\frac{1}{q^2}$ that $Z = Y^{s_i} = Y'^{s_j}$ with $i \neq j$; second, Y maps to Z with probability pr_i; third, Y' maps to Z with probability pr_j. Therefore, overall the probability is $(\sum_{i\neq j} pr_i pr_j)/q^2 = (\sum_{i,j} pr_i pr_j - \sum_i pr_i^2)/q^2 = (1 - \sum_i pr_i^2)/q^2$. This means that the probability of an immediate collision from Y and Y' is $(1 - \sum_i pr_i^2)/q$, where we add over the q choices of Z.

Therefore, after t iterations there are $t(t-1)/2$ pairs of group elements. The inputs are not uniform distributed random group elements, and the pairs are not independent, however one might nevertheless expect that the overall success probability is approximately $1 - (1 - (1 - \sum_i pr_i^2)/q)^{t(t-1)/2}$, and the average number of iterations before success is approximately $\sqrt{\pi q/(2(1 - \sum_i pr_i^2))}$.

That is, for the multiplicative walk that maps Y to Y^{s_i} with probability pr_i, the reductions of randomness increase the expected number of iterations by a factor of $1/\sqrt{1 - \sum_i pr_i^2}$, which is very close to 1 in most cases.

3.3. Application to pairing based cryptosystems

For pairing-based cryptosystems, a bilinear map $e : G_1 \times G_2 \to G_3$ is used, and bases its security on the intractability of the DLP over G_1. When a Tate or Weil pairing is in use, G_1 is a subgroup of points on an elliptic curve $E(\mathbb{F}_{p^l})$ and G_3 is a cyclic subgroup of $\mathbb{F}^*_{p^{kl}}$, where k is the *embedding degree* of G_1 and is usually chosen so that it is difficult to solve DLP over $\mathbb{F}^*_{p^{kl}}$ through the index calculus method. The MOV attack [13] transforms the DLP on G_1 or G_2 into the DLP on a subgroup of G_3. Therefore, it is expected that the best way to break the pairing-based cryptosystem is to solve the DLP with the Pollard rho method on G_1 or on the corresponding subgroup of G_3.

Table 1
Speed up factors for short signature compared to the previous method.

Curve	l	Speed up factor under Kim's approach	Speed up factor under our approach
E^-	79	14.51	21.77
E^+	97	16.08	24.12
E^+	121	17.96	26.94
E^+	149	19.93	29.90
E^+	163	20.85	31.27
E^-	163	20.85	31.27
E^+	167	21.10	31.65

Here we illustrate the effect of our method when it is applied to the subgroup of G_3 by a specific instance. Boneh et al. [11] proposed using supersingular curves $E^+(\mathbb{F}_{3^l}) : y^2 = x^3+2x+1$ and $E^-(\mathbb{F}_{3^l}) : y^2 = x^3+2x-1$. These curves have embedding degree 6 and we have $G_3 = \mathbb{F}^*_{3^{6l}}$. Thus, when our method is applied to G_3, the complexity of the DLP on G_3 is reduced by a factor of $\sqrt{6l}$, rather than $\frac{2}{3}\sqrt{6l}$, the previous result [18] reported in the literature. Therefore, one should expect to use a larger cyclic subgroup of the elliptic curve than was previously used to achieve the same security level.

For the supersingular curves with different l proposed in [11], when computing DLP over G_3 with the Pollard rho method, we list in Table 1 the speed up factors one can achieve using Kim's approach [18] and our approach, respectively.

Moreover, the ID-based encryption proposed in [10] is another example. Originally, these systems were built on elliptic curves over a larger prime field $\mathbb{F}_p$ with embedding degree 2. However, for efficiency reasons Galbraith [12] suggested the use of elliptic curves over characteristic 2 or 3 fields with Tate pairing of embedding degrees 4 and 6, respectively. These curves are $y^2 + y = x^3 + x$ and $y^2 + y = x^3 + x + 1$ in characteristic 2 and $y^2 = x^3 + 2x \pm 1$ in characteristic 3. In each of these cases, our approach can be applied to reduce the complexity by a factor of $\sqrt{4l}$ and $\sqrt{6l}$, respectively. In practice, if one chooses $l = 283$ or 397 with the curve $y^2 + y = x^3 + x + 1$, we can speed up the Pollard rho method by a factor of 33.65 or 39.85, respectively, a further speed up compared to the method used in [18].

4. Experiments

To evaluate the efficiency of the new method, we implemented discrete logarithm computations with the variant Pollard rho method over finite extension fields $\mathbb{F}_{2^m}$, where m in certain range. In this section, we describe these experiments and analysis the results.

The purpose of our experiments is to examine the impact of the new iteration function on the randomness of the rho method. In more detail, we evaluate the number of iterations until a collision is found for the Pollard rho method with the r-adding walk and the new random mapping, respectively.

Algorithm 1 Experiments for the r-adding walk and the new method

Require: Different iteration functions $F : G \to G$
Ensure: The average ratio (number of steps)/$\sqrt{q}$: R_m
1: **for** m=31, 37 and 41 **do**
2: $q \leftarrow$ the largest prime factor of $2^m - 1$.
3: Set $\{\alpha, \alpha^2, \ldots, \alpha^{2^{m-1}}\}$ be a normal basis of $\mathbb{F}_{2^m}$.
4: Choose a random element $w \in \mathbb{F}^*_{2^m}$ such that the order of w = $2^m - 1$.
5: $g \leftarrow (w^{(2^m-1)/q})$ (the generator of G).
6: **for** $i = 1$ to 100000 **do**
7: Choose a random number $c \in [0, q - 1]$, $h \leftarrow g^c$.
8: Choose a random element in G be the initial point Y_0.
9: $j \leftarrow 1$.
10: **repeat**
11: $Y_j \leftarrow F(Y_{j-1})$.
12: Check whether the Hamming weight of Y_j less than certain value.
13: **until** there is a match among distinguished points
14: $R_i \leftarrow j/\sqrt{q}$
15: **end for**
16: $R_m \leftarrow (\sum R_i)/100000$
17: **end for**

Generally, we set an integer m, such that the order of group $\mathbb{F}^*_{2^m}$ has a large prime factor q in a certain range. We set $\{\alpha, \alpha^2, \ldots, \alpha^{2^{m-1}}\}$ to be a normal basis of $\mathbb{F}_{2^m}$ and write elements of $\mathbb{F}_{2^m}$ using the coordinates in the normal basis. Then we set the generator g of G and choose a random element h of G. When using Pollard's rho method to compute this discrete

Table 2
Different performances for the r-adding walk and the new method.

Bits of m	#DLPs	Ratio for r-adding walk	Ratio for the new method
31	100 000	1.286	1.325
37	100 000	1.291	1.314
41	100 000	1.282	1.317
Average	300 000	1.286	1.319

logarithm, we count the number of steps we performed until a collision is found with different iteration functions on the same g and h. Then we determine the ratio R of the number of steps and $\sqrt{q}$. We repeat this a couple of times with the same g but several randomly chosen h. We have the Algorithm 1.

For simplicity, let l be the Hamming weight $wt(Y_i)$ of Y_i, we define the distinguishing property as $wt(Y_i)$ less than certain value. To search for collisions it is necessary to have a unique representative per class. We choose the lexicographically smallest value of $Y_i^{2^0}, Y_i^{2^1}, \ldots, Y_i^{2^{m-1}}$. In normal basis representation this is easily done by inspecting all cyclic shifts of Y_i.

More precisely, for each $m \in \{31, 37, 41\}$, we set a multiplicative group $\mathbb{F}^*_{2^m}$, which has a subgroup G of prime order $q \in [2^{27}, 2^{31}]$. Then for each group G, we generate 100 000 DLPs with the same generator g but randomly generated h. For each DLP, we use the r-adding walk and the new iteration function, respectively. Once we reach a collision, we compute the ratio R_l as (the number of steps until match is found)$/\sqrt{q}$. Finally, we count the average ratio R_i of all 100 000 DLPs with the same m and q.

The experiment results are given in Table 2. It shows that on average the new method indeed reduces the randomness of the rho method, which gives a total increase in the number of iterations by a factor of about 1.025 1 compared to the r-adding walk. It confirms our theoretic analysis that the impact on the randomness by the new method is very small. Therefore, experimentally the new approach can speed up the computation of discrete logarithms over finite extension fields $\mathbb{F}_{p^m}$ by a factor of extremely close to $\sqrt{m}$.

5. Conclusion

In this paper, we proposed a variant of the Pollard rho method by make use of the distinctive feature of the normal basis representations, namely, the p-th power of an element is just the cyclic shift of its normal basis representation where p is the characteristic of the underlying field. The new approach can speed up the computation of discrete logarithms over finite extension fields $\mathbb{F}_{p^m}$ by a factor of $\sqrt{m}$. Besides the theoretical analysis, we also compare the performances of the new method with the previous algorithm in experiments, and the result confirms our analysis. Due to the MOV reduction, our method can be applied to paring-based cryptosystems over binary or ternary fields.

Acknowledgments

We gratefully acknowledge the reviewers for the helpful comments and suggestions.

References

[1] W. Diffie, M. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (1976) 644–654.
[2] T. ElGamal, A public-key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on Information Theory 31 (1985) 469–472.
[3] FIPS 186-2, Digital signature standard, Federal Information Processing Standards Publication 186-2, February 2000.
[4] C.P. Schnorr, Efficient signature generation by smart cards, Journal of Cryptology 4 (1991) 161–174.
[5] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation 48 (1987) 203–209.
[6] V. Miller, Use of elliptic curves in cryptography, in: Advances in Cryptology: Proceedings of Crypto'85, in: LNCS, vol. 218, Springer-Verlag, New York, 1986, pp. 417–426.
[7] A. Menezes, P.V. Oorschot, S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.
[8] L. Adleman, A subexponential algorithm for the discrete logarithm problem with applications to cryptography, in: Proc. of the IEEE 20th Annual Symposium on Foundations of Computer Science, FOCS, 1979, pp. 55–60.
[9] J.M. Pollard, Monte Carlo methods for index computation mod p, Mathematics of Computation 32 (1978) 918–924.
[10] D. Boneh, M. Franklin, Identity-based encryption from the Weil pairing, in: J. Kilian (Ed.), CRYPTO 2001, in: LNCS, vol. 2139, Springer, Heidelberg, 2001, pp. 213–229.
[11] D. Boneh, B. Lynn, H. Shacham, Short signatures from the Weil pairing, Journal of Cryptology 17 (2004) 297–319.
[12] S. Galbraith, Supersingular curves in cryptography, in: C. Boyd (Ed.), ASIACRYPT 2001, in: LNCS, vol. 2248, Springer, Heidelberg, 2001, pp. 495–513.
[13] A. Menezes, T. Okamoto, P. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, IEEE Transactions on Information Theory 39 (5) (1993) 1636–1649.
[14] G. Frey, H.G. Rück, A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of curves, Mathematics of Computation 62 (1994) 865–874.
[15] R. Gallant, R. Lambert, S. Vanstone, Improving the parallelized Pollard lambda search on binary anomalous curves, Mathematics of Computation 69 (1999) 1699–1705.
[16] M. Wiener, R. Zuccherato, Faster attacks on elliptic curve cryptosystems, in: Selected Areas in Cryptography'98, in: LNCS, vol. 1556, Springer-Verlag, 1998, pp. 120–190.

[17] P.V. Oorschot, M. Wiener, Parallel collision search with cryptanalytic applications, Journal of Cryptology 12 (1999) 1–28.
[18] M. Kim, J.H. Cheon, J. Hong, Subset-restricted random walks for Pollard rho method on F_{p^m}, Public Key Cryptography 2009, 2009, pp. 54–67.
[19] B. Harris, Probability distribution related to random mappings, Annals of Mathematical Statistics 31 (1960) 1045–1062.
[20] S.C. Pohlig, M.E. Hellman, An improved algorithm for computing logarithms over GF(p) and its cryptographic significance, IEEE Transactions on Information Theory 24 (1978) 106–110.
[21] E. Teske, Speeding up Pollard's rho method for computing discrete logarithms, in: Algorithmic Number Theory Symposium (ANTS IV), in: LNCS, vol. 1423, Springer-Verlag, 1998, pp. 541–553.
[22] S. Bai, R.P. Brent, On the efficiency of Pollard's rho method for discrete logarithms, in: J. Harland, P. Manyem (Eds.), CATS 2008, Australian Computer Society, 2008, pp. 125–131.
[23] R.P. Brent, J.M. Pollard, Factorization of the eighth Fermat number, Mathematics of Computation 36 (1981) 627–630.
[24] D.V. Bailey, B. Baldwin, L. Batina, D.J. Bernstein, P. Birkner, J.W. Bos, G.V. Damme, G. Meulenaer, J. Fan, T. Güneysu, F. Gurkaynak, T. Kleinjung, T. Lange, N. Mentens, C. Paar, F. Regazzoni, P. Schwabe, L. Uhsadel, The certicom challenges ECC2-X, Cryptology ePrint Archive, Report 2009/466, 2009.