On the Construction of Prime Order Elliptic Curves[⋆]

Elisavet Konstantinou[1,2], Yannis C. Stamatiou[3,4], and Christos Zaroliagis[1,2]

[1] Computer Technology Institute, P.O. Box 1122, 26110 Patras, Greece
[2] Dept of Computer Eng. & Informatics, University of Patras, 26500 Patras, Greece
{konstane,zaro}@ceid.upatras.gr
[3] Dept of Mathematics, University of the Aegean, 83200 Samos, Greece
[4] Joint Research Group (JRG) on Communications and Information Systems Security (University of the Aegean and Athens University of Economics and Business)
stamatiu@aegean.gr

Abstract. We consider a variant of the Complex Multiplication (CM) method for constructing elliptic curves (ECs) of prime order with additional security properties. Our variant uses Weber polynomials whose discriminant D is congruent to 3 (mod 8), and is based on a new transformation for converting roots of Weber polynomials to their Hilbert counterparts. We also present a new theoretical estimate of the bit precision required for the construction of the Weber polynomials for these values of D. We conduct a comparative experimental study investigating the time and bit precision of using Weber polynomials against the (typical) use of Hilbert polynomials. We further investigate the time efficiency of the new CM variant under four different implementations of a crucial step of the variant and demonstrate the superiority of two of them.

1 Introduction

Elliptic Curve (EC) cryptography has proven to be an attractive alternative for building fast and secure public key cryptosystems. 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 [5], small prime factors [1], etc), which gives rise to the problem of how such ECs can be generated.

A specific application domain that is our main concern in this work involves implementations of EC-based cryptosystems in computing devices with limited resources, or in systems operating under strict timing response constraints. Two specific scenarios in this framework involve: (i) The development of a proactive

[⋆] This work was partially supported by the IST Programme of EU under contracts no. IST-1999-14186 (ALCOM-FT) and no. IST-1999-12554 (ASPIS), and by the Human Potential Programme of EU under contract no. HPRN-CT-1999-00104 (AMORE).

cryptosystem approach (e.g., in the sense of [12]) in networks of resource limited hardware devices (e.g., microcontroller chips, smart dust clouds) working for some highly critical – with respect to security – task and which for that reason are frequently requested to refresh their security parameters. (ii) A wireless and web-based environment, as it is described in [11], in which millions of (resource-limited) client devices connect to secure servers. Clients may be frequently requested to choose different key sizes and EC parameters depending on vendor preferences, security requirements, and processor capabilities.

A frequently employed approach for generating ECs whose order satisfies certain desirable properties is the so-called *Complex Multiplication* (CM) method. This method was used by Atkin and Morain [1] for the construction of elliptic curves with good properties in the context of primality proving, and since then has been adapted to give rise to ECs with good security properties by Spallek [24], and independently by Lay and Zimmer [18]. Furthermore, a number of works appeared that compare variants of the CM method and also present experimental results concerning the construction efficiency, such as the recent works of Müller and Paulus [20], as well as the theses of Weng [26] and Baier [4].

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 the 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 large as D increases, while for the same D the Weber polynomials have much smaller coefficients (although their coefficients also grow with D) and thus are easier and faster to construct; (ii) the roots of the Hilbert polynomial construct directly the EC, while the roots of the Weber polynomial have to be transformed to the roots of its corresponding Hilbert polynomial to construct the EC. For a general discussion and comparison on class field polynomials, see [9].

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 resource requirements. 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., [22]), or to use Weber polynomials for certain values of D (see e.g., [2,4,15,17,18,25]) and produce the required Hilbert roots from them. The former approach [22] 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 $D \equiv 3 \pmod 8$ (a prime order is necessary in certain situations – see e.g., [6]). 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 [2,4,15,18], while the cases of $D \not\equiv 3 \pmod 8$ and $D \not\equiv 0 \pmod 3$ were treated in [17,25]. 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 [3], or they generate the EC parameters at random and use a point counting algorithm to compute the order of the curve [21]. To the best of our knowledge, the use of Weber polynomials within the CM method for the generation 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.

The first contribution of this paper is a new transformation for converting roots of Hilbert polynomials to roots of their corresponding Weber polynomials (Section 5) for the case $D \equiv 3 \pmod 8$, resulting in a new CM variant for generating ECs of prime order (Section 3) and which also satisfies the three conditions for cryptographic strength posed in [5, Sec. V.7]. We also investigate the theoretical (Section 4) and experimental (Section 6) bit-precision for the construction of Hilbert and Weber polynomials and present a new approximation bound of the precision required for the construction of Weber polynomials in the case $D \equiv 3 \pmod 8$. Our experiments showed that the new approximate bound is very close to the actual precision needed.

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 (a computation that can be performed off-line as we remarked above for various values of the discriminant D). We consider four different ways for implementing this step in the new CM variant (Section 3). The first method is similar to that in [17] and uses the modified Cornacchia's algorithm [8]. The second method generates p and m at random as it is described in [22]. The third method is the very efficient algorithm given in Baier's PhD thesis [4, p. 68]. The fourth method, which we introduce here, resembles the third one and constitutes a simpler and more space-efficient alternative to it.

The second contribution of this paper is a comparative experimental study (Section 6) regarding the four methods mentioned above for the computation of p and m in the construction of an EC, as well as an investigation of the time and bit precision requirements when constructing Weber polynomials on-line, in comparison with their Hilbert counterparts. Such an investigation is considerably important in resource-limited hardware systems which are requested to frequently change their security parameters. Our experiments revealed that despite the fact that Weber polynomials have a degree which is three times larger, the new CM variant using any of the four methods for computing p and m is considerably more space and time efficient than its counterpart which uses Hilbert polynomials. Regarding now the comparison of 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.

2 A Brief Overview of Elliptic Curve Theory

In this section we review some basic concepts and results of elliptic curve theory. It is assumed that the reader has some familiarity with elementary number theory. A detailed presentation of EC theory and its cryptographic significance can be found in [5,23].

An *elliptic curve* over a finite field F_p, p a prime larger than 3, is denoted by $E(F_p)$ and it is comprised of all the points $(x, y) \in F_p$ (in affine coordinates) such that

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

with $a, b \in F_p$ satisfying $4a^3 + 27b^2 \neq 0$. These points, together with a special point denoted by $\mathcal{O}$ (the *point at infinity*) and a properly defined addition operation form an Abelian group. This is the *Elliptic Curve group* and the point $\mathcal{O}$ is its identity element (see [5,23] for more details on this group). Finally, let m be the *order* of $E(F_p)$, that is, the number of points in the group.

The difference between m and p is measured by the so-called *Frobenius trace* $t = p + 1 - m$ for which Hasse's theorem (see e.g., [5,23]) states that $|t| \leq 2\sqrt{p}$, implying that

$$p + 1 - 2\sqrt{p} \leq m \leq p + 1 + 2\sqrt{p}. \tag{2}$$

This is an important inequality that provides lower and upper bounds on the number of points in an EC group.

If $P \in E(F_p)$, then the *order* of the point P is the smallest positive integer n for which $nP = \mathcal{O}$. According to Langrange's theorem the order of a point $P \in E(F_p)$ must divide the order of the EC group and, thus, $mP = \mathcal{O}$ for any $P \in E(F_p)$. This also implies that the order of a point is never larger than the order of the EC.

Among the most important quantities defined for an EC $E(F_p)$ given by Eq. (1) are the *curve discriminant* Δ and the *j-invariant*. These two quantities are given by the equations $\Delta = -16(4a^3 + 27b^2)$ and $j = -1728(4a)^3/\Delta$.

For a specific j-invariant $j_0 \in F_p$ (where $j_0 \neq 0, 1728$) two ECs can be readily constructed. Let $k = j_0/(1728 - j_0) \bmod p$. One EC is given by Eq. (1) with $a = 3k \bmod p$ and $b = 2k \bmod p$. The other, which is called the *twist* of the first, is given by

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

where c is any quadratic non-residue in F_p. If m_1 and m_2 are the orders of an EC and its twist respectively, then $m_1 + m_2 = 2p + 2$. This implies that if one of the curves has order $p + 1 - t$, then its twist has order $p + 1 + t$, or vice versa (see [5, Lemma VIII.3]).

EC cryptosystems base their security on the difficulty of solving efficiently the discrete logarithm problem (DLP) on the EC group. To increase the difficulty of the solution of DLP (and hence the security of the EC cryptosystem), the order m of the EC should obey certain conditions which guarantee resistance to all known attacks [5, Sec. V.7]. An order m that satisfies these conditions is called *suitable*. We would like to note that sometimes there exists a fourth security requirement regarding the degree h of the class field polynomial. To the best of our knowledge, such requirement is only posed by the German Information Security Agency, which requires that h should be greater than 200. The reason is that there are few ECs produced from class field polynomials with smaller degrees and which may be amenable to specific attacks. However, no such attacks are known to date and this requirement is not part of the security requirements in any international security standard [2]. Despite this fact, we have taken into consideration such large values of h in our experimental study.

3 CM Method and Variants

The main idea behind the CM method is as follows (see [5,13] for a detailed discussion). According to Hasse's theorem the quantity $Z = 4p - (p + 1 - m)^2$ is positive. Thus, there is a unique factorization of Z of the form Dv^2, with D a square free positive integer. Consequently,

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

for some integer u satisfying

$$m = p + 1 \pm u. \tag{5}$$

The number D is called a *CM discriminant* for the prime p and the EC has a *CM by D*. The CM method uses D to determine the j-invariant, and then constructs 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 Eq. (4). 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 Section 4) 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 Section 2. 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.

The most time consuming part of the CM method is the construction of the Hilbert polynomial, as it requires high precision floating point and complex arithmetic. In order to overcome the high computational requirements of this construction, a variant of the CM method was proposed in [22]. 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 justification for D). It then computes p and the EC order m (the primality of m is the only requirement for cryptographic strength set in [22]). 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 [17], 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 [8] to solve Eq. (4), and requires that the resulting EC order m is suitable (cf. Section 2) but not necessarily prime. Moreover, like in [22], 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 [22,17], but also differentiates 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 (Section 5).

We are now ready to present the main steps of our variant. It starts with a CM discriminant $D \equiv 3 \pmod 8$ for the computation of the Weber polynomial[5], 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.

We now turn to the four different methods for computing u and v. The first is to use the modified Cornacchia's algorithm [7] as in [17]. The second is to generate them at random as it is done in [22]. The third method was proposed in [4, p. 68] and uses some clever heuristic in order to speed up the discovery of a suitable prime p. 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 Eqs. (4) and (5) 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 is prime). We will denote the former by m^- and the latter with

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

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 ($3 \cdot 5 \cdot 7 = 105$). 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, we 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 $\pmod 3$ and at the next step congruent to 2 $\pmod 3$.

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.

4 Hilbert and Weber Polynomials

The only input for the construction of the Hilbert or the Weber polynomials, denoted by $H_D(x)$ and $W_D(x)$ respectively, is the CM discriminant D. They both require complex floating point arithmetic. The Hilbert polynomial $H_D(x)$, for a given positive value of D, is defined as

$$H_D(x) = \prod_{\tau} (x - j(\tau)) \tag{6}$$

for a set of values of τ obtained by the expression $\tau = (-\beta + \sqrt{-D})/2\alpha$, for all integers α, β, and γ that satisfy 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 $\beta \geq 0$. The 3-tuple of integers $[\alpha, \beta, \gamma]$ satisfying these conditions, is a *primitive, reduced quadratic form* of $-D$, and τ is a root of the quadratic equation $\alpha z^2 + \beta z + \gamma = 0$. It can be proved that the set of primitive reduced quadratic forms of discriminant $-D$, denoted by $\mathcal{H}(-\mathcal{D})$, is finite. Moreover, it is possible to define an operation that gives to $\mathcal{H}(-\mathcal{D})$ the structure of an Abelian group whose neutral element is called the *principal form*. The principal form is equal to $[1, 0, D/4]$ if $D \equiv 0 \pmod 4$ and $[1, -1, (D + 1)/4]$ if $D \equiv 3 \pmod 4$. This means that $\tau = \sqrt{-D}/2$ for the first principal form and $\tau = (1+\sqrt{-D})/2$ for the second. The quantity $j(\tau)$ in Eq. (6) is called *class invariant* and is defined as follows. Let $z = e^{2\pi\sqrt{-1}\tau}$ and $h(\tau) = \frac{\Delta(2\tau)}{\Delta(\tau)}$, where $\Delta(\tau) = z\left(1 + \sum_{n\geq 1} (-1)^n \left(z^{n(3n-1)/2} + z^{n(3n+1)/2}\right)\right)^{24}$. Then, $j(\tau) = \frac{(256h(\tau)+1)^3}{h(\tau)}$.

If h is the *degree* or *class number* of $H_D(x)$, the bit precision required for the generation of $H_D(x)$ according to [18] is

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

where the sum runs over the same values of τ as the product in Eq. (6). Note that this is much smaller than the precision given in [1,5].

The Weber polynomials are defined using the Weber functions (see [1,13]):

$$f(y) = q^{-1/48}\prod_{m=1}^{\infty}(1+q^{(m-1)/2}) \qquad f_1(y) = q^{-1/48}\prod_{m=1}^{\infty}(1-q^{(m-1)/2})$$
$$f_2(y) = \sqrt{2}\;\; q^{1/24}\prod_{m=1}^{\infty}(1+q^{m}) \qquad \text{where } q = e^{2\pi y\sqrt{-1}}.$$

Then, the Weber polynomial $W_D(x)$, which has degree $3h$ as $D \equiv 3 \pmod 8$, is defined as

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

where $\ell = \frac{-b+\sqrt{-D}}{a}$ satisfies the equation $ay^2+2by+c=0$ for which $4b^2-4ac = -4d$, where $d = D/4$ if $D \equiv 0 \pmod 4$, and $d = D$ if $D \equiv 3 \pmod 4$. Let $\zeta = e^{\pi\sqrt{-1}/24}$. 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 $D \not\equiv 0 \pmod 3$, 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 $D \equiv 0 \pmod 3$. In [25], an upper bound of $v_0 + \frac{\pi\sqrt{D}}{\ln 2}\sum_{\ell}\frac{1}{a}$ is given for the precision required for the construction of $W_D(x)$, where the sum runs over the same values of ℓ as the product in Eq. (7) and v_0 is a positive constant that handles round-off errors (typically $v_0 = 33$). In [18] a more accurate precision estimate is given for the computation of Weber polynomials with discriminants $D \equiv 7 \pmod 8$. In particular, the bit precision in this case is given by

$$\text{W-Prec}(D) \approx \frac{\ln 10}{\ln 2}\left(\frac{h/4+5+\frac{\pi\sqrt{D}}{\ln 10}\sum_{\tau}\frac{1}{\alpha}}{47}+1\right)$$

where τ takes the same values as in the product in Eq. (6) for $H_D(x)$. This precision estimate however, can not be used in the case $D \equiv 3 \pmod 8$ which is of our concern. For this reason, we provide in the following lemma a new precision estimate specifically for this case.

Lemma 1. *The bit precision required for the construction of Weber polynomials with discriminant* $D \equiv 3 \pmod 8$ *and* $D \not\equiv 0 \pmod 3$ *is approximately* $3h + \frac{\pi\sqrt{D}}{24 \ln 2} \sum_\ell \frac{1}{a}$*, where the sum runs over the same values of* ℓ *as the product* $W_D(x) = \prod_\ell (x - g(\ell))$*. For the case of* $D \equiv 3 \pmod 8$ *and* $D \equiv 0 \pmod 3$ *the approximate precision becomes* $3h + \frac{\pi\sqrt{D}}{8 \ln 2} \sum_\ell \frac{1}{a}$*.*

Proof. From the proof of Proposition (B4.4) in [16], if the Weber polynomial is written in the form $W_D(x) = x^{3h} + w_{3h-1}x^{3h-1} + \ldots + w_1 x + w_0$, then $|w_i| \leq 2^{3h} M$, where $M = \prod_\ell \max(1, |g(\ell)|)$. This means that the bit precision required for the coefficient w_i of the polynomial is $\log_2(|w_i|) \leq 3h + \log_2 M \leq 3h + \sum_\ell \log_2(|g(\ell)|)$. Therefore, the bit precision required for the construction of the whole polynomial (i.e., the construction of its coefficients) is at most $3h + \sum_\ell \log_2(|g(\ell)|)$.

For the case $D \equiv 3 \pmod 8$ and $D \not\equiv 0 \pmod 3$, the precision required by each $g(\ell)$ is the same with the precision required by $f(\ell)$, $f_1(\ell)$ or $f_2(\ell)$ as it is evident from Eq. (8). In addition, it is known that $j(z) = \frac{(f^{24}(z)-16)^3}{f^{24}(z)} = \frac{(f_1^{24}(z)+16)^3}{f_1^{24}(z)} = \frac{(f_2^{24}(z)+16)^3}{f_2^{24}(z)}$. These equalities imply that the precision needed for $j(\ell)$ is approximately 48 times the precision needed for $f(\ell)$, $f_1(\ell)$ or $f_2(\ell)$. Using the expansion of j in terms of its Fourier series [5], we obtain that $|j(\ell)| \approx |e^{-2\pi\sqrt{-1}\ell}| = e^{2\pi\sqrt{D}/a}$. Therefore, the bit precision that is required for the computation of $j(\ell)$ is $\log_2 |j(\ell)| \approx \frac{2\pi\sqrt{D}}{a \ln 2}$ and, consequently, the precision required for $g(\ell)$ is given by $\log_2 |g(\ell)| \approx \frac{2\pi\sqrt{D}}{48 a \ln 2} = \frac{\pi\sqrt{D}}{24 a \ln 2}$. This, in turn, results in the total bit precision requirements for the computation of the Weber polynomial: $3h + \frac{\pi\sqrt{D}}{24 \ln 2} \sum_\ell \frac{1}{a}$.

In the case $D \equiv 3 \pmod 8$ and $D \equiv 0 \pmod 3$, the precision required by $g(\ell)$ is three times the precision required by $f(\ell)$, $f_1(\ell)$ or $f_2(\ell)$ as it is evident from Eq. (9). Using an analysis similar to the analysis used in the previous case, we obtain that the bit precision requirements in this case is given by $3h + \frac{\pi\sqrt{D}}{8 \ln 2} \sum_\ell \frac{1}{a}$ which completes the proof of the lemma. □

5 Transforming Weber Roots to Hilbert Roots

In this section we elaborate on the transformation of roots of Weber polynomials to roots of the corresponding (generated from the same discriminant value D) Hilbert polynomials. Note that for the particular case we consider ($D \equiv 3 \pmod 8$), the degree of the Weber polynomial is *three* times larger that the degree of its Hilbert counterpart, and this introduces an additional difficulty. We start with some basic relationships between the Weber functions and $j(z)$ (defined in the previous section). In particular,

$$f(z) f_2\left(\frac{1+z}{2}\right) = e^{\pi\sqrt{-1}/24}\sqrt{2} \tag{10}$$

$$j(z) = \frac{(f^{24}(z) - 16)^3}{f^{24}(z)} = \frac{(f_1^{24}(z) + 16)^3}{f_1^{24}(z)} = \frac{(f_2^{24}(z) + 16)^3}{f_2^{24}(z)}. \tag{11}$$

Hence, $f^{24}(z)$, $-f_1^{24}(z)$, and $-f_2^{24}(z)$ are the roots of the cubic equation $(x - 16)^3 - xj(z) = 0$.

It can be proved that any transformation of a real root of a weber polynomial to a real root of the corresponding Hilbert polynomial holds also for the roots of the polynomials when taken $\pmod p$. Suppose R_W is a real root of $W_D(x)$ to be transformed to the corresponding real root R_H of $H_D(x)$. In addition, $R_H = j(\tau)$, where τ corresponds to the principal form. First, R_W is transformed into one of the quantities $f^{24}(\tau)$, $-f_1^{24}(\tau)$ or $-f_2^{24}(\tau)$ (we will denote either of these quantities by A) and we set $R_H = (A - 16)^3/A$. The most complex part of the transformations is the first, which depends on the discriminant D. For different values of D, different class invariants are used, which in turn, define the relationship between R_W and the Weber functions.

The class invariant for $D \equiv 3 \pmod 8$ and $D \not\equiv 0 \pmod 3$ is $f(\sqrt{-D})$. That is, $R_W = f(\sqrt{-D})$. The principal form for such discriminants is $[1, 1, (D + 1)/4]$, and hence one of the roots of Eq. (11) is $f^{24}(\tau)$, $-f_1^{24}(\tau)$, or $-f_2^{24}(\tau)$, where $\tau = (1 + \sqrt{-D})/2$. According to Eq. (10)

$$f_2(\tau) = f_2\left(\frac{1 + \sqrt{-D}}{2}\right) = e^{\frac{\pi\sqrt{-1}}{24}}\sqrt{2}f^{-1}(\sqrt{-D}) = e^{\frac{\pi\sqrt{-1}}{24}}\sqrt{2}R_W^{-1}.$$

Consequently, $f_2^{24}(\tau) = -2^{12}R_W^{-24}$ and since $A = -f_2^{24}(\tau)$, we obtain

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

The class invariant for $D \equiv 3 \pmod 8$ and $D \equiv 0 \pmod 3$ is $f^3(\sqrt{-D})/2$. That is, $R_W = f^3(\sqrt{-D})/2$. The principal form is, again, $[1, 1, (D + 1)/4]$ and one of the roots of Eq. (11) is $f^{24}(\tau)$, $-f_1^{24}(\tau)$, or $-f_2^{24}(\tau)$, where $\tau = (1+\sqrt{-D})/2$. Following the same procedure as before we obtain that $f_2^{24}(\tau) = -2^{12}f^{-24}(\sqrt{-D}) = -2^4R_W^{-8}$. Since $A = -f_2^{24}(\tau)$, then

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

6 Implementation and Experimental Results

As mentioned in the introduction, one of our main concerns was to investigate the efficiency of implementing CM variants in resource-limited hardware devices (e.g., embedded systems). For that reason and for reasons of proper comparison, we have made all of our implementations in a unified framework using the same language and software libraries. Since the vast majority of language tools developed for such devices are based on ANSI C, we have made all of our implementations in this language using the (ANSI C) GNUMP [10] library for high precision floating point arithmetic and also for the generation and manipulation of integers of unlimited precision. Our goal was to boost portability as well as adaptability to the development tools for resource-limited hardware devices. Note that there are highly efficient and optimized `C++` libraries (e.g., LiDIA [14]) which however result in executables of a few MB, since they call dynamically linked libraries at run time. In contrast, our code does not call any such libraries at runtime. In particular, we have carried out our implementations and experiments on a Pentium III (933 MHz) running Linux and equipped with 256 MB of main memory. The Weber (resp. Hilbert) version of our code had size 53KB (resp. 49KB) including the code for the generation of the polynomials; exclusion of the latter (i.e., when polynomials are computed off-line) reduces the code size to 29KB if the modified Cornacchia's algorithm is used, to 25KB if p and m are selected at random as it is done in [22], to 28KB if Baier's algorithm [4, p. 68] is used, and to 26KB if the new method (described in Section 3) is used. All reported experimental values are averages over 3000 ECs for each value of the discriminant D. We considered two prime field sizes, 192 and 224 bits, which are typically used in such experiments.

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(left) illustrates the actual and the approximate estimate of the bit precision for both Weber and Hilbert polynomials.

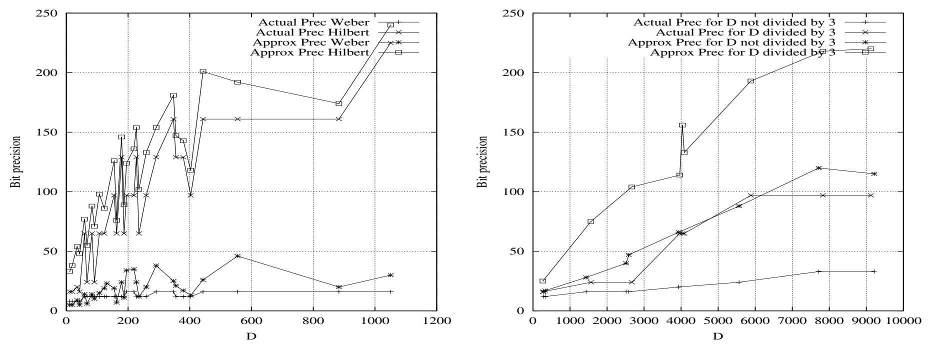

Fig. 1. Bit precision for the construction of Hilbert and Weber polynomials (left), and for the construction of Weber polynomials only (right).

As it 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 and h. We also observe (see Fig. 1(left)) that the approximate precision estimates are very close to the actual precision used in the implementation. For Hilbert polynomials the approximation from

Eq. (4) was used, while for Weber polynomials that of Lemma 1. Regarding the precision requirements of Weber polynomials and their theoretical estimates, illustrative results are reported in Figure 1(right). 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 approximate precision is larger than the actual precision for all values of D. The difference in the precision requirements of Weber polynomials for the two cases of D (divisible or not divisible by 3) is also reflected in the time requirements for their construction, shown in Figure 2(left). 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.

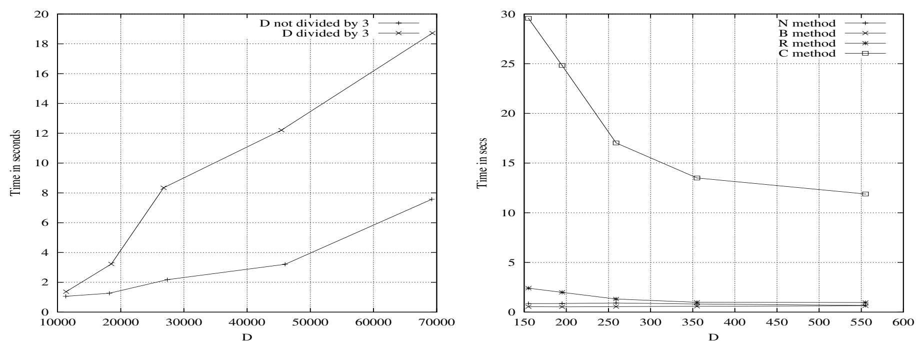

Fig. 2. Time in seconds for the construction of Weber polynomials (left) and for the computation of p, and m in the 224-bit field (right).

We next turn to 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. We shall refer to these methods as R (random choice used in [22]), C (modified Cornacchia's algorithm), B (Baier's algorithm in [4, 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. We report on the most representative results in Figures 2(right), and 3. Figure 2(right) 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. 2(right)); this is due to its time complexity which is $O(\log^4 p)$. Hence, 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 methods can be seen in Figure 3. Figure 3(left) involves values of D ranging from 163 to 2099, and values of h ranging from 10 to 20, while Figure 3(right) involves values of (D, h) in $\{(125579, 200),(184091, 250),(223739, 300),(294971, 350),(428819, 400),(539579, 450)\}$.

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. 3(right)). We would also like to note that the timings obtained by our implementation of B using GNUMP are very close to those reported in [4], which were based on a `C++` implementation of B using the advanced `C++` library LiDIA [14] and carried out on a similar machine.

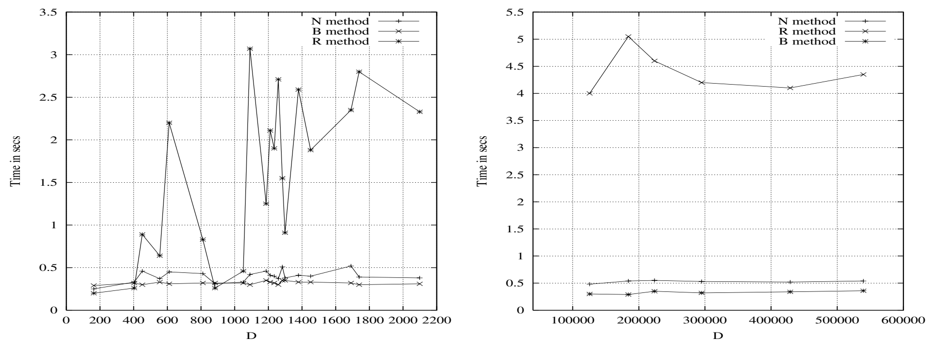

Fig. 3. Time requirements for the computation of p and m for various degrees $h \in [10, 20]$ (left) and $h \in [200, 400]$ (right).

References

1. A.O.L. Atkin and F. Morain, Elliptic curves and primality proving, *Mathematics of Computation* 61(1993), pp. 29-67.
2. H. Baier and J. Buchmann, Efficient construction of cryptographically strong elliptic curves, in *Progress in Cryptology* – INDOCRYPT 2000, LNCS Vol. 1977 (Springer-Verlag, 2000), pp. 191-202.
3. H. Baier, Elliptic Curves of Prime Order over Optimal Extension Fields for Use in Cryptography, in *Progress in Cryptology* – INDOCRYPT 2001, LNCS Vol. 2247 (Springer-Verlag, 2001), pp. 99-107.
4. 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.
5. I. Blake, G. Seroussi, and N. Smart, *Elliptic curves in cryptography*, London Mathematical Society Lecture Note Series 265, Cambridge Univ. Press, 1999.
6. D. Boneh, B. Lynn, and H. Shacham, Short signatures from the Weil pairing, in *ASIACRYPT 2001*, LNCS 2248, pp. 514-532, Springer-Verlag, 2001.

7. H. Cohen, *A Course in Computational Algebraic Number Theory*, Graduate Texts in Mathematics, **138**, Springer-Verlag, Berlin, 1993.
8. G. Cornacchia, Su di un metodo per la risoluzione in numeri interi dell' equazione $\sum_{h=0}^{n} C_h x^{n-h} y^h = P$, *Giornale di Matematiche di Battaglini* 46 (1908), pp. 33-90.
9. A. Enge and F. Morain, Comparing Invariants for Class Fields of Imaginary Quadratic Fields, in *ANTS V*, LNCS Vol. 2369, pp. 252-266, 2002.
10. GNU multiple precision library, edition 3.1.1, September 2000. Available at: `http://www.swox.com/gmp`.
11. N. Gura, H. Eberle, and S.C. Shantz, Generic Implementations of Elliptic Curve Cryptography using Partial Reduction, in *Proc. 9th ACM Conf. on Computer and Communications Security* – CCS'02, pp. 108-116.
12. A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, and M. Yung, Proactive Public Key and Signature Systems, in *Proc. 4th ACM Conf. on Computer and Communications Security* – CCS'97, pp. 100-110.
13. IEEE P1363/D13, *Standard Specifications for Public-Key Cryptography*, 1999. `http://grouper.ieee.org/groups/1363/tradPK/draft.html`.
14. LiDIA. *A library for computational number theory*, Technical University of Darmstadt. Available from `http://www.informatik.tu-darmstadt.de/TI/LiDIA/Welcome.html`.
15. E. Kaltofen, T. Valente, and N. Yui, An Improved Las Vegas Primality Test, in *Proc. ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation*, pp. 26-33, 1989.
16. E. Kaltofen and N. Yui, Explicit construction of the Hilbert class fields of imaginary quadratic fields by integer lattice reduction. Research Report 89-13, Renseelaer Polytechnic Institute, May 1989.
17. E. Konstantinou, Y. Stamatiou, and C. Zaroliagis, On the Efficient Generation of Elliptic Curves over Prime Fields, in *Cryptographic Hardware and Embedded Systems* – CHES 2002, LNCS Vol. 2523 (Springer-Verlag, 2002), pp. 333-348.
18. G.J. Lay and 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-Verlag, pp. 250-263, 1994.
19. F. Morain, Building Cyclic Elliptic Curves Modulo Large Primes, in *Advances in Cryptology – Eurocrypt '91*, LNCS 547 (Springer Verlag), pp. 328-336, 1991.
20. V. Müller and S. Paulus, On the Generation of Cryptographically Strong Elliptic Curves, preprint, 1997.
21. Y. Nogami and 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.
22. E. Savaş, T.A. Schmidt, and Ç.K. Koç, Generating Elliptic Curves of Prime Order, in *Cryptographic Hardware and Embedded Systems* – CHES 2001, LNCS Vol. 2162 (Springer-Verlag, 2001), pp. 145-161.
23. J. H. Silverman, *The Arithmetic of Elliptic Curves*, Springer, GTM 106, 1986.
24. A.-M. Spallek, *Konstruktion einer elliptischen Kurve über einem endlichen Körper zu gegebener Punktegruppe*, Master Thesis, Universität GH Essen, 1992.
25. T. Valente, *A distributed approach to proving large numbers prime*, Rensselaer Polytechnic Institute Troy, New York, PhD Thesis, August 1992.
26. A. Weng, *Konstruktion kryptographisch geeigneter Kurven mit komplexer Multiplikation*, PhD thesis, Institut für Experimentelle Mathematik, Universität GH Essen, 2001.