J. Cryptol.
DOI: 10.1007/s00145-009-9037-2

Journal of
CRYPTOLOGY

On the Efficient Generation of Prime-Order Elliptic Curves*

Elisavet Konstantinou
Department of Information and Communication Systems Engineering, University of the Aegean, 83200, Samos, Greece
ekonstantinou@aegean.gr
and
Computer Technology Institute, N. Kazantzaki Str, Patras University Campus, 26500 Patras, Greece

Aristides Kontogeorgis
Department of Mathematics, University of the Aegean, 83200, Samos, Greece
kontogar@aegean.gr

Yannis C Stamatiou
Department of Mathematics, University of Ioannina, 45110, Ioannina, Greece
istamat@cc.uoi.gr
and
Computer Technology Institute, N. Kazantzaki Str, Patras University Campus, 26500 Patras, Greece

Christos Zaroliagis
Department of Computer Engineering and Informatics, University of Patras, 26500 Patras, Greece
zaro@ceid.upatras.gr
and
Computer Technology Institute, N. Kazantzaki Str, Patras University Campus, 26500 Patras, Greece

Communicated by Johannes Buchmann

Received 9 September 2008 and revised 13 February 2009

Abstract. We consider the generation of prime-order elliptic curves (ECs) over a prime field $\mathbb{F}_p$ using the Complex Multiplication (CM) method. A crucial step of this method is to compute the roots of a special type of class field polynomials with the most commonly used being the Hilbert and Weber ones. These polynomials are uniquely determined by the CM discriminant D. In this paper, we consider a variant of the CM method for constructing elliptic curves (ECs) of prime order using Weber polynomials. In attempting to construct prime-order ECs using Weber polynomials, two difficulties arise (in addition to the necessary transformations of the roots of such polynomials to those of their Hilbert counterparts). The first one is that the requirement of prime order necessitates that $D \equiv 3 \pmod 8$, which gives Weber polynomials with degree

* This work was partially supported by the IST Programme of EU under contracts no. IST-2001-33116 (FLAGS), and by the Action IRAKLITOS (Fellowships for Research in the University of Patras) with matching funds from ESF (European Social Fund) and the Greek Ministry of Education.

© International Association for Cryptologic Research 2009

Published online: 31 March 2009

three times larger than the degree of their corresponding Hilbert polynomials (a fact that could affect efficiency). The second difficulty is that these Weber polynomials do not have roots in $\mathbb{F}_p$.

In this work, we show how to overcome the above difficulties and provide efficient methods for generating ECs of prime order focusing on their support by a thorough experimental study. In particular, we show that such Weber polynomials have roots in the extension field $\mathbb{F}_{p^3}$ and present a set of transformations for mapping roots of Weber polynomials in $\mathbb{F}_{p^3}$ to roots of their corresponding Hilbert polynomials in $\mathbb{F}_p$. We also show how an alternative class of polynomials, with degree equal to their corresponding Hilbert counterparts (and hence having roots in $\mathbb{F}_p$), can be used in the CM method to generate prime-order ECs. We conduct an extensive experimental study comparing the efficiency of using this alternative class against the use of the aforementioned Weber polynomials. Finally, we investigate the time efficiency of the CM variant under four different implementations of a crucial step of the variant and demonstrate the superiority of two of them.

Key words. Public key cryptography, Elliptic curve cryptosystems, Complex multiplication, Weber polynomials, Prime order

1. Introduction

Elliptic Curve (EC) cryptography has proven to be an attractive alternative for building fast and secure public key cryptosystems. Elliptic curves give rise to algebraic structures that offer a number of distinct advantages (smaller key sizes and highest strength per bit) over more customary algebraic structures used in various cryptographic applications (e.g., RSA). The use of smaller parameters for a given level of cryptographic strength makes them suitable for implementations on hardware devices of limited resources (e.g., memory, computing speed, bandwidth, etc.).

One of the fundamental problems in EC cryptography is the generation of cryptographically secure ECs over prime fields, suitable for use in various cryptographic applications. A typical requirement of all such applications is that the *order* of the EC (number of elements in the algebraic structure induced by the EC) possesses certain properties (e.g., robustness against known attacks [6], small prime factors [1], etc.), which give rise to the problem of how such ECs can be generated.

One of the most efficient methods that can be employed for the construction of ECs with specified order is the *Complex Multiplication* (CM) method [1,24]. In the case of prime fields, the CM method takes as input a given prime (the field's order) and determines a specific parameter, called the *CM discriminant* D of the EC. The EC of the desirable order is generated by constructing certain class field polynomials based on D and finding their roots. The construction and location of the roots (modulo the finite field's order) is one of the most crucial steps in the whole process. The most commonly used class field polynomials are the Hilbert (original version of the CM method) and the Weber polynomials. Their main differences are: (i) the coefficients of Hilbert polynomials grow unboundedly as D increases, while for the same D, the Weber polynomials have much smaller coefficients and thus are easier and faster to construct; (ii) the roots of a Hilbert polynomial construct directly the EC, while the roots of a Weber polynomial have to be transformed to the roots of its corresponding Hilbert polynomial to construct the EC. For a general discussion and comparisons between class field polynomials, see [11].

The CM method is not by itself adequate for applications that require robust ECs against cryptanalytic attacks. It turns out that the properties of the order of an EC play a central role in establishing cryptanalytic robustness. One way to establish robustness is to generate ECs whose order satisfies a certain number of properties designed to guard against the currently known most effective attacks [14,26,32,33]. An additional and equally important property that contributes to the cryptographic strength (see, e.g., [34]) requires that the order of the generated EC is a prime number. Note that in certain applications it is necessary to have ECs of prime order [7]. Prime-order ECs defined in various fields were also treated in [2,22,28,31].

In this paper, we follow the second approach and study the use of the CM method for generating ECs of prime order in $\mathbb{F}_p$. Although ECs with no restrictions on their order may be generated more efficiently using a point counting (such as Schoof's [36]) algorithm,[1] the requirement of prime order can severely change the situation. Point counting algorithms first choose the parameters of the EC and then compute its order. If this order is found non-prime, then another set of EC parameters is generated, and the process is repeated. This can be seen, approximately, as sampling from the set of ECs of prime order (for a fixed p). There is well-supported theoretical and experimental evidence [15] that this probability is, asymptotically, $\frac{c_p}{\log p}$, where c_p is a constant depending on p and satisfying $0.44 \leq c_p \leq 0.62$. Thus, it appears that prime orders are not especially favored by the point counting approach, as also noted in [15]. CM, on the other hand, starts with a prime number (the order of the EC) and *then* constructs the parameters, thus avoiding this averse prime order probability.

The use of Hilbert polynomials in the CM method requires high precision in the arithmetic operations involved in their construction, resulting in considerable increase in computing resources. This makes them rather inappropriate for fast and frequent generation of ECs. To overcome these shortcomings of Hilbert polynomials, two alternatives have been recently proposed: either to compute them off-line in powerful machines and store them for subsequent use (see, e.g., [34]), or to use Weber polynomials for certain values of D (see, e.g., [3,4,19,21,24,41]) and produce the required Hilbert roots from them. The former approach [34] tackles adequately the efficient construction of ECs, setting as a sole requirement for cryptographic strength that the order of the EC is prime which in turn implies that $D \equiv 3 \pmod 8$. However, there may still be problems with storing and handling several Hilbert polynomials with huge coefficients on hardware devices with limited resources. These problems are addressed by the second approach. Despite the space and time efficiency though, the known studies do not treat the case of $D \equiv 3 \pmod 8$ as these values of D give Weber polynomials with a degree three times larger than that of their corresponding Hilbert polynomial. For example, the case of $D \equiv 7 \pmod 8$ and not divisible by 3 is treated in [3,4,19,24], while the cases of $D \not\equiv 3 \pmod 8$ and $D \not\equiv 0 \pmod 3$ were treated in [21,41]. In addition, there are works that consider the generation of prime-order ECs over extension fields, but either they use the CM method with Hilbert polynomials [2], or they generate the EC parameters at random and use a point counting algorithm to compute the order of the curve [31]. To the best of our knowledge, the use of Weber polynomials within the CM method for the generation

[1] There are cases where point counting algorithms can be very inefficient compared to the CM method, e.g., when p is large and the discriminant value is small.

of prime-order ECs along with the necessary transformation of the Weber roots to their Hilbert counterparts for the case $D \equiv 3 \pmod 8$ has not been studied before.

In attempting to construct prime-order ECs using Weber polynomials, two additional difficulties arise. The first one is that the prime order requirement necessitates that $D \equiv 3 \pmod 8$, which in turn results in Weber polynomials with degree three times larger than the degree of their corresponding Hilbert polynomial. The second and most crucial difficulty is that such Weber polynomials (used for the construction of prime-order ECs) do not have roots in $\mathbb{F}_p$ for certain values of p, as is shown in Sect. 3.

Our work addresses the difficulties outlined above with an eye to applications and the practitioner's needs. We focus on supporting the theoretical findings with a thorough experimental study, thus shedding more light in the use of polynomials for the efficient generation of prime-order ECs using the CM method and providing guidance to the practitioner with respect to the resolution of these difficulties. In particular, we make the following contributions:

(i) We show that Weber polynomials defined on values of $D \equiv 3 \pmod 8$ and used in the CM method for generating ECs of prime order have roots in the extension field $\mathbb{F}_{p^3}$ and not in $\mathbb{F}_p$.
(ii) We present a set of simplified transformations that map the roots of the Weber polynomials in $\mathbb{F}_{p^3}$ to the roots of their corresponding Hilbert polynomials in $\mathbb{F}_p$. This implies that the particular Weber polynomials can be used to generate prime-order ECs with the CM method.
(iii) We show how an alternative class of polynomials can be used in the CM method for generating prime-order ECs. The advantage of these polynomials is that they have the same degree as their corresponding Hilbert polynomials and hence have roots in $\mathbb{F}_p$.
(iv) We perform a comparative experimental study regarding the efficiency of the CM method using the aforementioned Weber polynomials against using the alternative class of polynomials. Although it may seem that the use of Weber polynomials is inefficient due to their high degree and the fact that their roots lie in $\mathbb{F}_{p^3}$ (which requires operations with polynomials of degree 2), we provide experimental evidence which demonstrates that this is not always the case.

We would like to note that the case $D \equiv 3 \pmod 8$ can also be useful for the generation of ECs that do not necessarily have prime order [37] or for the generation of special curves, such as MNT curves [27,28]. This makes our analysis for class polynomials with such discriminants even more useful.

Another important step of the CM method is the determination of the order p of the underlying prime field and the construction of the order m of the EC. This step is independent of the computation of Hilbert or Weber polynomials. We consider four different ways for implementing this step in the CM method (Sect. 2). The first method is similar to that in [21] and uses the modified Cornacchia's algorithm [9]. The second method generates p and m at random as described in [34]. The third method is the very efficient algorithm given in Baier's PhD thesis [3, p. 68]. The fourth method, which we introduce here, resembles the third one and constitutes a simpler and more space-efficient alternative.

The final contribution of this paper is a comparative experimental study (Sect. 5) regarding the four methods mentioned above for the computation of p and m in the

construction of an EC. Comparing the four methods for computing p and m, Baier's method turns out to be the most time-efficient, followed very closely by the new method we present here. Hence, the latter could be used as a simpler, space-efficient, and easy-to-use alternative.

The rest of the paper is organized as follows. In Sect. 2 we review some basic definitions and facts about ECs, the CM method, and the variant we use, the Hilbert polynomials, and discuss some of their properties relevant to the generation of ECs. In Sect. 3 we present properties of Weber polynomials with $D \equiv 3 \pmod 8$ and describe their use in the CM method. In Sect. 4 we elaborate on the construction of an alternative class of polynomials that can also be used in the CM method. Finally, in Sect. 5 we present our experimental results, and we conclude in Sect. 6. Preliminary parts of this work appeared in [22,23].

2. A Brief Overview of Elliptic Curve Theory and Complex Multiplication

This section contains a brief introduction to elliptic curve theory, to the Complex Multiplication method for generating prime order elliptic curves, and to the Hilbert class field polynomials. Our aim is to facilitate the reading of the sections that follow. For full coverage of the necessary concepts and terms, the interested reader may consult [6]. Also, the proofs of certain theorems require basic knowledge of algebraic number theory and Galois theory. The interested reader is referred to [10,39,40] for definitions not given here.

2.1. *Preliminaries of Elliptic Curve Theory*

An *elliptic curve* defined over a finite field $\mathbb{F}_p$, $p > 3$ and prime, is denoted by $E(\mathbb{F}_p)$ and contains the points $(x, y) \in \mathbb{F}_p \times \mathbb{F}_p$ (in affine coordinates) that satisfy the equation (in $\mathbb{F}_p$)

$$y^2 = x^3 + ax + b \tag{1}$$

with $a, b \in \mathbb{F}_p$ satisfying $4a^3 + 27b^2 \neq 0$. The set of these points equipped with a properly defined point addition operation and a special point, denoted by $\mathcal{O}$ and called *point at infinity* (zero element for the addition operation), forms an Abelian group. This is the *Elliptic Curve group*, and the point $\mathcal{O}$ is its identity element (see [6,38] for more details on this group).

The *order*, denoted by m, is the number of points that belong to $E(\mathbb{F}_p)$. The numbers m and p are related by the *Frobenius trace* $t = p + 1 - m$. Hasse's theorem (see, e.g., [6,38]) implies that $|t| \leq 2\sqrt{p}$. Given a point $P \in E(\mathbb{F}_p)$, its *order* is the smallest positive integer n such that $nP = \mathcal{O}$. By Langrange's theorem, the order of a point $P \in E(\mathbb{F}_p)$ divides the order m of the group $E(\mathbb{F}_p)$. Thus, $mP = \mathcal{O}$ for any $P \in E(\mathbb{F}_p)$ and, consequently, the order of a point is always less than or equal to the order of the elliptic curve.

Two of the most important quantities of an elliptic curve $E(\mathbb{F}_p)$ defined through (1) are the *curve discriminant* Δ and the *j-invariant*: $\Delta = -16(4a^3 + 27b^2)$ and $j = -1728(4a)^3/\Delta$. Given $j_0 \in \mathbb{F}_p$ ($j_0 \neq 0, 1728$), two ECs of j-invariant j_0 can be easily constructed. If $k = j_0/(1728 - j_0) \bmod p$, one of these curves is given by (1) by setting

$a = 3k \bmod p$ and $b = 2k \bmod p$. The second curve (the *twist* of the first) is given by the equation

$$y^2 = x^3 + ac^2x + bc^3 \tag{2}$$

with c any quadratic non-residue of $\mathbb{F}_p$. If m_1 and m_2 denote the orders of an elliptic curve and its twist respectively, then $m_1 + m_2 = 2p + 2$, which implies that if one of the curves has order $p + 1 - t$, then its twist has order $p + 1 + t$, or vice versa (see [6, Lemma VIII.3]).

2.2. *The Complex Multiplication Method and a Variant*

As stated in the previous section, given a j-invariant one may readily construct an EC. Finding a suitable j-invariant for a curve that has a given order m can be accomplished through the theory of *Complex Multiplication* (CM) of elliptic curves over the rationals. This method is called the *CM method*, and in what follows we will give a brief account of it.

By Hasse's theorem, $Z = 4p - (p + 1 - m)^2$ is nonnegative, and, thus, there is a unique factorization $Z = Dv^2$ with D a square-free positive integer. Therefore,

$$4p = u^2 + Dv^2 \tag{3}$$

for some integer u that satisfies the equation

$$m = p + 1 \pm u. \tag{4}$$

The negative parameter $-D$ is called a *CM discriminant for the prime* p. For convenience throughout the paper, we will use (the positive integer) D to refer to the CM discriminant. The CM method uses D to determine a j-invariant. This j-invariant in turn, will lead to the construction of an EC of order $p + 1 - u$ or $p + 1 + u$.

The CM method requires as input a prime p. Then the smallest D is chosen that along with integers u, v satisfy (3). The next step is to check whether $p + 1 - u$ and/or $p + 1 + u$ is a suitable order. If none of them is suitable, then the whole process is repeated with another prime p as input. If one, however, is found to be suitable, then the Hilbert polynomial (see Sect. 2.3) is constructed, and its roots (modulo p) are computed. A root of the Hilbert polynomial is the j-invariant we are seeking. Then, the EC and its twist are constructed as explained in Sect. 2.1. Since only one of these ECs has the required suitable order, it can be found using Langrange's theorem by picking random points P in each EC until a point is found in some curve for which $mP \neq \mathcal{O}$. Then, the other curve is the one we are seeking.

In general, the most time-consuming part of the CM method is the construction of the Hilbert polynomial, as it requires high-precision floating-point arithmetic in the field of complex numbers. In order to overcome the high computational requirements of this construction, a variant of the CM method was proposed in [34]. In contrast with the CM method described above, this variant does not start with a specific p but with a CM discriminant $D \equiv 3 \pmod 8$, since it requires that the EC order m is prime (it is not hard to verify this constraint on D). It then computes p and the EC order m (the primality of m is the only requirement for cryptographic strength set in [34]). The prime

p is found by first picking randomly u and v of appropriate sizes and then checking if $(u^2 + Dv^2)/4$ is prime. An important aspect of the variant concerns the computation of the Hilbert polynomials: since they depend only on D (and not on p), they can be constructed in a preprocessing phase and stored for later use. Hence, the burden of their construction can be excluded from the generation of the EC.

In [21], another variant to the CM method was given which uses Weber polynomials. This variant starts with a discriminant $D \not\equiv 3 \pmod 8$ and a specific prime p chosen at random, or from a set of prescribed primes. It then computes u and v using Cornacchia's algorithm [9] to solve (3) and requires that the resulting EC order m is suitable (cf. Sect. 2.1) but not necessarily prime. Moreover, like in [34], the Weber polynomials can be constructed in a preprocessing phase as they also depend only on D.

In the rest of the section, we shall describe yet another variant of the CM method which shares similarities with those in [21,34] but also differs from them in several aspects. The new variant generates ECs of *prime and suitable* order, hence taking as input values of D which are congruent to $3 \pmod 8$, and determines the pair (u, v) that specifies p using four alternative implementations. Moreover, since Weber polynomials are used, which for these values of D, have a degree that is three times the degree of their corresponding Hilbert polynomials, a new transformation is presented for transforming Weber roots to Hilbert roots for this case (Sect. 3).

We are now ready to present the main steps of the variant. It starts with a CM discriminant $D \equiv 3 \pmod 8$ for the computation of the Weber polynomial,[2] and then generates at random, or selects from a pool of precomputed *good* primes (e.g., Mersenne primes), a prime p and computes odd integers u, v such that $4p = u^2 + Dv^2$. Those odd integers u, v can be computed with four different ways, which we will outline below. If no such numbers u and v can be found, then take another prime p and repeat. Otherwise, proceed with the next steps, which are similar to those of the original CM method. In particular, a Weber polynomial corresponding to the discriminant value D is constructed, and we locate a root of it. This root, however, cannot lead to the construction of the j-invariant directly, since j-invariants are roots of the Hilbert polynomials. Therefore, we must transform this root to a root of the corresponding (constructed with the same discriminant) Hilbert polynomial. The necessary transformations are given in Sect. 3.

We now turn to the four different methods for computing u and v. The first is to use the modified Cornacchia's algorithm [8]. In particular, a prime p is chosen at random, or from a set of prescribed primes, and then the modified Cornacchia's algorithm is used in order to find a solution (u, v) to (3). If there is a solution to this equation, we check if the resulting EC order m is prime. If it is not or there is no solution to (3), then another prime p is chosen. The second method generates odd parameters u and v at random as it is done in [34]. Once these parameters are created, we must check if the number $p = (u^2 + Dv^2)/4$ is prime. If it is, then the order m is constructed. In the case that m or p are not prime, then new parameters u and v are generated at random, and the same process is followed. The third method was proposed in [3, p. 68] and uses some clever heuristic in order to speed up the discovery of a suitable prime p. This method follows the idea of the previous approach but improves it considerably by posing several restrictions to the choice of u and v in order to increase the possibility that p and m are

[2] Although the variant defaults to the use of Weber polynomials, Hilbert polynomials can be used as well.

prime numbers. Despite its efficiency, this approach is quite complicated and uses an auxiliary table and two sieving arrays. Motivated by this approach, we have developed a fourth method, which is simpler and does not use any auxiliary tables or sieving arrays. The method is outlined in the following paragraph.

From (3) and (4) we know that if we compute u and v such that $4p = u^2 + Dv^2$, then the order m of the EC is given either by $p + 1 - u$ or $p + 1 + u$ (recall that m should be prime). We will denote the former by m^- and the latter by m^+. Since m is prime, u and v must be odd. In addition, u and v should not have common divisors because then p would not be a prime. With this observation in mind, we start our method by randomly picking odd u and v of appropriate sizes such that $u = 210x + 1$ and $v = 210y + 105$, where x, y are random numbers. In this way, u and v do not have common divisors the numbers 3, 5, and 7 ($3 \cdot 5 \cdot 7 = 105$). Simply, we chose v to be an odd integer having as common divisors the numbers 3, 5, and 7 (thus, $v = 105 \cdot (2y + 1)$) and u to be an odd integer that certainly would not have these divisors (u could be any number of the form $210x + c$, where c is not a multiple of 3, 5, or 7). Then, we check whether $(u^2 + Dv^2)/4$ is prime. If it is, then we check for primality the quantities m^- and m^+. If $(u^2 + Dv^2)/4$ is not prime, then we add to u an integer keeping the same value for v, calculate a new value for p, and repeat the whole process. An issue arises here as to what integer we add to u. Note that when $u = 210x + 1 \equiv 1 \pmod 3$, then $p \equiv 1 \pmod 3$, $m^- \equiv 1 \pmod 3$ and $m^+ \equiv 0 \pmod 3$. Thus, only m^- can be a prime. If u were equal to $u = 210x + 107 \equiv 2 \pmod 3$, then again $p \equiv 1 \pmod 3$, but $m^- \equiv 0 \pmod 3$ and $m^+ \equiv 1 \pmod 3$. Therefore, at the first iteration of our method we select $u = 210x + 1$, at the second $u = 210x + 107$, and so on, in order to check for primality m^- and m^+ in tandem. In particular, if the choice $u = 210x + 1$ does not give primes p and m, then we add to u the number 106, in the next iteration we add 104, and so on. In this way, u is at one step congruent to 1 (mod 3) and at the next step congruent to 2 (mod 3).

As mentioned earlier, the other most complicated part of the CM method is the construction of the polynomials (Weber or Hilbert), which is addressed in the next section.

2.3. *Hilbert Polynomials*

Every CM discriminant D defines a unique Hilbert polynomial, denoted by $H_D(x)$. Given a CM discriminant D, the Hilbert polynomial $H_D(x) \in \mathbb{Z}[x]$ is defined as

$$H_D(x) = \prod_{\tau} \big(x - j(\tau)\big), \tag{5}$$

where $j(\tau) = \frac{(256h(\tau)+1)^3}{h(\tau)}$, $h(\tau) = \frac{\Delta(2\tau)}{\Delta(\tau)}$, $\Delta(\tau) = \eta(\tau)^{24} = q(1 + \sum_{n\geq 1}(-1)^n \times (q^{n(3n-1)/2} + q^{n(3n+1)/2}))^{24}$, and $q = e^{2\pi i \tau}$. The quantity $j(\tau)$ in (5) is called *class invariant*. Every value of τ is constructed from a 3-tuple of integers $[\alpha, \beta, \gamma]$ by the equation $\tau = (-\beta + \sqrt{-D})/(2\alpha)$ (notice that τ is a root of the quadratic equation $\alpha z^2 + \beta z + \gamma = 0$). Every such 3-tuple of integers is called a *primitive reduced quadratic form* of $-D$ and satisfies the following conditions: (i) $\beta^2 - 4\alpha\gamma = -D$, (ii) $|\beta| \leq \alpha \leq \sqrt{D/3}$, (iii) $\alpha \leq \gamma$, (iv) $\gcd(\alpha, \beta, \gamma) = 1$, and (v) if $|\beta| = \alpha$ or $\alpha = \gamma$, then $\beta \geq 0$. Clearly, the set of primitive reduced quadratic forms of a given discriminant is finite.

Let h be the number of primitive reduced quadratic forms (and thus the number of τ values), which determines the *degree* or *class number* of $H_D(x)$. Then, the bit precision required for the generation of $H_D(x)$ can be estimated (see [24]) by

$$\text{H-Prec}(D) \approx \frac{\ln 10}{\ln 2}(h/4+5) + \frac{\pi\sqrt{D}}{\ln 2}\sum_{\tau}\frac{1}{\alpha} \tag{6}$$

with the sum running over the same values of τ as the product in (5). Hilbert polynomials have roots modulo p under certain conditions stated in the following theorem.

Theorem 1. *A Hilbert polynomial $H_D(x)$ with degree h has exactly h roots modulo p if and only if the equation $4p = u^2 + Dv^2$ has integer solutions and p does not divide the discriminant $\Delta(H_D)$ of the polynomial.*

Proof. Let H_K be the Hilbert class field of the imaginary quadratic field $K = \mathbb{Q}(\sqrt{-D})$, and let $\mathcal{O}_{H_K}$ and $\mathcal{O}_K$ be the rings of algebraic integers of H_K and K, respectively.

Let p be a prime such that $4p = u^2 + Dv^2$ has integer solutions. Then, according to [10, Theorem 5.26], p splits completely in H_K. Let $H_D(x) \in \mathbb{Z}[x]$ be the Hilbert polynomial with root the real algebraic integer $j(\tau)$. Proposition 5.29 in [10] implies that $H_D(x)$ has a root modulo p if and only if p splits in H_K and does not divide its discriminant[3] $\Delta(H_D)$. However, since $\frac{\mathcal{O}_{H_K}}{p\mathcal{O}_{H_K}}/\mathbb{F}_p$ is Galois, $H_D(x)$ has not only one root modulo p but h distinct roots modulo p. □

There are finitely many primes dividing the discriminant $\Delta(H_D)$ of the Hilbert polynomial and infinitely many primes to choose. In elliptic curve cryptosystems the prime p is at least 160 bits. Therefore, an arbitrary prime almost certainly does not divide the discriminant.

3. The CM Method Using Weber Polynomials

In this section we define Weber polynomials for discriminant values $D \equiv 3 \pmod 8$ and prove that they do not have roots in $\mathbb{F}_p$ for certain primes p but do have roots in the extension field $\mathbb{F}_{p^3}$. We then discuss their efficiency when used in the CM method and present a transformation that maps roots of Weber polynomials in $\mathbb{F}_{p^3}$ into the roots of their Hilbert counterparts in $\mathbb{F}_p$.

[3] For a definition of the discriminant of a polynomial, see [8].

3.1. *Weber Polynomials and Their Roots in Finite Fields*

Weber polynomials are defined using the Weber functions (see [1,17]):

$$f(\tau) = q^{-1/48}\prod_{r=1}^{\infty}\left(1+q^{r-1/2}\right), \qquad f_1(\tau) = q^{-1/48}\prod_{r=1}^{\infty}\left(1-q^{r-1/2}\right),$$

$$f_2(\tau) = \sqrt{2}\ q^{1/24}\prod_{r=1}^{\infty}\left(1+q^{r}\right) \quad \text{where } q = e^{2\pi i\tau}.$$

The Weber polynomial $W_D(x) \in \mathbb{Z}[x]$ for $D \equiv 3 \pmod 8$ is defined as

$$W_D(x) = \prod_{\ell}\left(x - g(\ell)\right), \tag{7}$$

where $\ell = \frac{-b+\sqrt{-D}}{a}$ satisfies the equation $ay^2 + 2by + c = 0$ for which $b^2 - ac = -D$ and (i) $\gcd(a, b, c) = 1$, (ii) $|2b| \le a \le c$, and (iii) if either $a = |2b|$ or $a = c$, then $b \ge 0$. Let $\zeta = e^{\pi i/24}$. The class invariant $g(\ell)$ for $W_D(x)$ is defined by

$$g(\ell) = \begin{cases} \zeta^{b(c-a-a^2c)} \cdot f(\ell) & \text{if } 2 \nmid a \text{ and } 2 \nmid c, \\ -(-1)^{\frac{a^2-1}{8}} \cdot \zeta^{b(ac^2-a-2c)} \cdot f_1(\ell) & \text{if } 2 \nmid a \text{ and } 2 \mid c, \\ -(-1)^{\frac{c^2-1}{8}} \cdot \zeta^{b(c-a-5ac^2)} \cdot f_2(\ell) & \text{if } 2 \mid a \text{ and } 2 \nmid c, \end{cases} \tag{8}$$

if $D \equiv 3 \pmod 8$ and $D \not\equiv 0 \pmod 3$, and

$$g(\ell) = \begin{cases} \frac{1}{2}\zeta^{3b(c-a-a^2c)} \cdot f^3(\ell) & \text{if } 2 \nmid a \text{ and } 2 \nmid c, \\ -\frac{1}{2}(-1)^{\frac{3(a^2-1)}{8}} \cdot \zeta^{3b(ac^2-a-2c)} \cdot f_1^3(\ell) & \text{if } 2 \nmid a \text{ and } 2 \mid c, \\ -\frac{1}{2}(-1)^{\frac{3(c^2-1)}{8}} \cdot \zeta^{3b(c-a-5ac^2)} \cdot f_2^3(\ell) & \text{if } 2 \mid a \text{ and } 2 \nmid c, \end{cases} \tag{9}$$

if $D \equiv 3 \pmod 8$ and $D \equiv 0 \pmod 3$. The above equations were derived from [17] in an attempt to simplify the corresponding (and rather tedious) equations for the construction of Weber polynomials.

We would like to note that the conditions for the construction of the quadratic forms $[a, 2b, c]$ are slightly different from the conditions for the computation of the primitive quadratic forms $\left[\alpha, \beta, \gamma\right]$ needed in Sect. 2.3. This means that having as input the same value of D, the quadratic forms which are constructed for Hilbert and Weber polynomials are different. In particular, when $D \equiv 3 \pmod 8$, the number of these quadratic forms for Weber polynomials is three times larger than the corresponding number of quadratic forms for Hilbert polynomials.

Thus, for these cases of the discriminant ($D \equiv 3 \pmod 8$), the Weber polynomial $W_D(x)$ has degree three times larger than the degree of its corresponding Hilbert polynomial $H_D(x)$. An upper bound for the precision requirements of Weber polynomials for both cases of D was presented in [22] and is equal to $3h + \frac{\pi\sqrt{D}}{24\ln 2}\sum_{\ell}\frac{1}{\alpha}$ for $D \not\equiv 0 \pmod 3$ and to $3h + \frac{\pi\sqrt{D}}{8\ln 2}\sum_{\ell}\frac{1}{\alpha}$ for $D \equiv 0 \pmod 3$. The sum runs over the same values

of ℓ as the product of (7), and $3h$ is the degree of the Weber polynomial (h is the degree of the corresponding Hilbert polynomial).

Consider the modular function

$$\Phi_2(x, j) = (x - 16)^3 - jx \tag{10}$$

where j is a class invariant for the Hilbert polynomial. The three roots of the equation $\Phi_2(x, j) = 0$ are the powers f^{24}, $-f_1^{24}$, and $-f_2^{24}$ of the Weber functions (it is known that $j(z) = \frac{(f^{24}(z)-16)^3}{f^{24}(z)} = \frac{(f_1^{24}(z)+16)^3}{f_1^{24}(z)} = \frac{(f_2^{24}(z)+16)^3}{f_2^{24}(z)}$, see [6]). A transformation (used in the CM method) from roots of Weber polynomials to roots of Hilbert polynomials was presented in [22] and is derived from the modular equation $\Phi_2(x, j) = 0$. The transformation for $D \not\equiv 0 \pmod 3$ is

$$R_H = \frac{(2^{12} R_W^{-24} - 16)^3}{2^{12} R_W^{-24}} \tag{11}$$

and for $D \equiv 0 \pmod 3$ is

$$R_H = \frac{(2^4 R_W^{-8} - 16)^3}{2^4 R_W^{-8}} \tag{12}$$

where R_W is a root of $W_D(x)$, and R_H is a root of $H_D(x)$. To use these transformations, we have to locate R_W on a specific field, an issue not addressed in [22].

In the rest of this section we will show that when u, v are odd numbers and $D \equiv 3 \pmod 8$, then $W_D(x)$ does not have roots modulo p, but its roots belong to the extension field $\mathbb{F}_{p^3}$ (recall that the order $m = p + 1 \pm u$ of the elliptic curve can be prime only if u is odd, which means that in (3) v must be odd, too).

Theorem 2. *Let $D \equiv 3 \pmod 8$ and assume that the equation $4p = u^2 + Dv^2$ has a solution (u, v), where u, v are odd integers. Then, the Weber polynomial $W_D(x)$ with degree $3h$ has no roots modulo p.*

Proof. In order to prove that the Weber polynomial $W_D(x)$ has no roots modulo p, we must prove first that the equation $\Phi_2(x, j) = 0 \pmod p$ has no roots $x \pmod p$ for a given $j \pmod p$. Let j be a root of Hilbert polynomial modulo p (denoted also as R_H), which we know from Theorem 1 that always exists. According to (11) and (12), the modular equation $\Phi_2(x, j) = 0 \pmod p$ (for a given $j \pmod p$) has a solution $x = 2^{12} R_W^{-24}$ if $D \not\equiv 0 \pmod 3$ and $x = 2^4 R_W^{-8}$ if $D \equiv 0 \pmod 3$. If the Weber polynomial $W_D(x)$ had a root R_W modulo p, then $\Phi_2(x, j) = 0 \pmod p$ would also have a root x modulo p. Consequently, if we could prove that $\Phi_2(x, j) = 0 \pmod p$ has no roots x modulo p, then the same will be true for the Weber polynomial $W_D(x)$.

For an integer c, let $(\frac{c}{2})$ denote the *Kronecker* symbol. From [30, Theorem 3.1] we conclude that if $(\frac{-Dv^2}{2}) = -1$, then the polynomial $\Phi_2(x, j) \pmod p$ is irreducible modulo p. This means that in this case the equation $\Phi_2(x, j) = 0 \pmod p$ has no roots $x \pmod p$ for a given $j \pmod p$. Thus, it suffices to prove that $(\frac{-Dv^2}{2}) = -1$. Using the Kronecker symbol, we know that $(\frac{-Dv^2}{2}) = -1$ if $-Dv^2$ is odd and $-Dv^2 \equiv$

$\pm 3 \pmod 8$. We will show that $Dv^2 \equiv 3 \pmod 8$. Clearly, since $D \equiv 3 \pmod 8 = 8d_1 + 3$ and $v = 2v_1 + 1$ is odd, Dv^2 is also odd. We have $Dv^2 = (8d_1+3)(2v_1+1)^2 = (8d_1+3)(4v_1^2+4v_1+1)$. That is, $Dv^2 \equiv 3(4v_1^2+4v_1+1) \pmod 8$, and because $v_1^2+v_1$ is even, it is easily seen that $Dv^2 \equiv 3 \pmod 8$, which completes the proof. □

The next theorem establishes the main result of this section.

Theorem 3. *Let $D \equiv 3 \pmod 8$ and assume that the equation $4p = u^2 + Dv^2$ has a solution (u, v), where u, v are odd integers. Then, the Weber polynomial $W_D(x)$ has h monic irreducible factors of degree 3 modulo p and $3h$ roots in the extension field $\mathbb{F}_{p^3}$.*

Proof. We have proved in Theorem 2 that the Weber polynomial does not have roots modulo p if u, v are odd numbers and that the polynomial $\Phi_2(x, j)$ is irreducible modulo p. This means that $\Phi_2(x, j) = 0$ has three roots $x \in \mathbb{F}_{p^3}$ for a root $j \in \mathbb{F}_p$ of the Hilbert polynomial.[4] According to (11) and (12), $x = 2^{12} R_W^{-24}$ if $D \not\equiv 0 \pmod 3$, and $x = 2^4 R_W^{-8}$ if $D \equiv 0 \pmod 3$. Thus, there are at least three roots of the Weber polynomial that correspond to a root $j \in \mathbb{F}_p$ of the Hilbert polynomial and which are either in $\mathbb{F}_{p^3}$ or in an extension field of greater degree (at most 72 if $D \not\equiv 0 \pmod 3$ and at most 24 if $D \equiv 0 \pmod 3$).

Let $R_{W,j}$ be a root of the Weber polynomial that corresponds to a root j of the Hilbert polynomial. Let $f_j(x)$ be the minimal polynomial of $R_{W,j} \pmod p$. The degree of this polynomial will be at least 3, because the root $R_{W,j}$ is at least in $\mathbb{F}_{p^3}$. Then, the Weber polynomial can be written as

$$W_D(x) = \prod_j f_j(x) \pmod p. \tag{13}$$

Since the degree of the Weber polynomial is $3h$ and the roots j modulo p of the Hilbert polynomial are h (see Theorem 1) we have that every minimal polynomial $f_j(x)$ will have degree 3. Thus, Weber polynomials have h irreducible cubic factors. Every factor $f_j(x)$ has 3 roots in $\mathbb{F}_{p^3} \cong \mathbb{F}_p[x]/f_j(x)$, which means that there are totally $3h$ roots in $\mathbb{F}_{p^3}$. □

3.2. *The Use of Weber Polynomials in the CM Method*

In this subsection we will elaborate on the use of Weber polynomials for the generation of prime-order ECs. The idea is that we replace Hilbert polynomials with Weber polynomials and then try to compute a root of the Hilbert polynomial from a root of its corresponding Weber polynomial. To compute the desired Hilbert root, we proceed in three stages. First, we construct the corresponding Weber polynomial. Second, we compute its roots in $\mathbb{F}_{p^3}$. Finally, we transform the Weber roots to the desired Hilbert roots in $\mathbb{F}_p$. The first stage is accomplished using the definition of Weber polynomials in Sect. 3.1. To compute a root of $W_D(x)$ in $\mathbb{F}_{p^3}$, we have to find an irreducible factor (modulo p) of degree 3 of the polynomial. This is achieved using Algorithm 3.4.6

[4] Each root x is an element of $\mathbb{F}_{p^3}$ because $\Phi_2(x, j)$ is irreducible modulo p and $\mathbb{F}_{p^3}$ is isomorphic to $\mathbb{F}_p[x]/\Phi_2(x, j)$.

from [8]. The irreducible factor has 3 roots in $\mathbb{F}_{p^3}$, from which it suffices to choose one in order to accomplish the third stage.

Suppose that $x^3 + ax^2 + bx + c$ is an irreducible factor modulo p of the Weber polynomial. From this irreducible factor we can compute three roots (one suffices for the CM method) of the Weber polynomial if we have already defined the reduction polynomial of the extension field $\mathbb{F}_{p^3}$. We simply set the reduction polynomial to be equal to the irreducible factor $x^3 + ax^2 + bx + c$, and then a root of the Weber polynomial would be just x.

Let us see an example: if $W_{403}(x) = x^6 - 12x^5 - 26x^4 + 4x^3 + 36x^2 + 20x + 4$ and $p = 7221076618803527297111165735009$, then a factor of the Weber polynomial modulo p is $x^3 + 530841998355731959331093661138x^2 + 2654209991778659796655\ 46830567x + 7221076618803527\ 29711165735007$. Note that 403 is not divisible by 3 and that $7221076618803527297111165735007 = p - 2 \equiv -2 \pmod{p}$.

The following lemma allows us to determine the constant term of the irreducible factor and consequently to simplify the roots' transformation as we will see later.

Lemma 1. *Let $x^3 + ax^2 + bx + c$ be an irreducible factor (modulo p) of the Weber polynomial with $D \equiv 3 \pmod 8$. Then, the following hold*: (i) *if $D \equiv 0 \pmod 3$, then $c = -1$*; (ii) *if $D \not\equiv 0 \pmod 3$, then $c = -2$.*

Proof. The constant term of the Weber polynomial is equal to $(-1)^h$ for the first case of D and $(-2)^h$ for the second case (see [18]). The Galois group of the extension H_K/K operates on the roots modulo p of $H_D(x)$ and therefore on the cubic irreducible factors of $W_D(x)$ (every root of $H_D(x)$ corresponds to three roots of $W_D(x)$ and thus to a cubic irreducible factor). Since every element in this Galois group induces the identity on $\mathbb{F}_p$, all cubic factors of $W_D(x)$ will have the same constant term. Because the constant term of a monic polynomial is equal to the product of the constant terms of its monic irreducible factors, it can be easily seen that $c = -1$ for the first case of D and $c = -2$ for the second. □

We are now ready to present the transformations for mapping a Weber root in $\mathbb{F}_{p^3}$ to its corresponding Hilbert root in $\mathbb{F}_p$. Suppose that $R_W = x$ is a root of a Weber polynomial $W_D(x)$ in the extension field $\mathbb{F}_{p^3}$. The calculations in the transformations must be in $\mathbb{F}_{p^3}$ with reduction polynomial $x^3 + ax^2 + bx + c$, since R_W is a root in $\mathbb{F}_{p^3}$.

The transformations may seem quite complicated because of the arithmetic operations that take place in the extension field, but they can be simplified due to Lemma 1. Consider the case $D \not\equiv 0 \pmod 3$, for which an irreducible factor of the Weber polynomial is equal to $x^3 + ax^2 + bx - 2$. Then, $R_W^{-24} = x^{-24} = (\frac{x^2+ax+b}{x(x^2+ax+b)})^{24} = (\frac{x^2+ax+b}{2})^{24}$. This means that $2^{12}R_W^{-24} = \frac{(x^2+ax+b)^{24}}{2^{12}}$. Substituting it into Eq. (11), we finally have

$$R_H = \frac{((x^2 + ax + b)^{24} - 2^{16})^3}{2^{24}(x^2 + ax + b)^{24}}. \tag{14}$$

Similarly, for $D \equiv 0 \pmod 3$, the transformation becomes

$$R_H = \frac{2^8((x^2+ax+b)^8-1)^3}{(x^2+ax+b)^8}. \tag{15}$$

The nominator and denominator of the two transformations are elements of $\mathbb{F}_{p^3}$. However we know that R_H is in $\mathbb{F}_p$, and we can find its value dividing only the leading coefficients of these two elements modulo p. To illustrate the above transformations, consider again the Weber polynomial W_{403}. Let p be a prime as in the previous example, and let the reduction polynomial be the factor of the $W_{403}(x)$ presented also in the previous example. Then, $((x^2+ax+b)^{24}-2^{16})^3 = 485216670393361675137940525358x^2 + 498390024660218217560914441491x + 437\ 505083747867349301080018378$ and $(x^2+ax+b)^{24} = 372203635398289746518033\ 419220x^2 + 193471851293797158505478806686x + 105818622204842691408284289\ 782$. The root R_H of the Hilbert polynomial is equal to $\frac{485216670393361675137940525358}{2^{24}372203635398289746518033419220} \pmod p = 188541528108458443856585415294$.

4. The CM Method Using an Alternative Class of Polynomials

Even though Weber polynomials have much smaller coefficients than Hilbert polynomials and can be computed very efficiently, the fact that their degree for $D \equiv 3 \pmod 8$ is three times larger than the degree of the corresponding Hilbert polynomials can be a potential problem, because it involves computations in extension fields. Moreover, the computation of a cubic factor modulo p in a polynomial with degree $3h$ is more time consuming than the computation of a single root modulo p of a polynomial with degree h.

To alleviate these problems, we can use in the CM method a relatively new class of polynomials which have degree h like Hilbert polynomials. In particular, two types of polynomials can be constructed in $\mathbb{Z}[x]$ using two families of η-products: $m_l(z) = \frac{\eta(z/l)}{\eta(z)}$ [29] for an integer l, and $m_{p_1,p_2}(z) = \frac{\eta(z/p_1)\eta(z/p_2)}{\eta(z/(p_1p_2))\eta(z)}$ [12], where p_1, p_2 are primes such that $24|(p_1-1)(p_2-1)$. We will refer to the minimal polynomials of these products (powers of which generate the Hilbert class field and are called class invariants like $j(\tau)$) as $M_{D,l}(x)$ and $M_{D,p_1,p_2}(x)$, respectively, where D is the discriminant used for their construction.

The polynomials are obtained from these two families by evaluating their value at a suitably chosen system of quadratic forms. Once a polynomial is computed, we can use the modular equations $\Phi_l(x,j)=0$ or $\Phi_{p_1,p_2}(x,j)=0$, in order to compute a root modulo p of the Hilbert polynomial from a root modulo p of the $M_{D,l}(x)$ or the $M_{D,p_1,p_2}(x)$ polynomial, respectively. In this section we will construct polynomials using only the m_l family for prime values of l, in particular for $l = 3, 5, 7, 13$. The reason is that only for these values of l, the modular equations have degree 1 in j. For all other values of l or for the m_{p_1,p_2} family, the degree in j is at least 2 (which makes the computations more "heavy"), and the coefficients of the modular equations are quite large (which makes their use less efficient).[5]

[5] For example, notice in [13] the size of the smallest modular polynomial $\Phi_{5,7}(x,j)$.

Table 1. Class invariants for different values of l.

l	Class invariant
3	m_3^{12}
5	m_5^6
7	m_7^4
13	m_{13}^2

In order to construct the polynomial $M_{D,l}(x)$ with $l = 3, 5, 7, 13$, we used Theorem 2 from [11], which for our purposes boils down to the following statement.

Theorem 4. [11] *Let $l \in \{3, 5, 7, 13\}$ and $D > 0$ a discriminant such that $l|D$. Choose the power m_l^e as specified in Table 1. Assume that $Q = [A, B, C]$ is a primitive quadratic form of discriminant D with $\gcd(A, l) = 1$, $\gcd(A, B, C) = 1$, and $B^2 \equiv -D \pmod{4l}$. If $\tau_Q = \frac{-B+\sqrt{-D}}{2A}$, then the minimal polynomial of $m_l^e(\tau_Q)$ has integer coefficients and can be computed from an l-system.*

An l-system is a system $S = \{(A_i, B_i, C_i)\}_{1 \le i \le h}$ of representatives of the reduced primitive quadratic forms of a discriminant $-D$ such that $B_i^2 - 4A_iC_i = -D$, $\gcd(A_i, l) = 1$, and $B_r \equiv B_s \pmod{2l}$ for all $1 \le r, s \le h$. For a more formal definition, see [35].

Although the construction of $M_{D,l}(x)$ polynomials is explained in [11,29,30], the required computation of the primitive forms is not provided. In the following, we provide all the details for computing these forms, which we also used in our implementation. Possibly there are alternative ways to generate the same polynomial $M_{D,l}(x)$ with other, equivalent forms.

For the construction of the polynomials $M_{D,l}(x)$, according to Theorem 4, the condition $B_r \equiv B_s \pmod{2l}$ can be replaced by the condition $B_i^2 \equiv -D \pmod{4l}$, and because $D \equiv 0 \pmod{l}$, we can write $B_i = l + 2lk_i \equiv l \pmod{2l}$ for an integer $k_i \ge 1$. In particular, $M_{D,l}(x) = \prod_{\tau_Q}(x - m_l^e(\tau_Q))$, where $Q = [A_i, B_i, C_i]$ is a primitive form satisfying the conditions $\gcd(A_i, l) = 1$, $B_i = l + 2lk_i$, and $\tau_Q = \frac{-B_i+\sqrt{-D}}{2A_i}$. The set of forms $[A_i, B_i, C_i]_{1 \le i \le h}$ can be computed from the set of the reduced primitive quadratic forms $[\alpha, \beta, \gamma]$ that are used for the construction of $H_D(x)$.

A form $[A_i, B_i, C_i]$ can be computed from a reduced primitive quadratic form $[\alpha, \beta, \gamma]$ using (at most) two transformations from [35, Proposition 3]. The first one transforms a form $[a, b, c]$ to the equivalent (having the same discriminant $-D$) form $[a, b + 2ak, c + bk + ak^2]$ for an integer k, and the second transforms a form $[a, b, c]$ to the equivalent form $[a + bn + cn^2, b + 2cn, c]$ for an integer n. In order to compute a form $[A_i, B_i, C_i]$, we first transform a reduced primitive form $[\alpha, \beta, \gamma]$ to a form $[\alpha_1, \beta_1, \gamma_1]$ such that β_1 and γ_1 are divided by l, using the first transformation. This means that we choose an integer k such that $\beta_1 = \beta + 2\alpha k \equiv 0 \pmod{l}$ and $\gamma_1 = \gamma + \beta k + \alpha k^2 \equiv 0 \pmod{l}$. If $\alpha \equiv 0 \pmod{l}$, we just set $\alpha_1 = \gamma$ and $\gamma_1 = \alpha$, and we do not apply the transformation ($\beta_1 = \beta \equiv 0 \pmod{l}$, because $D \equiv 0 \pmod{l}$). After this transformation, we use the second transformation from [35] to compute the

Table 2. Modular functions for different values of l.

l	$\Phi_l(x, j)$
3	$(x+27)(x+3)^3 - jx$
5	$(x^2+10x+5)^3 - jx$
7	$(x^2+13x+49)(x^2+5x+1)^3 - jx$
13	$(x^2+5x+13)(x^4+7x^3+20x^2+19x+1)^3 - jx$

final form $[A_i, B_i, C_i]$ from $[\alpha_1, \beta_1, \gamma_1]$. Thus, $A_i = \alpha_1 + \beta_1 n + \gamma_1 n^2$, $B_i = \beta_1 + 2\gamma_1 n$, and $C_i = \gamma_1$ for an integer n such that $A_i > B_i > C_i$.

It is easy to see why this process yields a form that satisfies the desired conditions. The requirement $A_i > B_i > C_i$ exists because our experiments showed that it is necessary for the proper construction of the polynomial $M_{D,l}(x)$. For example, for $D = 51$, the reduced forms are $[1, 1, 13]$, $[3, 3, 5]$, and the corresponding forms $[A_i, B_i, C_i]$ for $l = 3$ are $[67, 63, 15]$, $[11, 9, 3]$.

The invariants $m_l^e(\tau)$ are related with $j(\tau)$ through the modular equation $\Phi_l(m_l^e(\tau), j(\tau)) = 0$, based on the definitions of $\Phi_l(x, j)$ for the different values of l given in Table 2.

Theorem 5. *A polynomial $M_{D,l}(x)$ has h roots modulo p if and only if the equation $4p = u^2 + Dv^2$ has an integer solution and p does not divide the discriminant $\Delta(M_{D,l})$ of the polynomial.*

Proof. It follows the same lines as that of Theorem 1. We know that the class invariants m_l^e generate the Hilbert class field, and therefore Proposition 5.29 from [10] hold. This implies that $M_{D,l}(x)$ has a root modulo p when $4p = u^2 + Dv^2$ has an integer solution, and since $\frac{\mathcal{O}_{H_K}}{p\mathcal{O}_{H_K}}/\mathbb{F}_p$ is Galois, the polynomial $M_{D,l}(x)$ has h distinct solutions modulo p. □

The polynomials $M_{D,l}(x)$ can be used in the CM method in a more straightforward way, compared to that of Weber polynomials for the case of prime order elliptic curves. Since $M_{D,l}(x)$ has roots R_M modulo p, we use an algorithm for their computation (for example, Berlekamp's algorithm [5]), and then we can compute the roots R_H modulo p of the corresponding Hilbert polynomial $H_D(x)$ from the modular equation $\Phi_l(R_M, R_H) = 0$.

We finally note that the precision required for the construction of the $M_{D,l}(x)$ polynomials is approximately $\frac{1}{l}$H-Prec(D) [11].

5. Implementation and Experimental Results

In this section, we discuss some issues regarding the implementation of our variant of the Complex Multiplication method and our experimental results concerning its time and space efficiency. All of our implementations were made in ANSI C using the (ANSI C) GNUMP [16] library for high-precision floating-point arithmetic and also

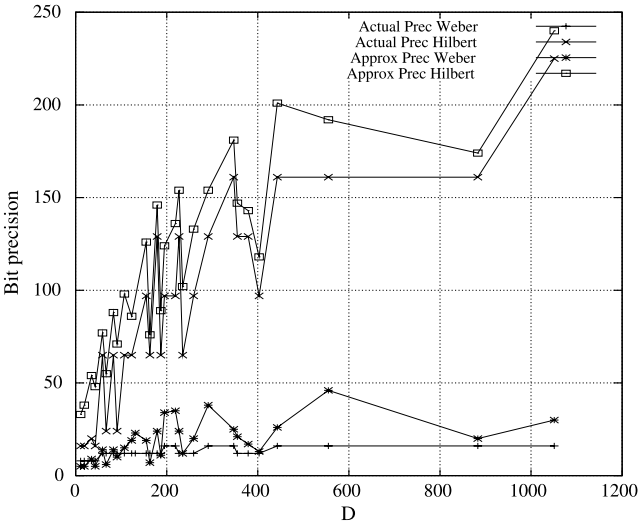

Fig. 1. Bit precision for the construction of Hilbert and Weber polynomials for various degrees $h \in [3, 15]$.

for the generation and manipulation of integers of unlimited precision. The implementation includes the construction of the Hilbert, Weber, and $M_{D,l}(x)$ polynomials, algorithms for the computation of roots modulo p of a polynomial, algorithms for the computation of a cubic factor of a polynomial modulo p, and of course all the steps of the CM method for the generation of prime-order elliptic curves. All implementations and experiments have been carried out on a Pentium III (933 MHz) running Linux and equipped with 256 MB of main memory. Our implementation is also part of a software library for EC cryptography that we build [20]. The library is available from http://www.ceid.upatras.gr/faculty/zaro/software/ecc-lib/.

5.1. *Construction of Class Field Polynomials*

Our experiments first focused on the bit precision and the time requirements needed for the construction of Hilbert and Weber class field polynomials. We have considered various values of D and h and made several experiments. We observed a big difference in favor of Weber polynomials both w.r.t. precision and time. This was evident even for small values of D and h. Figure 1 illustrates the approximate (theoretical) estimate of the bit precision required for the construction of Weber and Hilbert polynomials and the *actual precision*, i.e., the minimal precision required for their actual construction during the experiments. The actual precision was computed by gradually increasing the bit precision used in GNUMP library for floating-point arithmetic and stop at the first value for which each polynomial was correctly constructed.

As is evident from the figure, there is a large difference in the required precision between the two types of polynomials. The difference grows considerably larger for bigger values of D. The degree h for these values of D ranges from 3 to 15. This means that for different values of D in Fig. 1, the corresponding degrees h can be the same. For this reason, it is more convenient to show the bit precision requirements for the construction of the polynomials with respect to the discriminant D

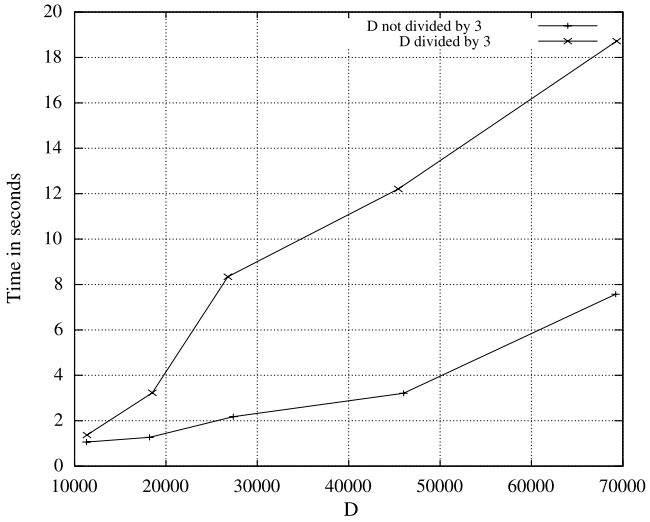

Fig. 2. Time in seconds for the construction of Weber polynomials for various degrees $h \in [50, 150]$.

(and not h). We also observe (see Fig. 1) that the approximate precision estimates are very close to the actual precision used in the implementation. For Hilbert polynomials, the approximation from (6) was used, while for Weber polynomials, that of Sect. 3. Regarding the time requirements of Weber polynomials, illustrative results are reported in Fig. 2. It is clear that the precision required for the case of $D \equiv 0 \pmod 3$ is bigger than the precision required for $D \not\equiv 0 \pmod 3$ for similar values of D and h. The degree h of the polynomials ranges from 50 to 150, while D ranges from 11299 to 69315 (for $D = 69211$ and $h = 150$, the time for the construction of the polynomial is only 7.57 seconds). The difference between these two cases can be readily explained from the EC theory: the class invariants for such values of D are raised to the power of three, and since they increase in magnitude, the time requirements are expected to be much larger than the requirements for Weber polynomials corresponding to other values of the discriminant. This fact implies that values of D divisible by 3 should be avoided.

Our experiments focused also on the bit precision and the time requirements needed for the construction of Weber and $M_{D,l}(x)$ polynomials with $D \equiv 3 \pmod 8$. We would like to note here that the construction of Hilbert polynomials is much less efficient than the construction of Weber (as seen in Fig. 1) or $M_{D,l}(x)$ polynomials for all values of D and l. Concerning Weber polynomials, we used discriminants $D \not\equiv 0 \pmod 3$. We avoid discriminants $D \equiv 0 \pmod 3$ because the precision requirements are greater than those of the case $D \not\equiv 0 \pmod 3$. We have considered various values of D and report on our experimental results in Figs. 3 and 4. The degree h for the first value of the discriminant D in the figures is equal to 32, while for the last two values, it is 48. We noticed, as the theory dictates, that the precision required for the construction of Weber polynomials $W_D(x)$ is less than the precision required for the construction of $M_{D,l}(x)$ polynomials for all the values of l that we examined (in Sect. 4 we explained why we consider these particular values of l). Among the $M_{D,l}(x)$ polynomials, the least precision is required

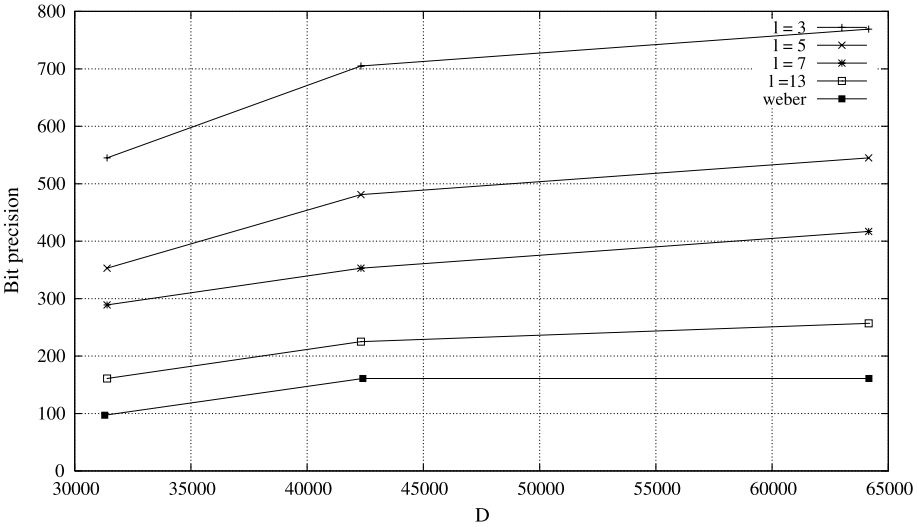

Fig. 3. Bit precision for the construction of class polynomials.

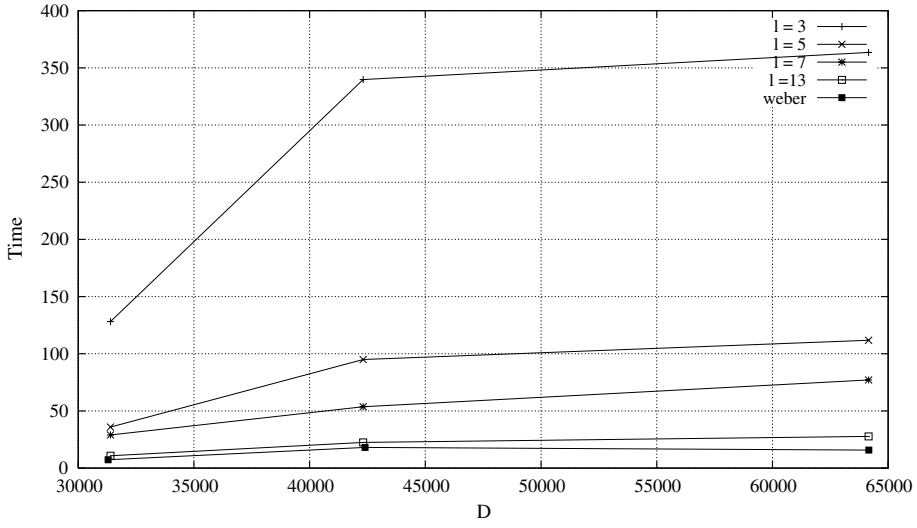

Fig. 4. Time requirements for the construction of class polynomials.

for the construction of $M_{D,13}(x)$, followed by the construction of $M_{D,7}(x)$, followed by the construction of $M_{D,5}(x)$. The greatest requirements in precision are set by the $M_{D,3}(x)$ polynomials.

The same ordering can be observed in the construction time. For Fig. 4 (time in seconds), we used the same values of D as in Fig. 3, and also in this figure the differences among the polynomials are very clear. We observed that the time for the construction of $M_{D,l}(x)$ depends not only on the precision requirements of the polynomials but also

on the convergence rate of η-products. The greater the l, the slower the convergence. This is why in Fig. 4 the differences do not seem to be analogous with the differences in Fig. 3. This favors Weber polynomials, as the η-products in their construction converge faster than any of the $M_{D,l}(x)$ polynomials, making their generation even more efficient.

The operations that are responsible for the construction time are two: firstly the computation of the class invariants and secondly the construction of the polynomial via the product $(x-a_1)(x-a_2)\cdots(x-a_h)$, where each a_i is a class invariant. As the degree of the polynomials increases, the computation of this product becomes much more time consuming and is responsible for the largest part of the construction time. For example, in the case of $M_{D,3}(x)$ polynomials in Fig. 4, the time requirements for the construction of the polynomials for the three values of the discriminant D were 128.04, 339.74, and 363.43 seconds, respectively. The total time required for the computation of all the class invariants for these three cases was only 19.23, 52.81, and 58.08 seconds, respectively. The remaining time of 108.81, 286.93, and 305.35 seconds, respectively, from the total construction time of the polynomials is due to the computation of the product $(x-a_1)(x-a_2)\cdots(x-a_h)$.

The coefficients of the Weber polynomials are also smaller than the coefficients of the $M_{D,l}(x)$ polynomials, following the same relative order with precision and time. However, the disadvantage of Weber polynomials is that their degree is three times larger than the degree of the $M_{D,l}(x)$ polynomials. Therefore, the space required for the storage of a Weber polynomial $W_D(x)$ can be larger than the space required for the storage of $M_{D,13}(x)$ or $M_{D,7}(x)$. Actually, it turns out that $M_{D,l}(x)$ polynomials can be even more advantageous when it comes to storage requirements as our experiments showed. Suppose that $M_{D,l}(x)=x^h+M_1x^{h-1}+\cdots+M_{h-1}x+M_h$ and h is even. We noticed that every coefficient M_i of $M_{D,l}(x)$ is divisible by l. Moreover, when $l=13$, then $M_h=13^{h/2}$ and $\frac{M_{h-i}}{M_i}=13^{h/2-i}$ for $1\le i\le(h/2-1)$. For $l=7$, $M_h=7^h$ and $\frac{M_{h-i}}{M_i}=7^{h-2i}$; for $l=5$, $M_h=5^{3h/2}$ and $\frac{M_{h-i}}{M_i}=5^{3h/2-3i}$; and finally, for $l=3$, we have $M_h=3^{3h}$ and $\frac{M_{h-i}}{M_i}=3^{3h-6i}$. Using these properties of the $M_{D,l}(x)$ polynomials, we can reduce the space required for their storage (if someone wants to store them for subsequent use).

This is not the only advantage of $M_{D,l}(x)$ against $W_D(x)$. The large degree of the Weber polynomials is a disadvantage for the time efficiency of the CM method, because the time for finding a cubic factor of the polynomial can be much larger than the time for finding a single root modulo p of a polynomial with three times smaller degree. In Table 3 we report on the time (in seconds) that is required for the computation of a cubic factor modulo p of $W_D(x)$, denoted by T_W, and the time that is required for the computation of a linear factor modulo p of the $M_{D,l}(x)$ polynomials denoted by T_M, for various values of l. The prime p has size 160 bits. C_W and C_M is the time required for the construction of the $W_D(x)$ and the $M_{D,l}(x)$ polynomials, respectively. The degree of $W_D(x)$ is $3h$. Note that C_W+T_W (resp. C_M+T_M) is the time that mostly dominates and differentiates the use of polynomials (Weber versus $M_{D,l}(x)$) in the CM method, since the time for the other steps of the method is practically independent of the polynomials used.

In particular, the last two steps of the CM method, which include the transformation of a root to a root of the corresponding Hilbert polynomial and the choice of the correct

Table 3. Time for the computation of a cubic factor of Weber polynomials and of a linear factor of the $M_{D,l}(x)$ polynomials, together with their construction time.

D	h	l	T_W	C_W	T_M	C_M
403	2	13	0.12	0.63	0.01	0.36
1027	4	13	0.40	1.31	0.02	0.38
2035	8	5	1.53	2.35	0.07	1.31
2795	12	13	3.88	3.60	0.13	2.12
4403	20	7	13.12	5.15	0.44	8.71
5603	22	13	16.97	6.94	0.50	8.38
6995	32	5	41.05	9.64	1.72	36.03
22435	32	5	41.05	17.80	1.72	72.94

EC, require on average 0.32 seconds in the 192-bit field. This time is not influenced by the size of the CM discriminant D. The transformation of a Weber root to the corresponding Hilbert root requires 0.75 msecs, while the transformation of a root of the $M_{D,l}(x)$ polynomials requires 0.70 msecs. It is clear that the choice of the correct EC is responsible for the total time of these two last steps. Recall that the correct EC is found by using Langrange's theorem, which involves one (at least) point multiplication operation. This operation is much more time consuming than a simple transformation.

We observe from Table 3 that $C_W + T_W$ is almost always larger than $C_M + T_M$, implying that the use of Weber polynomials is more time consuming than the use of the $M_{D,l}(x)$ polynomials. However, we also observed that in some cases where D increases, h is of moderate size, and $l \in \{3, 5\}$, the construction of the $M_{D,l}(x)$ polynomials may become less efficient (cf. last line of Table 3), and the total time of the CM method with these polynomials can be larger than the time required by the method when their corresponding Weber polynomials are used.

In conclusion, the type of polynomial that one should use depends on the particular application. If the main focus is on time or precision regarding the construction of the polynomials, then Weber polynomials should be preferred. If the focus is on fast and frequent generation of ECs and which implies storage of polynomials for subsequent use in the CM method, then the $M_{D,l}(x)$ polynomials ($l \neq 3$) must be preferred. Finally, if the class polynomials are computed online with the CM method, then the selection of the proper polynomial depends on D and h. Notice, though, that Weber polynomials can be constructed for any value of $D \equiv 3 \pmod 8$, while $M_{D,l}(x)$ polynomial add a restriction for D, demanding that $D \equiv 0 \pmod l$.

5.2. *Computation of p and m*

In this section we elaborate on the four methods for the calculation of the prime order p of the underlying field and the prime (and suitable) order m of the EC (recall Sect. 2.2). We shall refer to these methods as R (random choice used in [34]), C (modified Cornacchia's algorithm), B (Baier's algorithm in [3, p. 68]), and N (new method). We have made several experiments both in the 192-bit and in the 224-bit fields with various values of D and h. All reported experimental values are averages over 3000 ECs for each value of the discriminant D.

In the figures and the tables that follow, we report on the time requirements for each method and on the number of integers p and m that must be computed by each method

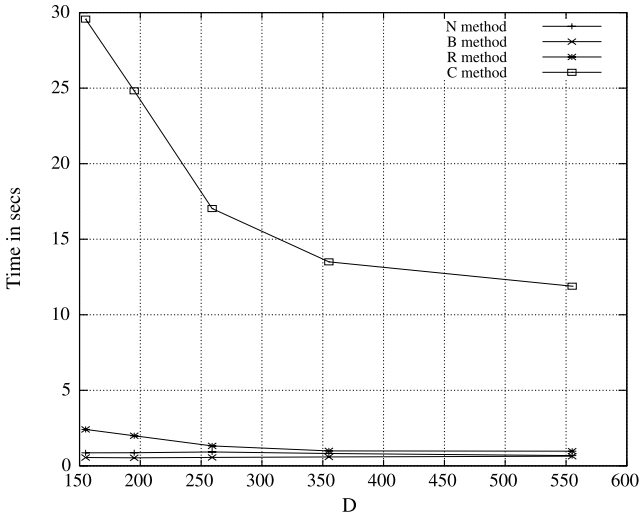

Fig. 5. Time in seconds for the computation of p and m in the 224-bit field.

until both of them are primes. For example, in the R method, random numbers u and v of proper size are generated, and then it is checked whether the number $p = (u^2 + Dv^2)/4$ is prime. If it is not, then other parameters u and v are created, and the same process is followed. After a number of iterations (we will denote it by $\#p$) a prime p is found, the two EC orders m^+ and m^- are constructed, and their primality is checked. The number of iterations required to find a prime order m will be denoted by $\#m$. This means that the total number of pairs (u, v) generated in order to find a prime p and a prime m are $\#p \cdot \#m$.

Figure 5 presents the time requirements of the four methods for various discriminants D in the 224-bit field. Clearly, C is by far the slowest, even for small values of h ($h \leq 10$ in Fig. 5); this is due to its time complexity, which is $O(\log^4 p)$. More timing details on the four methods are given in Table 4. The B method is the fastest, followed closely by the N method. The performance of these two methods is not influenced by the discriminant D. It is clear that the B method has an advantage against the other methods because the number of iterations required to find the primes p and m are much smaller compared to the corresponding iterations for the other methods. Concerning the other two methods (C and R), notice that their performance improves as the discriminant D increases and the degree h remains the same. The number of primes that we had to try in order to find a solution (u, v) with the modified Cornacchia's algorithm is equal to $3h$ with surprising accuracy. This fact can be seen also in Tables 5 and 6. Moreover, it is interesting that the values of $\#m$ are very similar for C and R methods. This can be explained easily as the order m in both methods is constructed by randomly chosen primes p. In contrast, the heuristics used in B and N methods reduce the numbers $\#p$ and $\#m$. In Tables 5 and 6 we report on the performance of the four methods in the 192-bit field. We notice that as D and h increase, the time requirements of C and R methods increase, while the performance of the other two methods remains the same.

Table 4. Timing estimations (in secs) of all methods in the 224-bit finite field.

D	h	C method			R method			B method			N method		
		$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time
155	4	12	121	29.57	117	106	2.41	17	33	0.55	39	43	0.86
195	4	12	78	24.82	104	87	1.99	17	33	0.53	40	43	0.87
259	4	12	54	17.03	91	59	1.32	18	33	0.56	43	45	0.92
355	4	12	46	13.50	81	43	0.99	18	34	0.59	39	42	0.82
555	4	12	42	11.89	61	42	0.97	19	37	0.64	31	38	0.68

Table 5. Timing estimations (in secs) of all methods in the 192-bit finite field.

D	h	C method			R method			B method			N method		
		$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time
59	3	9	138	19.70	123	134	1.84	16	28	0.32	24	26	0.31
83	3	9	99	15.97	104	103	1.49	15	27	0.31	31	32	0.41
107	3	9	94	12.53	92	85	1.26	15	27	0.31	36	38	0.50
379	3	9	26	4.39	49	30	0.48	16	30	0.34	25	28	0.36
883	3	9	15	2.26	32	17	0.26	16	28	0.32	25	26	0.31
179	5	15	123	34.03	118	124	1.81	15	28	0.30	31	34	0.42
227	5	15	97	24.42	105	106	1.54	15	26	0.28	31	32	0.41
347	5	15	74	18.45	85	83	1.32	16	28	0.31	34	39	0.51
443	5	15	65	16.08	76	70	1.19	16	28	0.31	30	35	0.45
1051	5	15	26	6.92	50	29	0.46	16	29	0.32	25	27	0.33

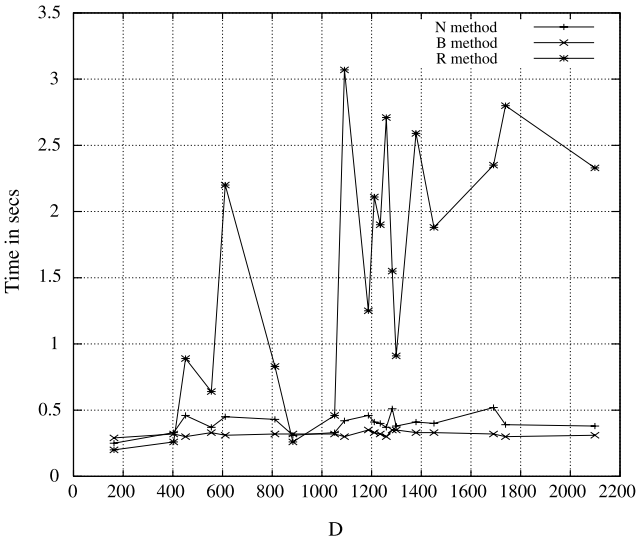

Fig. 6. Time requirements for the computation of p and m for various degrees $h \in [10, 20]$.

Since the C method is by far the less efficient one, we do not consider C when reporting results with larger values of D and h, and concentrate on the comparison among methods R, B, and N. The difference in efficiency among these three

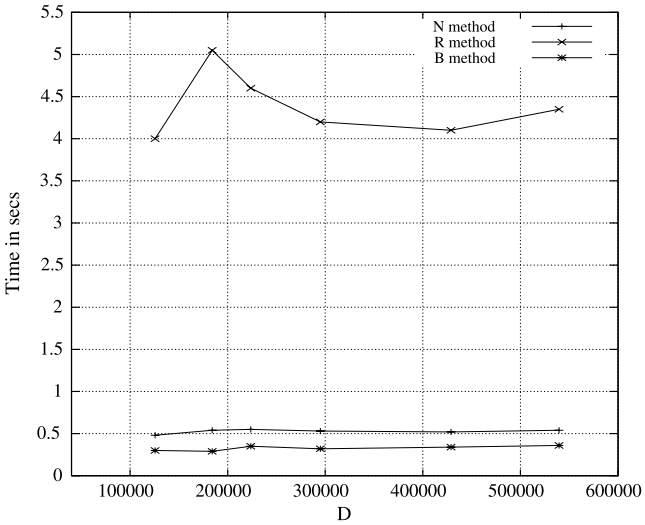

Fig. 7. Time requirements for the computation of p and m for various degrees $h \in [200, 400]$.

Table 6. Timing estimations (in secs) of all methods in the 192-bit finite field for various degrees h.

D	h	C method			R method			B method			N method		
		$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time
163	1	3	12	0.67	24	12	0.20	15	28	0.29	19	20	0.25
403	2	6	15	1.65	31	16	0.26	15	30	0.32	25	26	0.33
883	3	9	16	2.26	32	17	0.26	15	29	0.32	25	26	0.31
555	4	12	36	7.52	54	39	0.64	16	30	0.33	26	28	0.37
1051	5	15	23	6.79	50	29	0.46	15	29	0.32	25	27	0.33
451	6	18	53	16.70	90	57	0.89	15	28	0.30	35	37	0.46
811	7	21	43	15.63	78	50	0.83	15	29	0.32	30	34	0.43
1299	8	24	53	22.01	91	57	0.91	16	31	0.35	30	28	0.38
1187	9	27	78	30.62	84	76	1.25	16	31	0.35	34	37	0.46
611	10	30	128	55.04	131	134	2.20	15	29	0.31	34	36	0.45

methods can be seen in Figs. 6 and 7. Figure 6 involves values of D ranging from 163 to 2099 and values of h ranging from 10 to 20, while Fig. 7 involves values of (D, h) in $\{(125579, 200), (184091, 250), (223739, 300), (294971, 350), (428819, 400), (539579, 450)\}$. More details for these values are given in Table 7. We notice that the performance of the three methods is not affected by the degree h and the discriminant D. This is surprising for the R method, whose time requirements (as seen in Table 6 and Fig. 6) increase with h, when h is small. We conclude that there is an upper bound on the degree h after which the performance of the method stabilizes.

In either case, we observe a similar behavior in the relative efficiency among the three methods: R is the most time consuming, while the most efficient is B. The new method (N) is slightly slower than B, but it is simpler and uses less memory. The difference between R, and B or N becomes more apparent as D and h increase (cf. Fig. 7). We would also like to note that the timings obtained by our implementation of B using GNUMP

Table 7. Timing estimations (in secs) of the three methods in the 192-bit finite field.

D	h	R method			B method			N method		
		$\#p$	$\#m$	Time	$\#p$	$\#m$	Time	$\#p$	$\#m$	Time
125579	200	187	204	4.01	15	28	0.30	36	35	0.48
184091	250	217	252	5.13	14	28	0.29	38	40	0.54
223739	300	211	243	4.56	15	31	0.35	41	43	0.55
294971	350	205	212	4.18	15	29	0.32	41	41	0.53
428819	400	192	220	4.09	15	30	0.34	39	41	0.52
539579	450	206	222	4.41	16	31	0.36	39	40	0.54

are very close to those reported in [3], which were based on a C++ implementation of B using the advanced C++ library LiDIA [25] and carried out on a similar machine.

6. Conclusions

We have presented a variant of the Complex Multiplication method for the generation of prime-order ECs using Weber polynomials. We have shown that Weber polynomials in this case do not have roots in $\mathbb{F}_p$ but do have in the extension field $\mathbb{F}_{p^3}$. We have also presented a set of transformations for mapping roots of Weber polynomials in $\mathbb{F}_{p^3}$ to roots of their corresponding Hilbert polynomials in $\mathbb{F}_p$, and we have shown how a new class of polynomials can be used instead of Weber polynomials in the CM method. Finally, we have compared experimentally the use of Weber polynomials against the use of this new class, and we have investigated the efficiency of the computation of p and m under four different implementations showing the superiority of two of them. We believe that our experimental results can be used as a guideline for the construction of prime-order elliptic curves, as the potential designer can have an estimate of the computation time, and the precision required before the actual implementation is accomplished.

Acknowledgements

We are indebted to the anonymous referees for their criticism and valuable comments that improved the presentation.

References

[1] A.O.L. Atkin, F. Morain, Elliptic curves and primality proving. *Math. Comput.* **61**, 29–67 (1993)
[2] H. Baier, Elliptic curves of prime order over optimal extension fields for use in cryptography, in *Progress in Cryptology—INDOCRYPT 2001*. Lecture Notes in Computer Science, vol. 2247 (Springer, Berlin, 2001), pp. 99–107
[3] H. Baier, Efficient algorithms for generating elliptic curves over finite fields suitable for use in cryptography. PhD Thesis, Dept. of Computer Science, Technical Univ. of Darmstadt, May 2002
[4] H. Baier, J. Buchmann, Efficient construction of cryptographically strong elliptic curves, in *Progress in Cryptology—INDOCRYPT 2000*. Lecture Notes in Computer Science, vol. 1977 (Springer, Berlin, 2000), pp. 191–202
[5] E.R. Berlekamp, Factoring polynomials over large finite fields. *Math. Comput.* **24**, 713–735 (1970)
[6] I. Blake, G. Seroussi, N. Smart, *Elliptic Curves in Cryptography*. London Mathematical Society Lecture Note Series, vol. 265 (Cambridge University Press, Cambridge, 1999)

[7] D. Boneh, B. Lynn, H. Shacham, Short signatures from the Weil pairing, in *ASIACRYPT 2001*. Lecture Notes in Computer Science, vol. 2248 (Springer, Berlin, 2001), pp. 514–532

[8] H. Cohen, *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics, vol. 138 (Springer, Berlin, 1993)

[9] G. Cornacchia, Su di un metodo per la risoluzione in numeri interi dell' equazione $\sum_{h=0}^{n} C_h x^{n-h} y^h = P$. *G. Mat. Battaglini* **46**, 33–90 (1908)

[10] D.A. Cox, *Primes of the Form* $x^2 + ny^2$ (Wiley, New York, 1989)

[11] A. Enge, F. Morain, Comparing invariants for class fields of imaginary quadratic fields, in *Algebraic Number Theory—ANTS V*. Lecture Notes in Computer Science, vol. 2369 (Springer, Berlin, 2002), pp. 252–266

[12] A. Enge, R. Schertz, Constructing elliptic curves from modular curves of positive genus. Preprint (2003)

[13] A. Enge, R. Schertz, Modular curves of composite level. *Acta Arith.* **118**(2), 129–141 (2005)

[14] G. Frey, H.G. Rück, A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of curves. *Math. Comput.* **62**, 865–874 (1994)

[15] S. Galbraith, J. McKee, The probability that the number of points on an elliptic curve over a finite field is prime. *J. Lond. Math. Soc.* **62**(3), 671–684 (2000)

[16] GNU multiple precision library, edition 3.1.1, September 2000. Available at: http://www.swox.com/gmp

[17] IEEE P1363/D13, Standard Specifications for Public-Key Cryptography, 1999. http://grouper.ieee.org/groups/1363/tradPK/draft.html

[18] E. Kaltofen, N. Yui, Explicit construction of the Hilbert class fields of imaginary quadratic fields by integer lattice reduction. Research Report 89-13, Rensselaer Polytechnic Institute, May 1989

[19] E. Kaltofen, T. Valente, N. Yui, An improved Las Vegas primality test, in *Proc. ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation* (1989), pp. 26–33

[20] E. Konstantinou, Y. Stamatiou, C. Zaroliagis, A software library for elliptic curve cryptography, in *Proc. 10th European Symposium on Algorithms—ESA 2002 (Engineering and Applications Track)*. Lecture Notes in Computer Science, vol. 2461 (Springer, Berlin, 2002), pp. 625–637

[21] E. Konstantinou, Y. Stamatiou, C. Zaroliagis, On the efficient generation of elliptic curves over prime fields, in *Cryptographic Hardware and Embedded Systems—CHES 2002*. Lecture Notes in Computer Science, vol. 2523 (Springer, Berlin, 2002), pp. 333–348

[22] E. Konstantinou, Y.C. Stamatiou, C. Zaroliagis, On the construction of prime order elliptic curves, in *Progress in Cryptology—INDOCRYPT 2003*. Lecture Notes in Computer Science, vol. 2904 (Springer, Berlin, 2003), pp. 309–322

[23] E. Konstantinou, A. Kontogeorgis, Y. Stamatiou, C. Zaroliagis, Generating prime order elliptic curves: difficulties and efficiency considerations, in *International Conference on Information Security and Cryptology—ICISC 2004*. Lecture Notes in Computer Science, vol. 3506 (Springer, Berlin, 2005), pp. 261–278

[24] G.J. Lay, H. Zimmer, Constructing elliptic curves with given group order over large finite fields, in *Algorithmic Number Theory—ANTS-I*. Lecture Notes in Computer Science, vol. 877 (Springer, Berlin, 1994), pp. 250–263

[25] LiDIA. A library for computational number theory. Technical University of Darmstadt. Available from http://www.informatik.tu-darmstadt.de/TI/LiDIA/Welcome.html

[26] A.J. Menezes, T. Okamoto, S.A. Vanstone, Reducing elliptic curve logarithms to a finite field. *IEEE Trans. Inf. Theory* **39**, 1639–1646 (1993)

[27] A. Miyaji, M. Nakabayashi, S. Takano, Characterization of elliptic curve traces under FR-reduction, in *International Conference on Information Security and Cryptology—ICISC 2000*. Lecture Notes in Computer Science, vol. 2015 (Springer, Berlin, 2001), pp. 90–108

[28] A. Miyaji, M. Nakabayashi, S. Takano, New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Trans. Fundam.* **E84-A**(5), 1234–1243 (2001)

[29] F. Morain, Modular curves and class invariants. Preprint, June 2000

[30] F. Morain, Computing the cardinality of CM elliptic curves using torsion points. Preprint, October 2002

[31] Y. Nogami, Y. Morikawa, Fast generation of elliptic curves with prime order over $F_{p^{2^c}}$, in *Proc. of the International workshop on Coding and Cryptography*, March 2003

[32] G.C. Pohlig, M.E. Hellman, An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Trans. Inf. Theory* **24**, 106–110 (1978)

[33] T. Satoh, K. Araki, Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Comment. Math. Univ. St. Pauli* **47**, 81–91 (1998)
[34] E. Savaş, T.A. Schmidt, Ç.K. Koç, Generating elliptic curves of prime order, in *Cryptographic Hardware and Embedded Systems—CHES 2001*. Lecture Notes in Computer Science, vol. 2162 (Springer, Berlin, 2001), pp. 145–161
[35] R. Schertz, Weber's class invariants revisited. *J. Théor. Nr. Bordx.* **4**, 325–343 (2002)
[36] R. Schoof, Counting points on elliptic curves over finite fields. *J. Théor. Nr. Bordx.* **7**, 219–254 (1995)
[37] M. Scott, P.S.L.M. Barreto, Generating more MNT elliptic curves, Cryptology ePrint Archive, Report 2004/058 (2004)
[38] J.H. Silverman, *The Arithmetic of Elliptic Curves* (Springer, Berlin, 1986). GTM 106
[39] I. Stewart, *Galois Theory*, 3rd edn. (Chapman & Hall/CRC, Boca Raton, 2004)
[40] I. Stewart, D. Tall, *Algebraic Number Theory*, 2nd edn. (Chapman & Hall, London, 1987)
[41] T. Valente, A distributed approach to proving large numbers prime. Rensselaer Polytechnic Institute Troy, New York, PhD Thesis, August 1992