19

Systematic Generation of An Irreducible Polynomial of An Arbitrary Degree m over $\mathbb{F}_p$ Such That $p > m$

Hiroaki Nasu[1], Yasuyuki Nogami[1], Yoshitaka Morikawa[1],
Shigeki Kobayashi[2] and Tatsuo Sugimura[2]
[1]*Okayama University,*
[2]*Shinshu University*
Japan

1. Introduction

There are many studies for generating irreducible polynomials (L. M. Adleman & H. W. Lenstra (1986)) − (Ian. F. Blake et al., (1993)). This is because irreducible polynomials play critical roles in the cases such as constructing extension field or generating random sequence. The problem of generating irreducible polynomial is theoretically interesting and have attracted many scientists and engineers. Those previous works are roughly classified by the objective: one is arbitrary − degree and the other is efficient for fast arithmetic operations in extension field. This paper is related to the former. As an application of the proposed method, the authors consider variable key − length public key cryptography (M. Scott (2006)).

Adleman et al. (L. M. Adleman & H. W. Lenstra (1986)) have shown that an irreducible polynomial of degree m over $\mathbb{F}_p$ with an arbitrary pair of p and m is generated by using a Gauss period normal basis (GNB) in $\mathbb{F}_{p^m}$ and Shoup shown almost the same idea (V. Shoup (1990)). Because, as introduced in Gao's paper (S. Gao (1993)), a GNB in $\mathbb{F}_{p^m}$ always exists for an arbitrary pair of p and m such that $4p$ does not divide $m(p - 1)$. However, they do not explicitly give a concrete generating algorithm. Of course, their calculation costs are not explicitly evaluated. Their methods are based on the minimal polynomial determination and efficiently using Newton's formula (R. Lidl & H. Niederreiter (1984)). On the other hand, the authors (K. Makita et al., (2005)) have explicitly given efficient generating algorithms in which characteristic $p = 2$ is only dealt with. These algorithms (K. Makita et al., (2005)) determine the minimal polynomial of TypeII ONB in $\mathbb{F}_{2^m}$ quite fast; however, if TypeII ONB does not exist in $\mathbb{F}_{2^m}$, it does not work. Thus, our previous works restrict not only degrees but also the characteristic to 2. Using Newton's formula and a certain special class of Gauss period normal bases in $\mathbb{F}_{p^m}$, this paper gives a concrete algorithm that efficiently generates an irreducible polynomial of degree m over $\mathbb{F}_p$ for an arbitrary pair of m and $p > m$. When $p > m$, it is automatically satisfied that $4p$ does not divide $m(p - 1)$. The restriction $p > m$

Source: Convergence and Hybrid Information Technologies, Book edited by: Marius Crisan,
ISBN 978-953-307-068-1, pp. 426, March 2010, INTECH, Croatia, downloaded from SCIYO.COM

comes from using Newton's formula. When one uses the distributive law instead, the proposed algorithm can avoid the restriction.

The main idea is as follows. Just like the previous works (L. M. Adleman & H. W. Lenstra (1986)), (V. Shoup (1990)), if we have arithmetic operations in $\mathbb{F}_{p^m}$, for a proper element a in $\mathbb{F}_{p^m}$ we can calculate its minimal polynomial $M_\alpha(x)$ with respect to the prime field $\mathbb{F}_p$, where a proper element means that it belongs to $\mathbb{F}_{p^m}$ but not to its proper subfield. It is well-known that $M_\alpha(x)$ becomes an irreducible monic polynomial over $\mathbb{F}_p$ and the coefficients of $M_\alpha(x)$ are systematically calculated from its vector representation by Newton's formula (V. Shoup (1990)), (R. Lidl & H. Niederreiter (1984)). In order to carry out multiplications in $\mathbb{F}_{p^m}$ without using an irreducible polynomial of degree m over $\mathbb{F}_p$, this paper uses cyclic vector multiplication algorithm (CVMA) (Y. Nogami et al., (2003)).

As previously described, this paper uses a special class of Gauss period normal bases (GNB). The special class normal basis is given from TypeI ONB (Y. Nogami et al., (2003)). In what follows, we call it TypeI-X NB (TypeI eXtended normal basis). The authors have proposed a multiplication algorithm for TypeI-X NB, it is cyclic vector multiplication algorithm (CVMA) (Y. Nogami et al., (2003)). It is noted that CVMA can calculate a multiplication in $\mathbb{F}_{p^m}$ without explicitly preparing an irreducible polynomial of degree m over $\mathbb{F}_p$ as the modulus polynomial of $\mathbb{F}_{p^m}$. Arithmetic operations in extension field $\mathbb{F}_{p^m}$ is defined by an irreducible polynomial $f(x)$ over $\mathbb{F}_p$ of degree m in which $f(x)$ is often called the modulus polynomial of $\mathbb{F}_{p^m}$. Using CVMA for TypeI-X NB, this paper shows an efficient algorithm for generating an irreducible polynomial of an arbitrary degree m over an arbitrary prime field $\mathbb{F}_p$ such that $p > m$. It uses Newton's formula. In other words, this paper explicitly gives an efficient algorithm for the ideas introduced by (L. M. Adleman & H. W. Lenstra (1986)), (V. Shoup (1990)). After that, this paper shows that the proposed algorithm can quite efficiently determine the minimal polynomial of TypeI-X NB in $\mathbb{F}_{p^m}$. The proposed algorithm has the following features: 1) it efficiently determines the minimal polynomial of the special class normal basis (TypeI-X NB), 2) its calculation complexity does not closely depend on the size of characteristic p, 3) its calculation cost is clearly given with degree m, thus we can estimate how much calculation time the proposed algorithm needs, 4) it can generate primitive polynomials if $(p^m - 1)$ is factorized as the product of prime numbers, and 5) as compared to distinct degree factorization based irreducibility testing algorithm (J. Gathen & D. Panario (2001)) and the case using the distributive law instead of Newton's formula, it generates an irreducible polynomial much faster.

As an application, this paper considers variable key − length public key cryptography in which one fixes characteristic p within the word length of the processor concerned and varies degree m appropriately.

Throughout this paper, #SADD, #SMUL and #SINV denote the number of additions, multiplications, and that of inversions in $\mathbb{F}p$, respectively. In this paper, a subtraction in $\mathbb{F}p$ is counted up as an addition in $\mathbb{F}p$. p and m denote characteristic and extension degree, respectively, where p is a prime number. $\mathbb{F}_{p^m}$ denotes the m-th extension field over $\mathbb{F}p$ and $\mathbb{F}^*_{p^m}$ denotes the multiplicative group in $\mathbb{F}_{p^m}$. $X \mid Y$ means that X divides Y. Without any additional explanation, lower and upper case letters show elements in prime fields and extension fields, respectively, and a Greek character shows a zero of modulus polynomial. Polynomials in this paper are all monic polynomials.

2. Fundamentals

In this section, we briefly go over several classes of irreducible polynomials over $\mathbb{F}_p$.

2.1 Irreducible binomial

The well-known optimal extension field (OEF) adopts an irreducible binomial as the modulus polynomial (D. Bailey & C. Paar (2000)). We can easily prepare an irreducible binomial by the following theorem (R. Lidl & H. Niederreiter (1984)).

Theorem 1 There exist irreducible binomials in the form $x^m - s$, $s \in \mathbb{F}_p$ if and only if each prime factor of m divides $p - 1$ and $4 \mid (p - 1)$ when $4 \mid m$. ■

For example, let m be a prime, if the following relation holds, $x^m - s$ becomes irreducible over $\mathbb{F}_p$.

$$s^{(p-1)/m} \neq 1. \tag{1}$$

According to Theo.1, in this case p −1 must be divisible by the prime number m. Therefore, when m is large, irreducible binomials of degree m over $\mathbb{F}_p$ are quite restricted corresponding to p.

2.2 Irreducible trinomial

Irreducible trinomials have been studied especially for characteristic $p = 2$. For an arbitrary pair of p and m, irreducible trinomials do not always exist (E. Berlekamp (1968)). In addition, it is not explicitly known when the following trinomial becomes irreducible over $\mathbb{F}_p$.

$$x^m + ax^n + b, \ a, b \in \mathbb{F}_p. \tag{2}$$

In general, in order to generate an irreducible trinomial in the form of Eq.(2), we need irreducibility tests with changing the parameters a, b, and n. Therefore, when both p and m are large, searching an irreducible trinomial becomes quite time-consuming.

2.3 Variable transformation

According to the following theorems (R. Lidl & H. Niederreiter (1984)), (Y. Nogami et al., (1999)), we can generate higher degree irreducible polynomials with corresponding variable transformations.

Theorem 2 For an irreducible polynomial $f(x)$ of degree m over $\mathbb{F}_p$, if and only if $f(x)$ satisfies

$$x^{(p^m-1)/k} \not\equiv 1 \bmod f(x) \tag{3}$$

for a certain prime number k such that k divides $p^m - 1$, $f(x^k)$ becomes irreducible over $\mathbb{F}_p$. ■

Theorem 3 For an irreducible polynomial $f(x)$ of degree m over $\mathbb{F}_p$, if and only if the $(m - 1)$-th coefficient is not 0, $f(x^p - x)$ becomes irreducible over $\mathbb{F}_p$. ■

Based on these theorems, we can generate infinite number of irreducible polynomials of degree mk^i (R. Lidl & H. Niederreiter (1984)) and mp^i (Y. Nogami et al., (1999)), respectively; however, prime degree irreducible polynomials are not generated. In addition, we need a certain seed irreducible polynomial $f(x)$.

2.4 Cyclotomic irreducible polynomial

According to the next theorem, we can easily obtain all one irreducible polynomial $(x^{m+1}-1)/(x-1)$ (T. Sugimura & Y. Suetugu (1991)). The coefficients of $(x^{m+1}-1)/(x-1)$ are all one, therefore it is called all one polynomial of degree m.

Theorem 4 All one polynomial $(x^{m+1}-1)/(x-1)$ of degree m is irreducible over $\mathbb{F}_p$ if and only if the following conditions are both satisfied.

1. $m + 1$ is a prime number, therefore m is even.
2. p is a primitive element in $\mathbb{F}_{m+1}$, where note that $m + 1$ is a prime number, $\mathbb{F}_{m+1}$ denotes the prime field of order $m + 1$. ■

Sugimura et al. introduced all varieties of the cyclotomic irreducible polynomials (T. Sugimura & Y. Suetugu (1991)); however, as shown in the above theorem, the degree is a certain even number. In other words, odd degree irreducible polynomials can not be obtained as cyclotomic polynomials.

2.5 Distinct degree factorization

We can generate an irreducible polynomial $f(x)$ of a certain prime degree m over $\mathbb{F}_p$ by randomly preparing a polynomial of degree m over $\mathbb{F}_p$ and then testing its irreducibility over $\mathbb{F}_p$. For this irreducibility test, we can apply the distinct degree factorization (DDF) (E. Berlekamp (1968)). In the case that the degree m is a prime number, DDF checks the following relation:

$$f(x) \mid (x^{p^m} - x)/(x^p - x). \tag{4}$$

Noting that this paper mainly deals with characteristic p larger than m, $f(x)$ is irreducible over $\mathbb{F}_p$ if and only if $f(x)$ satisfies Eq.(4). This calculation requires polynomial multiplications and modulo operations, therefore it becomes more time-consuming as characteristic p and degree m become larger. Moreover, the possibility that a polynomial $f(x)$ of degree m becomes irreducible over $\mathbb{F}_p$ is about $1/m$. Therefore, when we apply such an irreducibility testing algorithm for generating an irreducible polynomial, it becomes a probabilistic problem. Since the calculation Eq.(4) needs $\mathcal{O}(m^{2.7} \log p)$ multiplications in $\mathbb{F}_p$ when we apply the well-known Karatsuba method for polynomial multiplications and the binary method for the exponentiation x^{p^m}(D. Knuth (1981)), generating an irreducible polynomial of degree m over $\mathbb{F}_p$ needs $\mathcal{O}(m^{3.7} \log p)$ multiplications in $\mathbb{F}_p$. Therefore, when both p and m are large, it will be a quite time-consuming operation.

2.6 Recursive generation

If we have an irreducible polynomial, we can recursively generate a lot of irreducible polynomials of the same degree (A. J. Menezes, editor (1993)); however, we need an irreducible polynomial as a generator.

2.7 Minimal polynomial determination

If we have the arithmetic operations in $\mathbb{F}_{p^m}$, we can generate an irreducible polynomial as the minimal polynomial of an arbitrary proper element in $\mathbb{F}_{p^m}$. If an element belongs to $\mathbb{F}_{p^m}$but not to its proper subfield, the element is called proper element in $\mathbb{F}_{p^m}$. In general, the arithmetic operations are defined by the modulus polynomial that is a certain irreducible polynomial of degree m over $\mathbb{F}_p$; however, some extension fields do not explicitly need an irreducible polynomial of degree m such as TypeII AOPF (Y. Nogami et al.,(2005)). TypeII AOPF adopts TypeII optimal normal basis (ONB).

The authors (K. Makita et al., (2005)) have proposed efficient algorithms for determining the minimal polynomial of TypeII ONB (Y. Nogami et al., (2005)). TypeII ONB only exists in the following extension fields $\mathbb{F}_{p^m}$.

Theorem 5 TypeII ONB exists in $\mathbb{F}_{p^m}$ if and only if p and m satisfy (1) and either (2a) or (2b):

1. $2m + 1$ is a prime number.

2.a p is a primitive element in $\mathbb{F}_{2m+1}$.

2.b The order of p mod $2m + 1$ is m and $2 \mid (m - 1)$. ■

The algorithms proposed in (K. Makita et al., (2005)), in which the case of $p = 2$ is only dealt with, are quite fast; however, they have the following problems:

- They determine the minimal polynomial of TypeII ONB in $\mathbb{F}_{2^m}$. In other words, if TypeII ONB does not exist in $\mathbb{F}_{2^m}$, they does not generate an irreducible polynomial of degree m over $\mathbb{F}_2$.
- They do not generate an irreducible polynomial over $\mathbb{F}_p$ for an arbitrary pair of p and m.

Adleman et al. (L. M. Adleman & H. W. Lenstra (1986)) and Shoup (V. Shoup (1990)) have introduced that an irreducible polynomial of degree m over $\mathbb{F}_p$ with an arbitrary pair of p and m can be generated by using a GNB in $\mathbb{F}_{p^m}$; however, they do not give any explicit algorithms. Of course, their calculation costs are not explicitly evaluated. Thus, this paper explicitly gives an algorithm that generates an irreducible polynomial of degree m over $\mathbb{F}_p$ by using a GNB in $\mathbb{F}_{p^m}$. In addition, the calculation cost is explicitly given. It is applied for an arbitrary pair of p and m.

3. Irreducible polynomial generation

This section introduces the idea and algorithms.

3.1 Main idea

In this section, we introduce a special class of Gauss period normal bases (GNB). The special class normal basis is given from TypeI ONB (Y. Nogami et al., (2003)). In this paper, we call it TypeI-X NB (TypeI eXtended normal basis). The authors have proposed a multiplication algorithm named cyclic vector multiplication algorithm (CVMA) (T. Yoshida et al., (2006)). It is also available for TypeI-X NB. It is noted that CVMA calculates a multiplication in $\mathbb{F}_{p^m}$ without explicitly preparing an irreducible polynomial of degree m over $\mathbb{F}_p$ as the modulus polynomial of $\mathbb{F}_{p^m}$. Using CVMA with TypeI-X NB, this paper shows an efficient algorithm for generating an irreducible polynomial of an arbitrary degree m over an arbitrary prime field $\mathbb{F}_p$. After that, it is shown that the proposed algorithm quite efficiently determines the minimal polynomial of TypeI-X NB in $\mathbb{F}_{p^m}$.

3.2 Minimal polynomial

Let us briefly go over the fundamentals of minimal polynomial. Let α be a proper element in $\mathbb{F}_{p^m}$. Then, its minimal polynomial $M_\alpha(x)$ is given as

$$M_\alpha(x) = \prod_{i=0}^{m-1} (x - \alpha^{p^i}) \tag{5a}$$

$$= x^m + a_{m-1}x^{m-1} + \cdots + a_1 x + a_0, \tag{5b}$$

where $a_{m-1}, \ldots, a_1, a_0$ are in $\mathbb{F}_p$. $M_\alpha(x)$ becomes a monic irreducible polynomial of degree m over $\mathbb{F}_p$ (R. Lidl & H. Niederreiter (1984)). When the degree m is large, it is too time-consuming to directly develop Eq.(5a) into Eq.(5b) with the distributive law; however, if m is smaller than p, we can systematically obtain each coefficient of $M_\alpha(x)$ by Newton's formula as described in the next section. As previously introduced, the restriction thus comes from using Newton's formula.

3.3 Minimal polynomial and Newton's formula

First, we define the notation $\mathrm{Tr}^{[n]}(\alpha)$ as follows.

Definition 1 For a proper element a in $\mathbb{F}_{p^m}$, consider its m conjugates as follows:

$$\{\alpha, \alpha^p, \cdots, \alpha^{p^{m-1}}\}. \tag{6}$$

Let $1 \leq n \leq m$, $\mathrm{Tr}^{[n]}(\alpha)$ is defined as

$$\mathrm{Tr}^{[n]}(\alpha) = \sum_{0 \leq i_1 < i_2 < \cdots < i_n \leq m-1} \alpha^{p^{i_1}} \alpha^{p^{i_2}} \cdots \alpha^{p^{i_n}}. \tag{7}$$

■

According to the Newton's formula (R. Lidl & H. Niederreiter (1984)), (V. Shoup (1990)), each coefficient of the minimal polynomial $M_\alpha(x)$ that is defined by Eqs.(5) is systematically given by

$$\begin{aligned} a_{m-n} &= (-1)^n \mathrm{Tr}^{[n]}(\alpha) \\ &= n^{-1} \left\{ -\mathrm{Tr}^{[1]}(\alpha^n) + \sum_{i=1}^{n-1} (-1)^{i+1} \mathrm{Tr}^{[i]}(\alpha) \mathrm{Tr}^{[1]}(\alpha^{n-i}) \right\} \end{aligned} \tag{8}$$

where $1 \leq n \leq m$ and $\mathrm{Tr}^{[1]}(\alpha^{n-i})$ is the trace of $\alpha^{n-i}$with respect to $\mathbb{F}_p$. As shown in Eq.(8), we need to calculate n^{-1}. Therefore, the above equation can be applied for the case that $p > m$. Newton's formula needs a lot of trace calculations, for which TypeI-X NB is also efficient because it is a normal basis (R. Lidl & H. Niederreiter (1984)).

3.4 A special class of Gauss period normal bases

Let us consider a special class of type-h$\langle k,m \rangle$ Gauss period normal bases as follows (T. Yoshida et al., (2006)).

Let km+1 be prime and suppose that p is a primitive element in $\mathbb{F}_{km+1}$. Then, let ω be a primitive (km + 1) fist root of unity, ω belongs to $\mathbb{F}_{p^{km}}$. The conjugates of ω form a TypeI ONB as follows.

$$\{\omega, \omega^p, \cdots, \omega^{p^{km-1}}\}. \tag{9}$$

Then, consider a special class of GNB as

$$\{\gamma, \gamma^p, \cdots, \gamma^{p^{m-1}}\}, \quad \gamma = \sum_{j=0}^{k-1} \omega^{p^{jm}}. \tag{10}$$

In this paper, we call the normal basis Eq.(10) TypeI-X normal basis (TypeI-X NB). A lot of studies about GNB have been done (M. Nöcker (2001)), (R. Granger (2005)). Gao (S. Gao (1993)) has discussed from the viewpoints of normal basis and self dual normal basis in detail. According to Gao's paper (S. Gao (1993)), TypeI-X NB in $\mathbb{F}_{p^m}$ always exists for an arbitrary pair of p and m such that $4p$ does not divide $m(p-1)$. The authors also checked it experimentally (T. Yoshida et al., (2006)). In the next section, how to carry out a multiplication with TypeI-X NB is introduced.

3.4.1 Multiplication with TypeI-X NB

A multiplication $Z = XY$ with TypeI-X NB in $\mathbb{F}_{p^m}$ is carried out by the algorithm shown in Fig 1. It is named cyclic vector multiplication algorithm (CVMA) (Y. Nogami et al., (2003)). The authors have improved CVMA several times (T. Yoshida et al., (2006)), (Y. Nogami et al., (2005)). In Fig 1. $\langle\cdot\rangle$ means $\cdot$ mod km + 1.

```
Input: $X = \sum_{i=0}^{m-1} x_i \gamma^{p^i}$, $Y = \sum_{i=0}^{m-1} y_i \gamma^{p^i}$.
Output: $Z = XY = \sum_{i=0}^{m-1} z_i \gamma^{p^i}$.
Preparation:
    1. Determine $k$ such that TypeI ONB exists.
    2. For $0 \le i \le m,\ q[i] \leftarrow 0$.
    3. For $0 \le t \le m-1$ and $0 \le h \le k-1,\ g\left[\langle p^{t+hm} \rangle\right] \leftarrow t+1$.
    4. $g[0] \leftarrow 0$.
Procedure:
    For $0 \le i \le m-1,\ q[i+1] \leftarrow x_i y_i$.
    For $0 \le i < j \le m-1$, {
        $M \leftarrow (x_i - x_j)(y_i - y_j)$,
        For $0 \le h \le k-1$, {
            $q\left[g\left[\langle p^i + p^{j+hm} \rangle\right]\right] \leftarrow q\left[g\left[\langle p^i + p^{j+hm} \rangle\right]\right] + M$.
        }
    }
    For $0 \le i \le m-1,\ z_i \leftarrow kq[0] - q[i+1]$.
                                                            (End of algorithm)
```

Fig. 1. CVMA in $\mathbb{F}_{p^m}$ with TypeI-X NB

This algorithm needs the following cost:

$$\#_{\mathrm{SMUL}} = \frac{m(m+1)}{2} + 1, \tag{11a}$$

$$\#_{\mathrm{SADD}} = \frac{m(m-1)(k+2)}{2} + m. \tag{11b}$$

The well−known Karatsuba method calculates a polynomial multiplication of degree m with $\mathcal{O}(m^{1.7})$ $\mathbb{F}_p$−multiplications (D. Knuth (1981)); however, in this case we need a certain modulus polynomial of degree m over $\mathbb{F}_p$. On the other hand, CVMA does not need such an irreducible polynomial of degree m over $\mathbb{F}_p$; however, extension degree m is preferred to be small.

3.5 Minimal polynomial determination with CVMA

Using CVMA, as Sec.3.3 we can determine the minimal polynomial$M_\alpha(x)$ of a proper element $\alpha \in \mathbb{F}_{p^m}$.

a. Calculate α^i and then $\mathrm{Tr}^{[1]}(\alpha^i)$, where $1 \leq i \leq m$.

b. Calculate the coefficients a_i, $0 \leq i \leq m-1$.

Noting that TypeI-X NB Eq.(10) is a normal basis, $\mathrm{Tr}^{[1]}(\alpha^i)$ is calculated by $m-1$ additions in $\mathbb{F}_p$ with the vector coefficients of α^i. When a vector is represented with a normal basis, its trace is calculated by adding all of the vector coefficients. In addition, whether or not the element is a proper element in $\mathbb{F}_{p^m}$ is easily checked from its vector coefficients. For an arbitrary proper element α, determining its minimal polynomial $M_\alpha(x)$ takes the following calculation cost.

For the operation (a),

$$\#_{\mathrm{SMUL}} = (m-1)\left(\frac{m(m+1)}{2} + 1\right), \tag{12a}$$

$$\#_{\mathrm{SADD}} = (m-1)\left(\frac{m(m-1)(k+2)}{2} + m\right) + m(m-1). \tag{12b}$$

In detail, for α^i, $1 \leq i \leq m$, we need $m-1$ multiplications in $\mathbb{F}_{p^m}$ with CVMA. Then, for $\mathrm{Tr}^{[1]}(\alpha^i)$, we need $m-1$ additions in $\mathbb{F}_p$. In total, we need Eqs.(12). For the operation (b),

$$\#_{\mathrm{SMUL}} = \#_{\mathrm{SADD}} = \frac{m(m-1)}{2} + (m-1), \tag{13a}$$

$$\#_{\mathrm{SINV}} = m-2. \tag{13b}$$

The calculation cost Eqs.(13) is given from Eq.(8). The operations (a) and (b) need $\mathcal{O}(m^3)$ and $\mathcal{O}(m^2)$ $\mathbb{F}_p$-multiplications, respectively. Thus, the major computation is for the operation (a).

By the way, since the proposed algorithm is based on the minimal polynomial determination, it can be applied for generating a primitive polynomial. In detail, if p^m-1, that is the order of $\mathbb{F}^*_{p^m}$, is factorized as the product of prime numbers, we can prepare a primitive element in $\mathbb{F}_{p^m}$ as a proper element α (R. Lidl & H. Niederreiter (1984)). Then, using a primitive element, the proposed algorithm generates a primitive polynomial of degree m over $\mathbb{F}_p$. In the next section, applying one of the basis elements shown in Eq.(10) as a proper element in $\mathbb{F}_{p^m}$, we improve the operation (a).

3.6 Minimal polynomial of TypeI-X NB

Using γ defined in Eq.(10), that is a proper element in $\mathbb{F}_{p^m}$ and its conjugates form the TypeI-X NB Eq.(10), we calculate its minimal polynomial $M_\gamma(x)$. According to Eq.(8) and CVMA Fig 1., the minimal polynomial $M_\gamma(x)$ is calculated by the algorithm Fig 2.
In Fig 2., Tr $[i]$ denote $\mathrm{Tr}^{[1]}(\gamma^i)$, $1 \leq i \leq m$, respectively. In addition, $x[j]$, $0 \leq j \leq m-1$ denote the vector coefficients of γ^{i-1}, $2 \leq i \leq m$ in each loop from line 7: to line 18:. Since TypeI-X NB is a normal basis, traces are efficiently calculated as shown at line 17:. Lines 1:, 2:, and 3: are preparations for CVMA in $\mathbb{F}_{p^m}$. From line 9: to line 17:, $\gamma^{i-1} \times \gamma$, $1 \leq i \leq m$ is calculated by modifying CVMA. This calculation is quite simpli_ed because the vector representation of the input γ is (1, 0, 0, … , 0). Then, at line 18: $\mathrm{Tr}^{[1]}(\gamma^i)$ is calculated. At line 16:, noting that k is small (T. Yoshida et al., (2006)), $kq[0]$ is calculated with k¡1 additions in $\mathbb{F}_p$. Thus, the calculation cost for the operation (a) becomes

$$\#_{\mathrm{SMUL}} = 0, \tag{14a}$$

$$\begin{aligned}\#_{\mathrm{SADD}} &= (m-1)(m-1) + (m-1)(m-1)k + (m-1)(k-1) + m(m-1) + (m+1)(m-1) \\ &= (m-1)\left(mk + 3m - 1\right).\end{aligned} \tag{14b}$$

In the right hand side of Eq.(14b), the five terms correspond to line 11:, 13:, 16:, 17:, and 18:, respectively. Thus, the operation (a) does not need any multiplications in $\mathbb{F}_p$, therefore the major computation is changed to the operation (b) shown from line 20: to line 26: in Fig 2., which needs $\mathcal{O}(m^2)$ $\mathbb{F}_p$-multiplications. Line 23: corresponds to Eq.(8). In detail, its calculation cost is evaluated as Eq.(13).
As shown in Fig 2., the proposed algorithm needs to calculate the indexes such as ⟨1 + (p^{j+mt})⟩; however, these indexes can be previously calculated when extension degree m is small. Of course, we can directly write down the program with the previously calculated indexes, therefore, the calculation cost for these indexes is not taken into account in this paper. By the way, according to Gao's paper (S. Gao et al., (2000)), when parameter k is even and divisible by p, TypeI-X NB becomes a self dual normal basis and thus $M_\gamma(x)$ is the minimal polynomial of the self dual normal basis in $\mathbb{F}_{p^m}$.
As introduced in Sec.1, we can of course calculate$M_\gamma(x)$ with the distributive law instead of Newton's formula as follows.

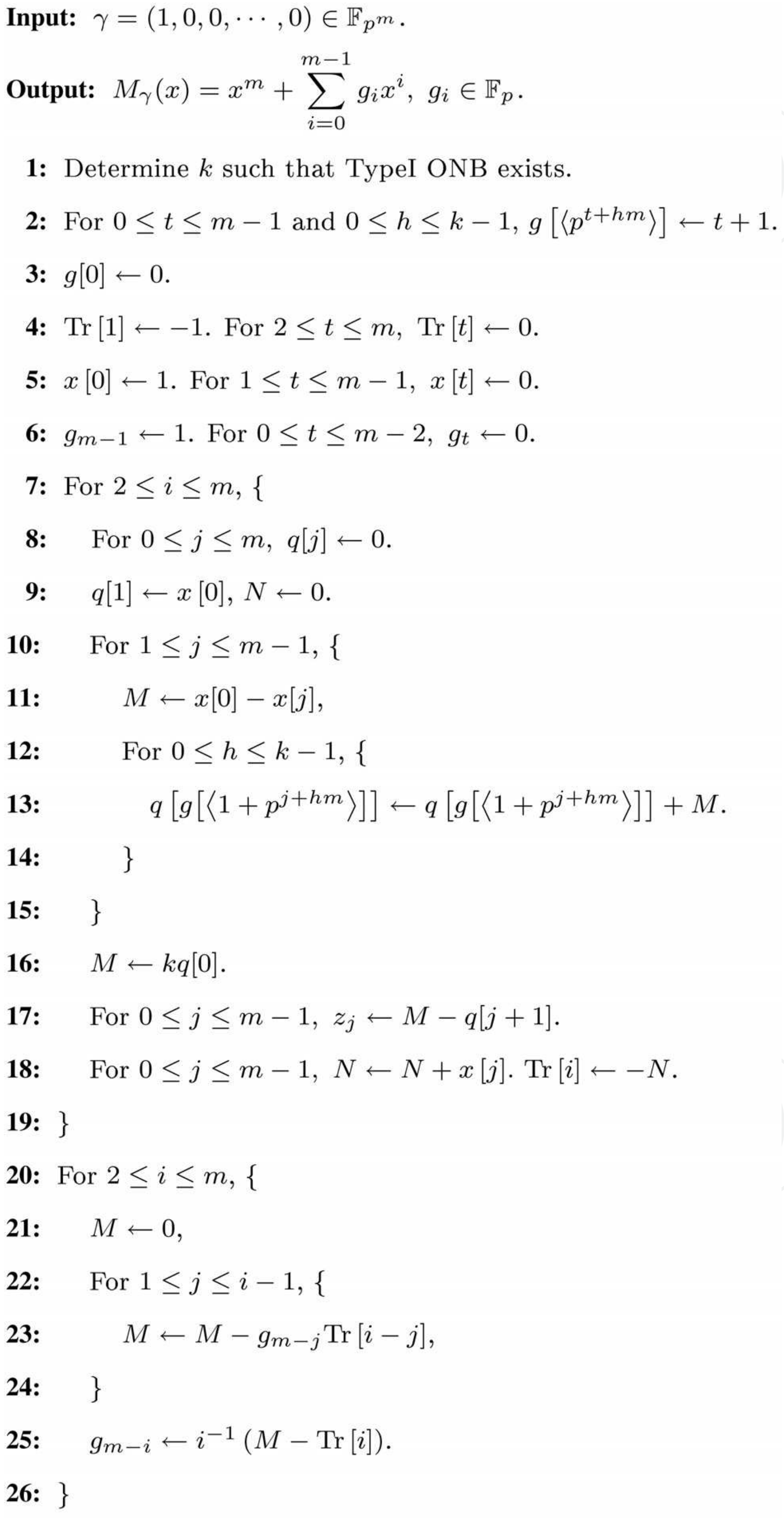
$$M_\gamma(x) = (x-\gamma)(x-\gamma^p)\cdots(x-\gamma^{p^{m-1}}). \tag{15}$$

In order to develop Eq.(15) in this case, we need $\times\gamma^{p^i}$ $m(m-1)/2$ times. Its calculation cost becomes

$$\#_{\mathrm{SMUL}} = 0, \tag{16a}$$

$$\#_{\mathrm{SADD}} = \frac{m(m-1)}{2}\left(mk + 3m - 1\right). \tag{16b}$$

Thus, it needs $\mathcal{O}(m^3)$ $\mathbb{F}_p$–additions. It does not need any $\mathbb{F}_p$–multiplications; however, the proposed algorithm becomes faster than using the distributive law as extension degree m becomes larger.



```
Input: $\gamma = (1, 0, 0, \cdots, 0) \in \mathbb{F}_{p^m}$.
Output: $M_\gamma(x) = x^m + \sum_{i=0}^{m-1} g_i x^i,\ g_i \in \mathbb{F}_p$.
Determine $k$ such that TypeI ONB exists.
For $0 \le t \le m-1$ and $0 \le h \le k-1$, $g\left[\langle p^{t+hm} \rangle\right] \leftarrow t+1$.
$g[0] \leftarrow 0$.
$\mathrm{Tr}\,[1] \leftarrow -1$. For $2 \le t \le m,\ \mathrm{Tr}\,[t] \leftarrow 0$.
$x\,[0] \leftarrow 1$. For $1 \le t \le m-1,\ x\,[t] \leftarrow 0$.
$g_{m-1} \leftarrow 1$. For $0 \le t \le m-2,\ g_t \leftarrow 0$.
For $2 \le i \le m$, {
    For $0 \le j \le m,\ q[j] \leftarrow 0$.
    $q[1] \leftarrow x\,[0],\ N \leftarrow 0$.
    For $1 \le j \le m-1$, {
        $M \leftarrow x[0] - x[j]$,
        For $0 \le h \le k-1$, {
            $q\left[g\left[\langle 1 + p^{j+hm} \rangle\right]\right] \leftarrow q\left[g\left[\langle 1 + p^{j+hm} \rangle\right]\right] + M$.
        }
    }
    $M \leftarrow kq[0]$.
    For $0 \le j \le m-1,\ z_j \leftarrow M - q[j+1]$.
    For $0 \le j \le m-1,\ N \leftarrow N + x\,[j]$. $\mathrm{Tr}\,[i] \leftarrow -N$.
}
For $2 \le i \le m$, {
    $M \leftarrow 0$,
    For $1 \le j \le i-1$, {
        $M \leftarrow M - g_{m-j}\mathrm{Tr}\,[i-j]$,
    }
    $g_{m-i} \leftarrow i^{-1}\,(M - \mathrm{Tr}\,[i])$.
}
```

(End of algorithm)

Fig. 2. Calculation of the minimal polynomial $M_\gamma(x)$

4. Consideration

This section shows some experimental results and comparison.

4.1 Experimental result and comparison

The authors have simulated the proposed algorithm on Pentium4 (1.7GHz) using C++ programming language and NTL (NTL). The authors also simulated the DDF-based irreducibility test which is introduced in Sec.2 and the case using the distributive law.

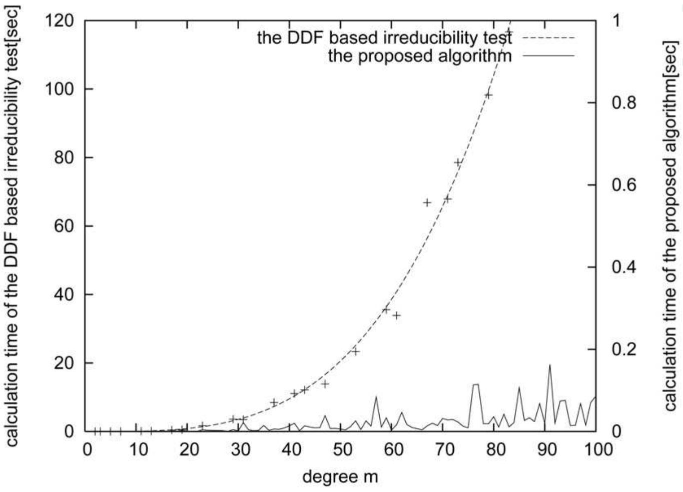


Fig. 3. The average computation time for generating an irreducible polynomial with the DDF-based irreducibility test and the proposed algorithm with p_1.

Let p_1 and p_2 be respectively given as follows:

$$p_1 = 65479 \text{ (16–bit)}, \tag{17}$$

$$p_2 = 1021076650872657639182783768587758285335306012183 \text{ (160–bit)}. \tag{18}$$

- For the proposed algorithm, the authors measured the average computation time for generating an irreducible polynomial of degree m over $\mathbb{F}_p$ by changing m from 2 to 100 with p_1 and p_2.
- For the DDF-based irreducibility test, inputting randomly generated polynomials of degree m over $\mathbb{F}_p$, the authors measured the average computation time for generating an irreducible polynomial with p_1 and the following prime degrees:

$$\{2,3,5,7,11,13,17,19,23,29,31,37,41,\\ 43,47,53,59,61,67,71,73,79,83\}, \tag{19}$$

and then with p_2 and the following prime degrees:

$$\{7,11,13,17,19,23,29,31,37,41,43,47\}. \tag{20}$$

Note that the irreducibility test is carried out by Eq.(4) when m is a prime number.

- For the case using the distributive law, the authors also measured the computation time for generating an irreducible polynomial of degree m over $\mathbb{F}_p$ by changing m from 2 to 100 with p_1.

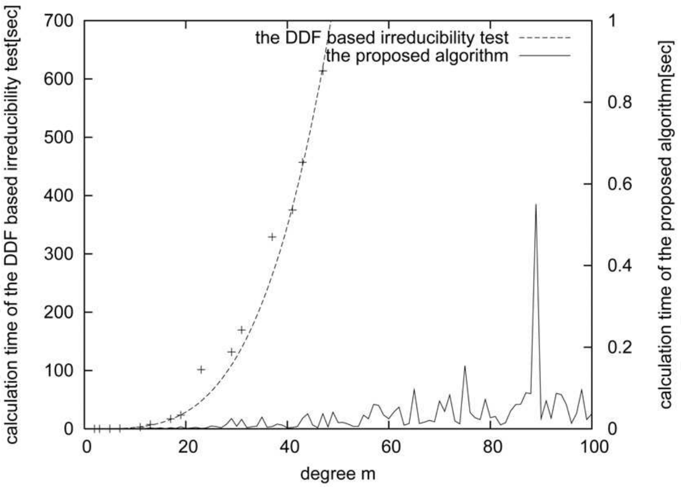


Fig. 4. The average computation time for generating an irreducible polynomial with the DDF-based irreducibility test and the proposed algorithm with p_2.

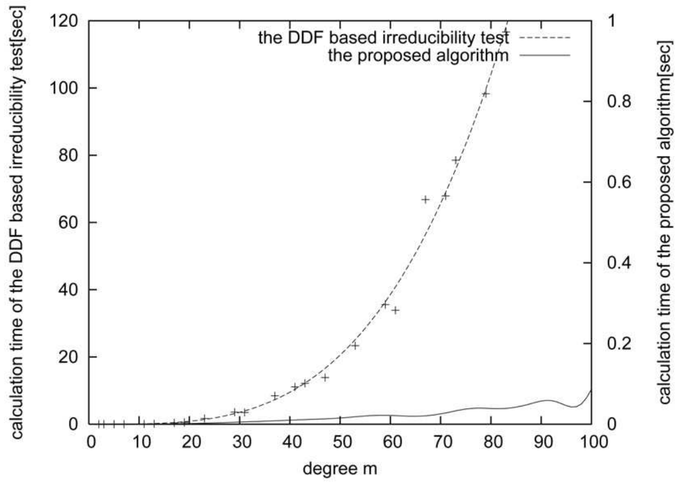


Fig. 5. The Bezier curve for the proposed algorithm in Fig 3.

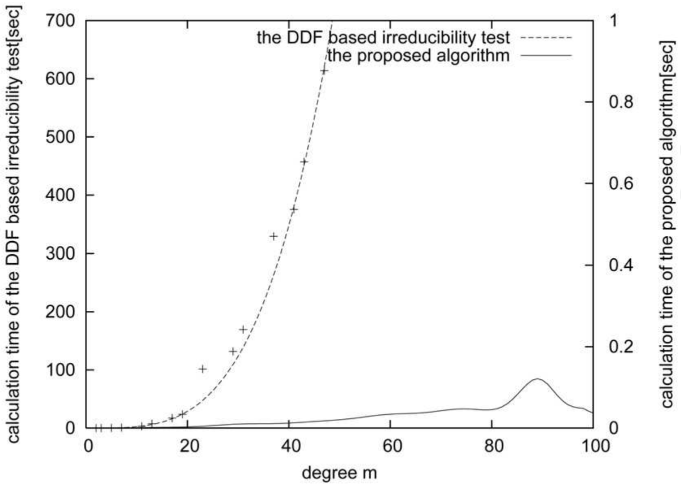


Fig. 6. The Bezier curve for the proposed algorithm in Fig 4.

The reason why the authors choose p_1 and p_2 given above is that the former does not need multi−precision arithmetic operations and the latter is sufficient secure size for elliptic curve

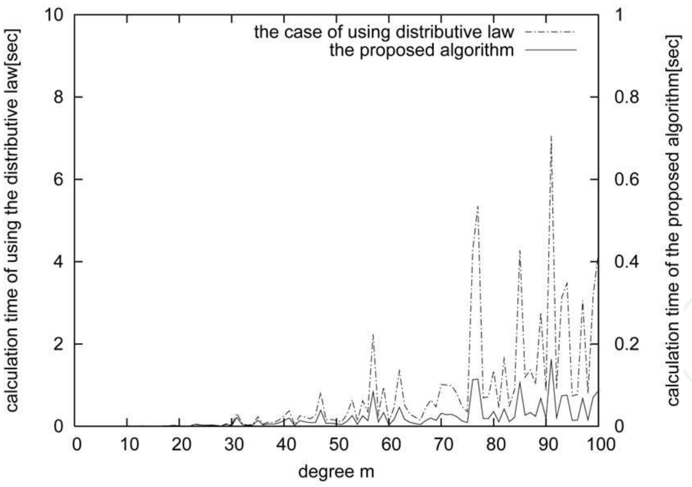


Fig. 7. The average computation time for generating an irreducible polynomial with the distributive law as Eq.(15) and the proposed algorithm with p_1.

cryptography (A. J. Menezes (1993)). Fig 3., Fig 4. and Fig 7. show the result. In Fig 3. and Fig 4., for example, when m = 83 with p_1, the proposed algorithm took 0.011 seconds with the parameter k = 2 and the DDF-based irreducibility test took 117 seconds. The proposed algorithm is about 10^4 times faster. As shown in the graphs, there are a few cases that the proposed algorithm is not very fast. For example, when m = 77 and 78 with p_1, the proposed algorithm took 0.115 and 0.019 seconds, respectively. The latter case is quite faster than the former. It is because of the parameter k. In the former case, k was 30, on the other hand, in the latter case, k was 4. Thus, the parameter k is preferred to be small. Of course, since the calculation cost of the proposed algorithm is clearly given as Eqs.(13) and Eqs.(14), in advance we can easily estimate how much calculation time the proposed algorithm needs. When the characteristic is p_1, the average of k's was 13.6. When the characteristic is p_2, the average was 12.8.

Fig 7. shows the comparison of the proposed algorithm and the case using the distributive law. For example, when m = 83 with p_1, the proposed algorithm took 0.011 seconds and the case using the distributive law took 0.496 seconds. The proposed algorithm is about 45 times faster. Fig 7. shows that the proposed algorithm is faster than using the distributive law. Therefore, using Newton's formula is better; however, it restricts p and m such that $p > m$.

As compared to the DDF-based irreducibility test, the proposed algorithm does not depend on the size of the characteristic p. It is because the calculation cost of the DDF-based irreducibility test depends on the size of p as introduced in Sec.2.5; however, that of the proposed algorithm does not. Therefore, as shown in Fig 3. and Fig 4., when p is large, the proposed algorithm generates an irreducible polynomial much faster than using the DDF-based irreducibility test. Fig 5. and Fig 6. are the Bezier curves for the data of the proposed algorithm in Fig 3. and Fig 4., respectively.

5. Conclusion

This paper has shown an efficient algorithm for generating an irreducible polynomial of an arbitrary degree m over an arbitrary prime field $\mathbb{F}_p$ such that $p > m$.

6. References

L. M. Adleman and H. W. Lenstra (1986). Finding irreducible polynomials over finite fields, *Proc. Of the eighteenth annual ACM Symp. on Theory of computing*, pp. 350-355.

Ian. F. Blake, S. Gao, and R. Lambert (1993). Constructive problems for irreducible polynomials over finite fields, *Proc. of the third Canadian workshop on Information theory and applications*, pp. 1-23.

M. Scott (2006). Scaling security in pairing-based protocols, available at http://mirror.cr.yp.to/eprint.iacr.org/2005/139.pdf, Cryptology Archive, ePrint.

V. Shoup (1990). New algorithms for finding irreducible polynomials over finite fields, *Math. of Comp*. vol. 54, pp. 435-447.

S. Gao (1993). Normal Bases over Finite Fields, *Doctoral thesis*, University of Waterloo, Ontario, Canada.

R. Lidl and H. Niederreiter (1984). Finite Fields, Encyclopedia of Mathematics and Its Applications, Cambridge University Press.

K. Makita, Y. Nogami, and T. Sugimura (2005). Generating Prime Degree Irreducible Polynomials by Usig Irreducible All-One Polynomial over F_2, *Electronics and Communications in Japan*, Part III : Fundamental Electronic Science, vol. 88, no. 7, pp. 23-32.

Y. Nogami, A. Saito, and Y. Morikawa (2003). Finite Extension Field with Modulus of All-One Polynomial and Representation of Its Elements for Fast Arithmetic Operations, *IEICE Trans.*, vol. E86-A, no. 9, pp. 2376-2387.

D. Bailey and C. Paar (2000). Optimal Extension Fields for Fast Arithmetic in Public-Key Algorithms, *Proc. Asiacrypt2000*, LNCS 1976, pp.248-258.

T. Yoshida, H. Katou, Y. Nogami, and Y. Morikawa (2006). Extension fields for an arbitrary pair of the characteristic and extension degree, *Proc. of Computer Security Symposium 2006 (CSS2006)*, pp. 43-48, in Japanese.

J. Gathen, and D. Panario (2001). Factoring Polynomials Over Finite Fields: A Survey, *J. Symbolic Computation*, vol. 31, pp. 3-17.

E. Berlekamp (1968). Algebraic Coding Theory, McGraw-Hill.

Y. Nogami, K. Tanaka, T. Sugimura, and S. Oshita (1999). Deriving In_nite Number of Irreducible Polynomials by Variable Transformation $x^p - x + s$, *Trans. of IEICE (A)*, vol. J82-A, no. 4, pp. 587-590, in Japanese.

T. Sugimura and Y. Suetugu (1991). Considerations on Irreducible Cyclotomic Polynomials, *Electronics and Communications in Japan*, Part3, vol. 74, no. 4, pp.106-113.

D. Knuth (1981). The Art of Computer Programming, *vol.2: Seminumerical Algorithms*, Addison-Wesley.

A. J. Menezes, editor (1993). Applications of Finite Fields, *Kluwer Academic Publishers*, Boston, MA.

Y. Nogami, S. Shinonaga, and Y. Morikawa (2005). Fast Implementation of Extension Fields with TypeII ONB and Cyclic Vector Multiplication Algorithm, *IEICE Trans. Fundamentals*, vol. E88-A, no. 5, pp. 1200-1208.

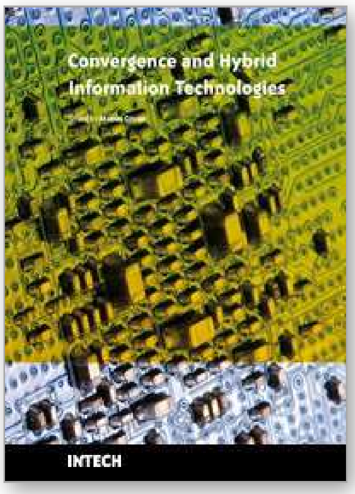
M. Nöcker (2001). Data Structures for Parallel Exponentiation in Finite Fields, available at http://deposit.ddb.de/.

R. Granger (2005). On Small Degree Extension Fields in Cryptology, available at http://www.cs.bris.ac.uk/Publications/Papers/2000527.pdf.

S. Gao, J. Gathen, D. Panario, and V. Shoup (2000). Algorithms for exponentiation in finite fields, *J. Symb. Comput*. 29, no. 6, pp. 879-889.

A Library for doing Number Theory., http://www.shoup.net/ntl/

A. J. Menezes (1993). Elliptic Curve Public Key Cryptosystems, *Kluwer Academic Publishers*.



Convergence and Hybrid Information Technologies

Edited by Marius Crisan

ISBN 978-953-307-068-1
Hard cover, 426 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

Starting a journey on the new path of converging information technologies is the aim of the present book. Extended on 27 chapters, the book provides the reader with some leading-edge research results regarding algorithms and information models, software frameworks, multimedia, information security, communication networks, and applications. Information technologies are only at the dawn of a massive transformation and adaptation to the complex demands of the new upcoming information society. It is not possible to achieve a thorough view of the field in one book. Nonetheless, the editor hopes that the book can at least offer the first step into the convergence domain of information technologies, and the reader will find it instructive and stimulating.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821