Finite Field Multiplication Combining AMNS and DFT Approach for Pairing Cryptography

Nadia El Mrabet[1] and Christophe Negre[2]

[1] LIRMM Laboratory, I3M, CNRS, University Montpellier 2, France
elmrabet@lirmm.fr
[2] DALI/ELIAUS, University of Perpignan, France

Abstract. Pairings over elliptic curves use fields $\mathbb{F}_{p^k}$ with $p \geq 2^{160}$ and $6 < k \leq 32$. In this paper we propose to represent elements in $\mathbb{F}_p$ with AMNS sytem of [1]. For well chosen AMNS we get roots of unity with sparse representation. The multiplication by these roots are thus really efficient in $\mathbb{F}_p$. The DFT/FFT approach for multiplication in extension field F_{p^k} is thus optimized. The resulting complexity of a multiplication in $\mathbb{F}_{p^k}$ combining AMNS and DFT is about 50% less than the previously recommended approach [2].

Keywords: Pairing, finite fields, AMNS, discrete Fourrier transform.

1 Introduction

Bilinear pairing in cryptography got an increasing interest during the past decade. Pairings were first use to attack discrete logarithm problem over elliptic curves like in MOV attack [3]. Since 2001, they are used also in a constructive way. Specifically, new important and original protocols using bilinear pairing have been proposed (e.g. Identity Based Cryptography [4] or Short Signature [5]). The most popular pairings used in pairing cryptography are defined over elliptic curves $E(\mathbb{F}_{q^k})$ (namely the Weil, Tate, η_T and Ate pairings [6]). Pairing evaluation over elliptic curve $E(\mathbb{F}_{q^k})$ involves arithmetical operations as multiplications and additions in the field $\mathbb{F}_{q^k}$ [2].

Fields $\mathbb{F}_{q^k}$ used in elliptic pairings are specific : q must have bit length bigger than 160 for security reasons and $6 < k < 32$ for optimization and security reasons. For now, the principal method [2] proposed to multiply elements in $\mathbb{F}_{q^k}$ uses a mix of Karatsuba and Toom-Cook multiplications methods. Consequently, they focus on k of the form $k = 2^i 3^j$, the resulting fields are called *friendly field* and the cost of a multiplication in $\mathbb{F}_{q^k}$ is equal to $3^i 5^j$ multiplications in $\mathbb{F}_q$.

Recently Discrete Fourier Transform approach has been proposed [7] to implement multiplication in $\mathbb{F}_{q^6}$ where $q = 3^n$. In practice Discrete Fourier Transform approach is interesting for quite large extension degree k. But here the underlying fields are quite big, so if the use of DFT can save even a small number of multiplications in $\mathbb{F}_q$ this can be advantageous.

In this paper, we extend the use of DFT for fields $\mathbb{F}_{p^k}$ where p is now a prime integer. The multiplication with DFT requires, in the best case, $2k - 1$

C. Boyd and J. González Nieto (Eds.): ACISP 2009, LNCS 5594, pp. 422–436, 2009.

multiplications in $\mathbb{F}_p$ and $O(k^2)$ multiplications by roots of unity. If FFT can be used, the cost of DFT approach becomes $O(n\log(n))$ operations in $\mathbb{F}_p$. If the field $\mathbb{F}_p$ is represented in usual way, the DFT approach remains too costly. Indeed, in this case the roots of unity are generally not nice (e.g. with a dense binary representation) and multiplications by these roots are costly. We propose here to use the AMNS system of $\mathbb{F}_p$ defined in [1]. In this situation we can manage to get roots of unity with nice representation, providing multiplications which are almost cost free. These multiplications can thus be neglect and the resulting multiplication algorithm in $\mathbb{F}_{p^k}$ requires only $2k-1$ multiplications in $\mathbb{F}_p$.

The paper is organized as follows : in Section 2 we recall the definition of the AMNS for representing integers modulo p and the arithmetic in this system. In Section 3 we recall the discrete Fourier transform approach for multiplication in $\mathbb{F}_{p^k}$ and extend it to specific cases in Subsection 3.3. We then focus on DFT friendly fieds (Section 4) which get benefit of a combination of AMNS and DFT for field multiplicattion. We evaluate the complexity of our approach for several field extensions and compare them to friendly fields. We finally conclude in Section 5.

2 Prime Field Arithmetic in AMNS System

Modular arithmetic operations like addition or multiplication modulo p consist to add or multiply two integers $0 \leq a, b < p$ and reduce the result modulo p if it is bigger than p.

Efficient arithmetic modulo a prime integer p is generally deeply related to the system of representation used to represent the elements. Generally integers are expressed as a sum $a = \sum_{i=0}^{\ell} a_i\gamma^i$ where $0 \leq a_i < \gamma$ and γ^ℓ has approximately the size of p. In practice γ is often chosen as 2^w where w is the size of computer words.

Here we will use an original system of representation in $\mathbb{F}_p$ introduced in [1] by Bajard, Imbert and Plantard called the *Adapted Modular Number System.* The main idea of the AMNS consists to relax the fact that $\gamma \cong p^{1/\ell}$. We take γ freely in $[0, p]$ such that each $0 \leq a < p$ can be written as $a = \sum_{i=0}^{\ell} a_i\gamma^i \mod p$ with $a_i \in [0, p^{1/\ell}]$. The advantage is that γ can be taken as $\gamma^\ell = \lambda \mod p$ where λ is small.

Definition 1 (AMNS [1]). *An Adapted Modular Number System $\mathcal{B}$, is a quadruple $(p, \ell, \gamma, \rho)_E$, where $E = t^\ell - \lambda$ such that $\gamma^\ell - \lambda = 0 \mod p$ and such that for all positive integers $0 \leq a < p$ there exists a polynomial $\mathfrak{a}(t) = \sum_{=0}^{\ell-1} a_i t^i$ satisfying*

$$\begin{aligned} &\mathfrak{a}(\gamma) = a \bmod p,\\ &\deg(\mathfrak{a}(t)) < \ell,\\ &\|\mathfrak{a}\|_\infty = \max_{i=1}^{\ell} |a_i| < \rho. \end{aligned} \tag{1}$$

The polynomial $\mathfrak{a}(t)$ is a representation of a in $\mathcal{B}$.

Generally in AMNS we have $\gamma \cong p$ and small coefficients $|a_i| < \rho \cong p^{1/\ell}$.

Table 1. The elements of $\mathbb{Z}_{17}$ in $\mathcal{B} = MNS(17,3,7,2)$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	1	$-t^2$	$1-t^2$	$-1+t+t^2$	$t+t^2$	$-1+t$	t	$1+t$	$-t-1$	$-t$	$-t+1$	$-t-t^2$	$1-t-t^2$	$-1+t^2$	t^2	-1

Example 1. In Table 1, we give the representation in the AMNS $\mathcal{B} = (17,3,7,2)$ for each element modulo $p = 17$.

In particular, we can verify that if we evaluate $(-1+t+t^2)$ in γ, we have $-1+\gamma+\gamma^2 = -1+7+49 = 55 \equiv 4 \bmod 17$. We have also that $\|-1+t+t^2\|_\infty = 1 < 2$.

In [8] the authors showed that it is possible to build an AMNS of length ℓ when it is possible to compute a polynomial $\mathfrak{m}(\gamma) = 0 \mod p$ with $\|\mathfrak{m}\|_\infty$ small.

Proposition 1. *Let p be a prime integer and $\lambda \in \mathbb{Z}, \ell \in \mathbb{N}$ such that the polynomial $E = t^\ell - \lambda$ admits a root γ in $\mathbb{F}_p$. Then the following statements are true.*

i) There exists a polynomial $\mathfrak{m}$ such that $\mathfrak{m}(\gamma) = 0$ and $\|\mathfrak{m}\|_\infty \leq (\ell!)^{1/\ell} p^{1/\ell}$.
ii) Let $\sigma = \|\mathfrak{m}\|_\infty$ and $\rho = 2|\lambda|\ell\sigma$ then the system $\mathcal{B} = (p,\ell,\gamma,\rho)_E$ is an AMNS of $\mathbb{F}_p$.

Fields used in cryptographic pairing have a p randomly constructed. Thus multiplication modulo p cannot use some rare property of p, like the primes considered in [1]. The better algorithm in AMNS which does not use rare property of prime p is the Montgomery-like multiplication presented in [8].

Algorithm 1: AMNS Multiplication

```
Input : $\mathfrak{a}, \mathfrak{b} \in \mathcal{B} = (p,\ell,\gamma,\rho)_E$ with $E = t^\ell - \lambda$
Data  : $\mathfrak{m}$ a polynomial such that $\mathfrak{m}(\gamma) \equiv 0 \pmod p$
        an integer $\phi$ and $\mathfrak{m}' = -\mathfrak{m}^{-1} \bmod (E,\phi)$
Output: $\mathfrak{r}(t)$ such that $\mathfrak{r}(\gamma) = \mathfrak{a}(\gamma)\mathfrak{b}(\gamma)\phi^{-1} \bmod p$
begin
    $\mathfrak{c} \leftarrow \mathfrak{a} \times \mathfrak{b} \bmod E$;
    $\mathfrak{q} \leftarrow \mathfrak{c} \times \mathfrak{m}' \bmod (E,\phi)$ ;
    $\mathfrak{r} \leftarrow (\mathfrak{c} + \mathfrak{q} \times \mathfrak{m} \bmod E)/\phi$;
end
```

According to [8] this algorithm is correct if $\phi \geq 2\ell\lambda\rho$. Concerning the implementation of Algorithm 1, it requires essentially three polynomial multiplications where polynomial coefficients are smaller than ρ and ϕ.

Such polynomial multiplication can be implemented using classical approach : for really small length ℓ schoolbook method are generally recommended, for bigger ℓ Karatsuba or Toom-Cook should be better. We will use here $\ell \leq 60$, thus, we will always use one of this two methods.

3 Field Extension Arithmetic

An extension field $\mathbb{F}_{p^k}$ can be seen as the set of polynomials with coefficient in $\mathbb{F}_p$ and degree less than k

$$\mathbb{F}_{p^k} = \{U(X) \in \mathbb{F}_p[X] \text{ s.t. } \deg U < k\}.$$

Arithmetic in this set is done modulo an irreducible polynomial P with degree k. Since p is large, the polynomial P can be taken, in general, with a binomial form $X^k - \alpha$ with small α (cf. [2,9]). In this situation the multiplication modulo P of two elements

$$U = \textstyle\sum_{i=0}^{k-1} u_i X^i \text{ and } V = \sum_{i=0}^{k-1} v_i X^i$$

consists first to compute the product $W = U \times V$ and then to reduce it modulo P. Since P is a binomial, the reduction modulo P is simple. We split $W = \underline{W} + X^k \overline{W}$ with $\deg \underline{W} \leq k$ and compute $\underline{W} + \alpha \overline{W}$ since $X^k \equiv \alpha \mod P$. The main challenge is thus to perform efficiently the polynomial multiplication $U \times V$.

3.1 Polynomial Multiplication Using DFT

We recall here the Discrete Fourier Transform (DFT) approach for polynomial multiplication. This approach is a special case of the multi-evaluation/ interpolation strategy [10]. Multi-evaluation/interpollation perform a polynomial multiplication of two polynomials U and V by evaluating both of them in $n \geq 2k - 1$ elements of $\mathbb{F}_p$. Then we deduce the evaluation of $W = U \times V$ by computing term by term the evaluation of U and V. Finally we perform a Lagrange interpolation to get the polynomial form of W.

In the DFT approach the evaluation set used is the set of n-th roots of unity. Specifically, let $\omega \in \mathbb{F}_p$ be a primitive root of unity, then the DFT works as follow.

1. *Multi-evaluation.* Let U, V be two polynomials in $\mathbb{F}_p[X]$ with degree k. We compute the multi-evaluation of U

$$\hat{U} = DFT_\omega(U) = (U(1), U(\omega), \ldots, U(\omega^{n-1})).$$

This operation is usually called the Discrete Fourier Transform of U. The same is done for V. This operation can be done through a matrix vector product

$$\hat{U} = \begin{bmatrix} 1 & \omega & \omega^2 & \cdots & \omega^{k-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{(k-1)2} \\ \vdots & & & & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(k-1)(n-1)} \end{bmatrix} \cdot \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_{k-1} \end{bmatrix}.$$

2. *Term by term multiplications.* They are performed on $\hat{U}$ and $\hat{V}$

$$\hat{W} = (\hat{u}_1 \times \hat{v}_1, \hat{u}_2 \times \hat{v}_2 \ldots, \hat{u}_n \times \hat{v}_n),$$

we get the multi-evaluation of W where $W = U \times V$.

3. *Interpolation.* The interpolation consists to compute the polynomial form of W knowing its multi-evaluation in $\boldsymbol{\omega} = (1, \omega, \omega^2, \ldots, \omega^{(n-1)})$.

Lemma 1. *Let $\mathbb{F}_p$ be a prime field and ω be a primitive n-th root of unity. Let*

$$\Omega = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{(n-1)2} \\ \vdots & & & & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{bmatrix} \tag{2}$$

its inverse is given by

$$\Omega^{-1} = \frac{1}{n} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega' & \omega'^2 & \cdots & \omega'^{n-1} \\ 1 & \omega'^2 & \omega'^4 & \cdots & \omega'^{(n-1)2} \\ \vdots & & & & \vdots \\ 1 & \omega'^{n-1} & \omega'^{2(n-1)} & \cdots & \omega'^{(n-1)(n-1)} \end{bmatrix} \tag{3}$$

where $\omega' = \omega^{-1} = \omega^{n-1}$.

In this situation the interpolation is computed by applying Ω^{-1} to $\hat{W}$ and keeping only the first $2k-1$ coefficients. We obtain

$$W = \frac{1}{n} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega' & \omega'^2 & \cdots & \omega'^{n-1} \\ 1 & \omega'^2 & \omega'^4 & \cdots & \omega'^{(n-1)2} \\ \vdots & & & & \vdots \\ 1 & \omega'^{2k-2} & \omega'^{2(2k-2)} & \cdots & \omega'^{(n-1)(2k-2)} \end{bmatrix} \cdot \begin{bmatrix} \hat{w}_1 \\ \hat{w}_2 \\ \vdots \\ \hat{w}_n \end{bmatrix}.$$

Remark 1 (Montgomery representation.). We can avoid the division by n in the interpolation process. Indeed, if we use a Montgomery representation (cf [11]) of U and V

$$\widetilde{U} = \frac{1}{n} U \text{ and } \widetilde{V} = \frac{1}{n} V.$$

If we perfom DFT approach without the division by n to multiply $\widetilde{U}$ and $\widetilde{V}$ we get

$$n\widetilde{U}\widetilde{V} = n(\frac{1}{n}U) \times (\frac{1}{n}V) = \widetilde{W},$$

where $W = U \times V$. In other words DFT multiplication without division by n is stable in Montgomery representation. This representation is also stable under addition and reduction modulo P. It can thus be used in a chain of multiplication/addition, like in pairing evaluation over elliptic curves.

3.2 Fast Fourier Transformation (FFT)

Let $U = \sum_{i=0}^{n-1} u_i X^i \in \mathbb{F}_p[X]$ and ω be a primitive root of unity. The fast Fourier transform (FFT) is an algorithm which performs efficiently the evaluation of U in $\boldsymbol{\omega} = (1, \omega, \omega^2, \ldots, \omega^{n-1})$. The FFT process is based on the following two-way splitting of U

$$U_1 = \textstyle\sum_{j=0}^{n/2-1} a_{2j} X^{2j},$$
$$U_2 = \textstyle\sum_{j=0}^{n/2-1} a_{2j+1} X^{2j},$$

such that $U = U_1 + XU_2$.

Let $\widehat{U}[i] = U(\omega^i)$ be the i-th coefficient of $\hat{U} = DFT_\omega(U)$. Let us also denote by $\hat{U_1}[i], \hat{U_2}[i]$ the coefficients of $DFT_{\omega^2}(U_1)$ and $DFT_{\omega^2}(U_2)$ in $\{1, \omega^2, \omega^4, \ldots, (\omega^2)^{n/2-1}\}$. If we evaluate $U = U_1 + XU_2$ in ω^i and $\omega^{i+n/2} = -\omega^i, i < n/2$ we get

$$\widehat{U}[i] = \widehat{U_1}[i] + \omega^i \widehat{U_2}[i]$$
$$\widehat{U}[i + n/2] = \widehat{U_1}[i] - \omega^i \widehat{U_2}[i]$$

The computation $DFT_\omega(U)$ is thus reduced to the computation of $DFT_{\omega^2}(U_1)$ and $DFT_{\omega^2}(U_2)$. These computations can be done recursivelly. The resulting algorithm has a cost of $\frac{n}{2}\log_2(n)$ multiplications by ω^i in $\mathbb{F}_p$ and $n\log_2(n)$ additions/subtractions.

3.3 Multiplication with DFT When $n \leq 2k - 2$

DFT approach for multiplication uses evaluations and interpolation in a set of $n \geq 2k-2$ roots of unity in order to get the correct product W. In some situations there is no primitive n-th root of unity with $n \geq 2k-1$ and n close to $2k-1$. In these situations DFT approach is not practical. We present here an extension of the DFT approach when there exists primitive a n-th root of unity smaller than $2k-1$. We focus here on two cases $n = 2k-2$ and $n = 2k-4$, which correspond to practical situations (see Section 4). The following approach generalizes the method presented in [7] when $k = 3$ and $n = 6$ to other value of k.

Lemma 2. *Let $\mathbb{F}_p$ be a prime field, ω be a primitive n-th root of unity and Ω and Ω^{-1} be the matrices defined in Lemma 1. We consider $U = \sum_{i=0}^{k-1} u_i X^i$, $V = \sum_{i=0}^{k-1} v_i X^i$ and $W = U \times V$ and we assume that $n = 2k - 2$. Then W can be computed as follow.*

1. $\hat{U} = DFT_\omega(U), \hat{V} = DFT_\omega(V)$.
2. $w_{2k-2} = u_{k-1} \times v_{k-1}$
3. *The coefficients* w_i *for* $i = 0, \ldots, 2k-3$ *are computed as*

$$\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{2k-3} \end{bmatrix} = \Omega^{-1} \cdot \begin{bmatrix} \hat{u}_i \times \hat{v}_i - w_{2k-2} \\ \hat{u}_i \times \hat{v}_i - w_{2k-2} \\ \vdots \\ \hat{u}_i \times \hat{v}_i - w_{2k-2} \end{bmatrix}. \tag{4}$$

Proof. Since $n = 2k-2$ and U and V have degree $k-1$ then $U \times V = W = \sum_{i=0}^{n} w_i X^i$ has degree n and $W = \sum_{i=0}^{n} w_i X^i$. The evaluation $\hat{W}$ of W in the n elements ω^i gives

$$\begin{cases} \hat{w}_0 = W(1) = w_0 + w_1 + \ldots + w_{n-1} + w_n \\ \hat{w}_1 = W(\omega) = w_0 + w_1\omega + \ldots + w_{n-1}\omega^{n-1} + w_n\omega^n \\ \hat{w}_2 = W(\omega^2) = w_0 + w_1(\omega^2) + \ldots + w_{n-1}(\omega^2)^{n-1} + w_n(\omega^2)^n \\ \quad\vdots \qquad\qquad \vdots \\ \hat{w}_{n-1} = W(\omega^{n-1}) = w_0 + w_1(\omega^{n-1}) + \ldots + w_{n-1}(\omega^{n-1})^{n-1} + w_n(\omega^{n-1})^n \end{cases}.$$

Now, since $\omega^n = 1$, we have $(\omega^i)^n = 1$ for $i = 1, \ldots, n-1$. The right part of the previous equations rewrites as

$$\begin{cases} \hat{w}_0 = W(1) = w_0 + w_1 + \cdots + w_n \\ \hat{w}_1 = W(\omega) = w_0 + w_1\omega + \cdots + w_n\omega^{n-1} + w_n \\ \hat{w}_2 = W(\omega^2) = w_0 + w_1(\omega^2) + \cdots + w_{n-1}(\omega^2)^{n-1} + w_n \\ \quad\vdots \qquad\qquad \vdots \\ \hat{w}_{n-1} = W(\omega^{n-1}) = w_0 + \cdots + w_{n-1}(\omega^{n-1})^{n-1} + w_n \end{cases} \tag{5}$$

The coefficient w_n is already known since $w_n = w_{2k-2} = u_{k-1}v_{k-1}$. Using (5), we remark that the vector $(\hat{w}_0 - w_n, \hat{w}_1 - w_n, \ldots, \hat{w}_{n-1} - w_n)$ is the discrete Fourier transform of the polynomial $W' = \sum_{i=0}^{n-1} w_i X^i$. Thus we get back to the coefficients of W' by computing

$$\Omega^{-1} \cdot \left[\hat{w}_0 - w_n \; \hat{w}_1 - w_n \; \cdots \; \hat{w}_{n-1} - w_n)\right]^t.$$

This corresponds to Eq. (4).

We focus now on the case $n = 2k-4$.

Lemma 3. *Let* $\mathbb{F}_p$ *be a prime field and* $\omega \in \mathbb{F}_p$ *a primitive n-th root of unity. Let* $U = \sum_{i=0}^{k-1} u_i X^i$ *and* $V = \sum_{i=0}^{k-1} v_i X^i$ *in* $\mathbb{F}_p[X]$. *Let* $W = U \times V$ *and assume that* $n = 2k-4$, *then the coefficients of* W *can be computed as follows.*

1. $\hat{U} = DFT_\omega(U), \hat{V} = DFT_\omega(V)$.
2. $w_{2k-2} = u_{k-1} \times v_{k-1}, w_0 = u_0 \times v_0$
3. $w_{2k-3} = u_{k-1} \times v_{k-2} + u_{k-2} \times v_{k-1}$

4. *The coefficients w_i for $i = 1, \ldots, 2k-4$ are computed as*

$$\begin{bmatrix} w_1 \\ w_2 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \Omega^{-1} \cdot \begin{bmatrix} \hat{w}_0 - w_0 - w_{n+1} - w_{n+2} \\ (\hat{w}_1 - w_0 - w_{n+1}\omega - w_{n+2}\omega^2)\omega^{-1} \\ (\hat{w}_2 - w_0 - w_{n+1}\omega^2 - w_{n+2}(\omega^2)^2)\omega^{-2} \\ \vdots \\ (\hat{w}_{n-1} - w_0 - w_{n+1}\omega^{n-1} - w_{n+1}(\omega^{n-1})^2)\omega^{-(n-1)} \end{bmatrix} \quad (6)$$

where Ω^{-1} is defined in Eq. (3)*.*

Proof. The proof is similar to the proof Lemma 2. Since $n = 2k-4$ and U and V have degree $k-1$ then $U \times V = W = \sum_{i=0}^{n+2} w_i X^i$ has degree $n+2 = 2k-2$. The evaluation $\hat{W}$ of W in the n elements ω^i gives

$$\left\{ \begin{array}{l} \hat{w}_0 = W(1) = w_0 + w_1 + \cdots + w_n + w_{n+1} + w_{n+1} \\ \hat{w}_1 = W(\omega) = w_0 + w_1\omega + \cdots + w_n\omega^n + w_{n+1}\omega^{n+1} + w_{n+2}\omega^{n+2} \\ \hat{w}_2 = W(\omega^2) = w_0 + w_1(\omega^2) + \cdots + w_n\omega^n + w_n(\omega^2)^n + w_{n+2}(\omega^2)^{n+2} \\ \quad \vdots \qquad\qquad\quad \vdots \\ \hat{w}_{n-1} = W(\omega^{n-1}) = w_0 + w_1(\omega^{n-1}) + \cdots + w_n(\omega^{n-1})^n \\ \qquad\qquad\qquad\qquad\qquad + w_{n+1}(\omega^{n-1})^{n+1} + w_{n+2}(\omega^{n-1})^{n+2} \end{array} \right.$$

Now, since $\omega^n = 1$, we have $(\omega^i)^{n+1} = \omega^i$ and $(\omega^i)^{n+2} = (\omega^i)^2$for $i = 1, \ldots, n-1$ in the right part of the previous equations. We get for $\hat{w}_i$

$$\left\{ \begin{array}{l} \hat{w}_0 = W(1) = w_0 + w_1 + \cdots + w_n + w_{n+1} + w_{n+2} \\ \hat{w}_1 = W(\omega) = w_0 + w_1\omega + \cdots + w_n\omega^n + w_{n+1}\omega + w_{n+1}\omega^2 \\ \hat{w}_2 = W(\omega^2) = w_0 + w_1(\omega^2) + \cdots + w_n(\omega^2)^n + w_{n+1}\omega^2 + w_{n+2}(\omega^2)^2 \\ \quad \vdots \qquad\qquad\quad \vdots \\ \hat{w}_{n-1} = W(\omega^{n-1}) = w_0 + \cdots + w_n(\omega^{n-1})^n + w_{n+1}\omega^{n-1} + w_{n+2}(\omega^{n-1})^2 \end{array} \right.$$

The three coefficients w_0, w_{n+1}, w_{n+2} are already known. Then we rewrite the previous equations as follows

$$\left\{ \begin{array}{rl} \hat{w}_0 - w_0 - w_{n+1} - w_{n+2} &= w_1 + \cdots + w_{n-1} \\ (\hat{w}_1 - w_0 - w_{n+1}\omega - w_{n+2}\omega^2)\omega^{-1} &= w_1 + \cdots + w_n\omega^{n-1} \\ (\hat{w}_2 - w_0 - w_{n+1}\omega^2 - w_{n+2}(\omega^2)^2)\omega^{-2} &= w_1 + \cdots + w_{n-1}(\omega^2)^{n-1} \\ \vdots \qquad & \qquad\quad \vdots \\ (\hat{w}_{n-1} - w_0 - w_{n+1}\omega^{n-1} - w_{n+1}(\omega^{n-1})^2)\omega^{-(n-1)} &= w_1 + \cdots + w_{n-1}(\omega^{n-1})^{n-1} \end{array} \right.$$

We remark now that the right part corresponds to the evaluation of the polynomial $W' = \sum_{i=0}^{n-1} w_{i+1}X^i$. Consequently we get the coefficients of W' by computing the matrix vector product

$$\begin{bmatrix} w_1 \\ w_2 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \frac{1}{n}\Omega \cdot \begin{bmatrix} \hat{w}_0 - w_0 - w_{n+1} - w_{n+2} \\ (\hat{w}_1 - w_0 - w_{n+1}\omega - w_{n+2}\omega^2)\omega^{-1} \\ (\hat{w}_2 - w_0 - w_{n+1}\omega^2 - w_{n+2}(\omega^2)^2)\omega^{-2} \\ \vdots \\ (\hat{w}_{n-1} - w_0 - w_{n+1}\omega^{n-1} - w_{n+1}(\omega^{n-1})^2)\omega^{-(n-1)} \end{bmatrix}$$

This corresponds to Eq. (6).

The previous Lemmas are just special cases. They can be extended to smaller value of n by increasing the number of precomputed w_i for small indices $i = 1, 2, \ldots$ and big indices $i = 2k-2, 2k-3, \ldots$ before the interpolation process using Ω^{-1}.

Remark 2. In the two situations of Lemma 2 and Lemma 3 we can use also Montgomery representation of polynomials to avoid division by n. For example if Lemma 2 is applied without the division by n to $\widetilde{U} = \frac{1}{n}U$ and $\widetilde{V} = \frac{1}{n}V$, we obtain the coefficients $\widetilde{w_i}$ of $\widetilde{W} = \frac{1}{n}U \times V$ for $i = 0, \ldots, 2k-2$. Only $\widetilde{w}_{2k-1}$ must be computed separatly $\widetilde{w}_{2k-2} = \frac{1}{n}w_{2k-1}$ to get all the coefficients of $\widetilde{W}$. The same strategy can be applied to Lemma 3 with 3 more multiplications by $\frac{1}{n}$.

3.4 Complexity of Different DFT Methods

We evaluate the complexity of the different DFT approaches. We distinguished the cases where DFT is performed through a matrix vector product and where DFT is performed using FFT algorithm. We express the costs in term of the number of operations in $\mathbb{F}_p$: the number of multiplication by roots of unity, addition/subtraction, and multiplication. For multi-evaluation and interpolation we used the fact that the entries in Ω and Ω^{-1} are all powers of ω. We also assume that multiplication are done in Montgomery representation, in order to avoid the division by $\frac{1}{n}$ (cf. Remark 1 and Remark 2). We obtain the complexity given in Table 2.

Table 2. Complexity of DFT approaches

Method	# Mult. by ω^i	# Mult.	# Add.
General DFT	$4nk - 3n$	n	$4nk - 3n$
General FFT	$\frac{3n}{2}\log_2(n)$	n	$3n\log_2(n)$
Lemma 2	$3(2k-3)^2$	$2k$	$(2k-2)(6k-8)$
Lemma 2 with FFT	$3(k-1)\log_2(2k-2)$	$2k$	$3(2k-2)\log_2(2k-2) + (2k-2)$
Lemma 3	$3(2k-5)^2 + 2(2k-5)$	$2k+3$	$3(2k-4)(2k-4-1) + 2(2k-4)$
Lemma 3 with FFT	$3(k-2)\log_2(2k-4) + 3(2k-5)$	$2k+3$	$3(2k-4)(\log_2(2k-4)+1)$

4 DFT Friendly Field

We focus in this section on specific fields called DFT friendly fields. These fields admit an AMNS which provide efficient multiplication by roots of unity.

4.1 Definition of DFT Friendly Field

The main goal is to find a way to obtain fields $\mathbb{F}_p$ with n-roots of unity such that $n \in \{2k-1, 2k-2, 2k-3\}$ and such that the multiplication by these roots are really efficient. We propose to consider fields $\mathbb{F}_{p^k}$ satisfying the following definition.

Definition 2 (DFT Friendly Field). *We call a DFT friendly field an extension field $\mathbb{F}_{p^k}$ such that $\mathbb{F}_p$ admits an AMNS $\mathcal{B} = (p, \ell, \gamma, \rho)_E$ of length ℓ and such that one of the following conditions holds*

1. $\lambda = 1$ *and* $\ell \in \{2k-1, 2k-2, 2k-4\}$ *and* γ *is primitive* ℓ*-th root of unity.*
2. $\lambda = -1$ *and* $\ell \in \{k-1, k-2\}$ *and* γ *is primitive* 2ℓ*-th root of unity.*

Since we have roots of unity with appropriate order, we can use DFT approaches presented in Section 3 to perform the multiplication of elements in $\mathbb{F}_{p^k}$. Indeed the condition on ℓ in each cases of Definition 2 enables us to use at least one of the strategies expressed in Subsection 3.1 or Lemma 2 and Lemma 3.

In DFT Friendly fields, the roots of unity are the elements $\pm\gamma^i$. The multiplication by these roots can be done using the formula stated in the following Lemma.

Lemma 4. *Let an AMNS* $\mathcal{B} = (p, \ell, \gamma, \rho)_E$ *and* $\mathfrak{a} = \sum_{i=0}^{n-1} a_i \gamma^i$ *be expressed in* $\mathcal{B}$*. The multiplication of* $\mathfrak{a}$ *by the power* γ^i *of* γ *is given by*

$$\mathfrak{a}\gamma^i = \lambda a_{n-i} + \lambda a_{n-i+1}\gamma + \cdots + \lambda a_{n-1}\gamma^{i-1} + a_0\gamma^i + \cdots + a_{n-i-1}\gamma^{n-1}$$

Proof. The proof is a direct consequence of the definition of an AMNS.

In our case, the field $\mathbb{F}_p$ is represented with an AMNS where $\lambda = \pm 1$. Consequently the multiplication by $\pm\gamma^i$ consists of only a cyclic shift, with eventually some changes of sign. The multiplication by $\pm\gamma^i$ is almost free of computations. Consequently, they do not add multiplications. The total cost of an AMNS multiplication in term of operations on the finite field is given in Table 3.

4.2 Fields Used Pairing Cryptography

We recall here different methods used to construct ordinary elliptic curves and corresponding finite fields providing pairing. The curve order $\#E(\mathbb{F}_p)$ must have a big prime factor, called r and an extension degree $6 < k \leq 32$. To get such curve the most used method is based on Complex Multiplication (CM).

The construction of a curve with the CM method requires to solve a system of equations (7) where the indeterminates are an integer D, the embedding degree k, the prime factor r, t the trace of the Frobenius on $E(\mathbb{F}_p)$ and p the characteristic of the finite field:

$$\begin{cases} r \mid p + t - 1, \\ r \mid p^k - 1, \text{ for primes } r, p, \\ Dy^2 = 4p - t^2 \text{ for some integer } y. \end{cases} \tag{7}$$

Several methods exist to solve this system. An overview of these different methods is given in [12]. We recall here the two following methods

- The Miyaji-Nakabayashi-Takano (MNT) strategy is one of the first CM method [13] to construct elliptic curves suitable for ECC. It was extended by Barreto and Naehrig [14] to construct elliptic curves with embedding degree 12. These curves with embedding degree 12 are given by the following parametrization:

$$\begin{aligned} k &= 12, \\ p &= x^4 + 36x^3 + 24x^2 + 6x + 1, \\ r &= 36x^4 + 36x^3 + 18x^2 + 6x + 1, \\ t &= 6x^2 + 1. \end{aligned}$$

– The second method which could be used in order to build curves with arbitrary embedding degree k is the Cocks-Pinch method [15]. This method generates curves with arbitrary r, such that $\#E(\mathbb{F}_p)/r \approx 2$. The extension of the Cocks-Pinch method given in [16] provides smaller value for $\#E(\mathbb{F}_p)/r$. Their method can be applied for general embedding degree. For example in [16] they generated a family of curves with embedding degree 16. This family is given by the following polynomials:

$$\begin{aligned} k &= 16, \text{ for } x \equiv \pm 25 \bmod(70) \\ p &= (x^{10} + 2x^9 + 5x^8 + 48x^6 + 152x^5 + 240x^4 \\ &\quad +625x^2 + 2398x + 3125)/980, \\ r &= (x^8 + 4x^4 + 625)/61250, \\ t &= (2x^5 + 41x + 35)/35. \end{aligned}$$

For all these constructions, the prime p is constructed randomly. Proposition 1 tells us that if there exists a primitive ℓ-th (or 2ℓ-th) root of unity, where ℓ satisfies the conditions of Definition 2, then we can construct an AMNS satisfying Definition 2.

For a random prime p, the probability that it has a primitive ℓ-th root of unity is roughly $1/(\ell - 1)$. Indeed p has a root of unity if and only if $p \equiv 1 \mod \ell$. But prime are equally distributed in the set of classes modulo ℓ. Consequently for small value of ℓ, we can easily find a DFT friendly field $\mathbb{F}_p$ and an elliptic curve over this field providing practical pairing.

4.3 Complexity Comparison

We present in Table 3 the complexity of a multiplication in DFT friendly fields $\mathbb{F}_{p^k}$ for different sizes of k. This complexity is given in term of the number of multiplications and additions in $\mathbb{F}_p$. The complexity is deduced from Table 2 : we neglect the cost of the multiplication by a root of unity since it is almost cost free. Indeed, a multiplication by a root of unity consists only in cyclic shifts. We also specify if we use FFT or not. Table 3 we also give the complexity of the multiplication in friendly field using Karatsuba and Toom Cook multiplications [9].

We remark that even for small value of k, DFT approach seems competitive regarding to the number of multiplications. When no FFT can be used, the number of additions increases significantly and should make our approach incompetitive in these special cases.

Table 3. Complexity comparison for practical extension degree k

Method	k	Cost of $Mult_{F_{p^k}}$	
		# Add. in $\mathbb{F}_p$	# Mult. in $\mathbb{F}_p$
Karatsuba/Toom-Cook [2,9]	6	60	15
Karatsuba/Toom-Cook [2,9]	8	72	27
Subsection 3.1 with FFT and $E = t^8 + 1$	8	192	16
Karatsuba/Toom-Cook [2,9]	9	160	25
Lemma 2 with FFT and $E = t^8 + 1$	9	208	18
Lemma 3 with FFT and $E = t^8 + 1$	10	240	23
Subsection 3.1 with $E = \sum_{i=0}^{10}(-t)^i$	11	902	22
Karatsuba/Toom-Cook [2,9]	12	180	45
Lemma 2 with $E = \sum_{i=0}^{10}(-t)^i$	12	1408	24
Lemma 3 with $E = \sum_{i=0}^{10}(-t)^i$	13	1430	28
Karatsuba/Toom-Cook [2,9]	16	248	81
Subsection 3.1 with FFT and $E = t^{16} + 1$	16	480	32
Lemma 2 with FFT and $E = t^{16} + 1$	17	512	34
Lemma 3 with FFT and $E = t^{16} + 1$	18	576	39
Karatsuba/Toom-Cook [2,9]	24	588	135

5 Conclusion

We have presented in this paper a new approach for multiplication in fields $\mathbb{F}_{p^k}$ used in pairing cryptography. We used AMNS [1] to represent elements in $\mathbb{F}_p$ and DFT approach for extension field arithmetic. Specificaly we pointed out that some AMNS provide efficient multiplication by roots of unity and thus optimize DFT approach. Our approch decrease the number of multiplications, but increase the number of additions in $\mathbb{F}_p$ in order to compute a multiplication in $\mathbb{F}_{p^k}$. However, the number additional additions is not so important compared to the saved multiplications. The resulting multiplication in the extension field $\mathbb{F}_{p^k}$ requires less multiplications in $\mathbb{F}_p$ for different practical sizes of k than previously recommended method [2,9]. Specifically for $k \geq 12$ combined AMNS and DFT approach in DFT friendly fields, proposed in this paper, decreases the number of multiplications in $\mathbb{F}_p$ by 50%.

References

1. Plantard, T.: Modular arithmetic for cryptography. PhD thesis, LIRMM, Université Montpellier 2 (2005)
2. Koblitz, N., Menezes, A.: Pairing-based cryptography at high security levels. In: Smart, N.P. (ed.) Cryptography and Coding 2005. LNCS, vol. 3796, pp. 13–36. Springer, Heidelberg (2005)
3. Menezes, A., Vanstone, S., Okamoto, T.: Reducing elliptic curve logarithms to logarithms in a finite field. In: STOC 1991: Proceedings of the twenty-third annual ACM symposium on Theory of computing, pp. 80–89. ACM Press, New York (1991)

4. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
5. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
6. Matsuda, S., Kanayama, N., Hess, F., Okamoto, E.: Optimised versions of the ate and twisted ate pairings. In: Galbraith, S.D. (ed.) Cryptography and Coding 2007. LNCS, vol. 4887, pp. 302–312. Springer, Heidelberg (2007)
7. Gorla, E., Puttmann, C., Shokrollahi, J.: Explicit formulas for efficient multiplication in $\mathbb{F}_{3^{6m}}$. In: Adams, C., Miri, A., Wiener, M. (eds.) SAC 2007. LNCS, vol. 4876, pp. 173–183. Springer, Heidelberg (2007)
8. Negre, C., Plantard, T.: Efficient modular arithmetic in adapted modular number system using lagrange representation. In: Proceedings of Australasian Conference on Information Security and Privacy (ACISPP 2008) (2008)
9. Bajard, J., Mrabet, N.E.: Pairing in cryptography: an arithmetic point of view. In: Advanced Signal Processing Algorithms, Architectures and Implementations XVI, SPIE (August 2007)
10. ZurGathen, J.V., Gerhard, J.: Modern Computer Algebra. Cambridge University Press, New York (2003)
11. Montgomery, P.L.: Modular multiplication without trial division. Mathematics of Computation 44(170), 519–521 (1985)
12. Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. Cryptology ePrint Archive (2006), `http://eprint.iacr.org/2006/372`
13. Miyaji, A., Nakabayashi, M., Takano, S.: New explicit conditions of elliptic curve traces for fr-reduction (2001)
14. Barreto, P., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 319–331. Springer, Heidelberg (2006)
15. Cocks, C., Pinch, R.: Identity-based cryptosystems based on the Weil pairing (2001)
16. Brezing, F., Weng, A.: Elliptic curves suitable for pairing based cryptography. Designs Codes and Cryptography 37(1), 133–141 (2005)
17. Kachisa, E.J., Schaefer, E.F., Scott, M.: Constructing brezing-weng pairing friendly elliptic curves using elements in the cyclotomic field. In: Pairing 2008: Proceedings of the 2nd international conference on Pairing-Based Cryptography, pp. 126–135 (2008)

6 Annexe

We present in this section some examples of curves with embedding degree $6 < k \leq 32$ over DFT friendly field.

We first briefly recall the method given in [8] to construct an AMNS for a fixed prime p. We choose a polynomial $E(t) = t^\ell - \lambda$ of degree ℓ and compute γ a root of E in $\mathbb{F}_p$. Then we construct the matrix M:

$$M = \begin{bmatrix} p & 0 & 0 & \cdots & 0 \\ -\gamma & 1 & 0 & \cdots & 0 \\ -\gamma^2 & 0 & 1 & \cdots & 0 \\ \vdots & & & \ddots & \vdots \\ -\gamma^{(\ell-1)} & 0 & \cdots & 0 & 1 \end{bmatrix}$$

We apply LLL algorithm to this matrix and we obtain a short vector $\mathfrak{m}$ satisfying $\mathfrak{m}(\gamma) = 0 \mod p$. We finally get $\rho = 2\ell|\lambda|\|\mathfrak{m}\|_\infty$.

6.1 Curves with Embedding Degree $k = 12$

We use the parametrization of Barreto and Naehrig [14] which provides elliptic curves with embedding degree 12:

$$\begin{aligned} k &= 12, \\ p &= 36x^4 + 36x^3 + 24x^2 + 6x + 1, \\ r &= 36x^4 + 36x^3 + 18x^2 + 6x + 1, \\ t &= 6x^2 + 1. \end{aligned}$$

We use also the polynomial $E(t) = \sum_{i=0}^{10} (-t)^i$ to build the AMNS of p. For a security level of 80 (i.e. the best attack requires 2^{80} operations) we find the following example:

$$\begin{aligned} x &= 1099511637026, \\ p &= 52614060714492069992659260093542155440429911322253, \\ r &= 52614060714492069992659252839987115706863666574197, \\ t &= 7253555039733566244748057, \\ \gamma &= 14348622953168487070046731700990451973985348345534, \\ \mathfrak{m}(X) &= 12376 - 49167X + 48460X^2 + 18281X^3 + 15213X^4 - 10299X^5 \\ &\quad +11263X^6 - 70120X^7 - 13636X^8 - 18106X^9. \end{aligned}$$

For a security level of 160 we found:

$$\begin{aligned} x &= 18446744073709692895, \\ p &= 416851521254351073793700573631526751155211200514430740527638681004 \\ &\quad 5976396949971, \\ r &= 4168515212543510737937005736315267511550070310942781743221955890 \\ &\quad 525925116063821, \\ t &= 2041694201525662054430919520051280886151, \\ \gamma &= 37771108084317046107302986198165197419883853864047699317 64 \\ &\quad 1286502190684739139, \\ \mathfrak{m}(X) &= 8053715 - 20923230X + 23417521X^2 - 26826999X^3 + 19243643X^4 \\ &\quad +1059907X^5 - 41954237X^6 - 42180723X^7 + 5371359X^8 - 19196965X^9. \end{aligned}$$

6.2 Curves with Embedding Degree $k = 16$

We use the parametrization of [17]:

$$\begin{aligned} k &= 16, \\ p &= (x^{10} + 2x^9 + 5x^8 + 48x^6 + 152x^5 + 240x^4 + 625x^2 + 2398x + 3125)/980, \\ r &= (x^8 + 48x^4 + 625)/61250, \\ t &= (2x^5 + 41x + 35)/35. \end{aligned}$$

We use the polynomial $E(t) = t^{16} + 1$.

For a security level of 80 we found the following example:

$$\begin{aligned}
x &= 74156485,\\
p &= 513174771603192518069857791127277415092088396567880595361684047 8\\
&\quad 933959934561,\\
r &= 1493093470726017930394028419006628852596285290848189053 6993,\\
t &= 128146760584932038247348983414439772062,\\
\gamma &= 386968286582177389475518658240604863510095482299776733841 3\\
&\quad 19721386977404674,\\
\mathfrak{m}(X) &= 7400X + 49262X^2 - 3010X^3 - 14335X^4 + 34360X^5\\
&\quad +43021X^6 + 6813X^7 + 5184X^8 + 13206X^9 + 10037X^{10}\\
&\quad +2540X^{11} - 7384X^{12} - 66117X^{13} - 57557X^{14} + 32450X^{15}.
\end{aligned}$$

For a security level of 160 we found :

$$\begin{aligned}
x &= 300650886015,\\
p &= 615742037941290064431987586434433942899976129045006271675961520 9\\
&\quad 3670796143014511127524512714937759452971210746 89,\\
r &= 108991796562856949188268637826460013069843005574422145615453925 9\\
&\quad 9303785820496070587549 93,\\
t &= 140370029614552009401267496693144064107592433030506438440,\\
\gamma &= 551737151471267013665906013312108810638488413639246814149706\\
&\quad 9474182490153198216829771585449969149779399226289 08,\\
\mathfrak{m}(X) &= 11792 + 15441X - 25387X^2 + 11348X^3 + 20103X^4 + 25605X^5\\
&\quad -8716X^6 + 9091X^7 + 19039X^8 + 13855X^9 - 22021X^{10} - 15182X^{11}\\
&\quad -4543X^{12} + 1417X^{13} - 26776X^{14} + 11502X^{15}.
\end{aligned}$$