Journal of Number Theory 131 (2011) 815–831

Contents lists available at ScienceDirect

Journal of Number Theory

www.elsevier.com/locate/jnt

Special Issue: Elliptic Curve Cryptography

Computing the endomorphism ring of an ordinary elliptic curve over a finite field

Gaetan Bisson [a,*], Andrew V. Sutherland [b]

[a] *LORIA, Équipe CACAO, Campus Scientifique, Vandœuvre-lès-Nancy Cedex, France*
[b] *Massachusetts Institute of Technology, Cambridge, MA, USA*

ARTICLE INFO

Article history:
Received 17 March 2009
Revised 7 August 2009
Available online 14 December 2009
Communicated by N. Koblitz and V.S. Miller

ABSTRACT

We present two algorithms to compute the endomorphism ring of an ordinary elliptic curve E defined over a finite field $\mathbb{F}_q$. Under suitable heuristic assumptions, both have subexponential complexity. We bound the complexity of the first algorithm in terms of $\log q$, while our bound for the second algorithm depends primarily on $\log|D_E|$, where D_E is the discriminant of the order isomorphic to $\mathrm{End}(E)$. As a byproduct, our method yields a short certificate that may be used to verify that the endomorphism ring is as claimed.

1. Introduction

Let E be an ordinary elliptic curve defined over a finite field $\mathbb{F}_q$, and let π denote the Frobenius endomorphism of E. We may view π as a quadratic integer of norm q, lying in some imaginary quadratic field K with discriminant D_K:

$$\pi = \frac{t + v\sqrt{D_K}}{2} \quad \text{with} \quad 4q = t^2 - v^2 D_K. \tag{1}$$

The trace t of π can be computed in polynomial time using Schoof's algorithm [31]. The negative integer D_K is the least fundamental discriminant dividing $t^2 - 4q$; both D_K and the integer v may then be determined in subexponential time using a probabilistic factoring algorithm [28].

[*] Corresponding author.
E-mail addresses: gaetan.bisson@loria.fr (G. Bisson), drew@math.mit.edu (A.V. Sutherland).

0022-314X/$ – see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jnt.2009.11.003

The endomorphism ring of E is isomorphic to an order $\mathcal{O}(E)$ of K. Once D_K and v are known, there are only finitely many possibilities for $\mathcal{O}(E)$, since

$$\mathbb{Z}[\pi] \subseteq \mathcal{O}(E) \subseteq \mathcal{O}_K. \tag{2}$$

Here $\mathbb{Z}[\pi]$ denotes the order generated by π, with discriminant $D_\pi = v^2 D_K$ and conductor v, and $\mathcal{O}_K$ is the maximal order of K (its ring of integers). The discriminant of $\mathcal{O}(E)$ is then of the form $D_E = u^2 D_K$, where the conductor u divides v and uniquely determines $\mathcal{O}(E)$. We wish to compute u.

Recall that two elliptic curves over $\mathbb{F}_q$ are isogenous if and only if they have the same trace [21, Chapter 13, Theorem 8.4]. Thus the set $\mathrm{Ell}_t(\mathbb{F}_q)$ of elliptic curves defined over $\mathbb{F}_q$ with trace t constitutes an isogeny class. Each curve in $\mathrm{Ell}_t(\mathbb{F}_q)$ has an endomorphism ring satisfying (2), and therefore a conductor dividing v.

In his seminal thesis, Kohel describes the structure of the graph of isogenies defined on $\mathrm{Ell}_t(\mathbb{F}_q)$, and its relationship to the orders in $\mathcal{O}_K$. He applies this structure to obtain a deterministic algorithm to compute u in time $O(q^{1/3+\varepsilon})$, assuming the generalized Riemann hypothesis (GRH) [24, Theorem 24].

Here we present two new methods to compute u that further exploit the relationship between the isogeny graph and ideal class groups. Under heuristic assumptions (including, but not limited to, the GRH), we achieve subexponential running times. Both methods yield *Las Vegas* algorithms: probabilistic algorithms whose output is unconditionally correct. We rely on heuristic assumptions only to bound their expected running times.

In practice we find the algorithms perform well, and are able to handle problem sizes that were previously intractable. We give computational examples over finite fields of cryptographic size where v is large and not smooth (the most difficult case). Over a 200-bit field the total running time is around ten minutes (see Section 5).

To express our complexity bounds, we adopt the notation

$$L[\alpha, c](x) = \exp\bigl(\bigl(c + o(1)\bigr)(\log x)^\alpha (\log\log x)^{1-\alpha}\bigr).$$

Under heuristic assumptions detailed in Section 4, we derive the bound

$$L[1/2, \sqrt{3}/2](q)$$

for the complexity of Algorithm 1 (Corollary 7), and the bound

$$L\bigl[1/2 + o(1), 1\bigr]\bigl(|D_E|\bigr) + L[1/3, c_f](q)$$

for that of Algorithm 2 (Proposition 10). The $L[1/3, c_f]$ term reflects the heuristic complexity of factoring $t^2 - 4q$ using the number field sieve [9]. Algorithm 2 is slower than Algorithm 1 in general, but may be much faster when $u \ll v$.

The endomorphism ring of an elliptic curve can be a relevant security parameter in elliptic curve cryptography, as discussed in [22,23] for example. The algorithms that we use to compute D_E may additionally generate a short *certificate* that allows a third party to verify that the endomorphism ring is as claimed. Both certification and verification have heuristically subexponential running times, and one may extend the certification phase in order to reduce the verification time, as discussed in Section 4. Under the same heuristic assumptions used in our complexity bounds, the size of the certificate is $O(\log^{2+\varepsilon} q)$ (Corollary 8).

2. Preliminaries

Kohel's algorithm treats each large prime power p^k dividing v by computing the kernel of a certain *smooth isogeny* of degree n. The prime factors of n are small (polynomial in $\log v$), but n itself is large (exponential in $\log v$), and this leads to an exponential running time (see [24, Lemma 29]). We replace this computation with a walk in the isogeny graph using isogenies of low degree (heuristically, subexponential in $\log v$). This walk tests a certain *smooth relation*, and by performing corresponding computations in class groups of orders in $\mathcal{O}_K$ we are able to determine the power of p dividing u (via Corollary 4). We adapt an algorithm of McCurley [29] to efficiently find smooth relations, achieving a heuristically subexponential running time. First, we present some necessary background.

2.1. Theoretical background

Let us fix an ordinary elliptic curve E defined over a finite field $\mathbb{F}_q$, with t, D_K, and v as in (1). We may verify that E is ordinary by checking that t is nonzero modulo the characteristic of $\mathbb{F}_q$ (see [37, Proposition 4.31]).

Recall that the j-invariant $j(E)$ may be computed as a rational function of the coefficients of E and, in particular, is an element of $\mathbb{F}_q$. Over the algebraic closure of $\mathbb{F}_q$, the j-invariant uniquely identifies E up to isomorphism, but this is not true over $\mathbb{F}_q$. However, two ordinary elliptic curves with the same trace are isomorphic over $\mathbb{F}_q$ if and only if they have the same j-invariant [11, Proposition 14.19]. Thus we may explicitly represent the set $\mathrm{Ell}_t(\mathbb{F}_q)$ as a subset of $\mathbb{F}_q$, namely, the j-invariants of all elliptic curves over $\mathbb{F}_q$ with trace t, and view each element of $\mathrm{Ell}_t(\mathbb{F}_q)$ as a particular elliptic curve representing its isomorphism class.

As noted above, each curve in $\mathrm{Ell}_t(\mathbb{F}_q)$ has an associated u dividing v that identifies its endomorphism ring, and we may partition $\mathrm{Ell}_t(\mathbb{F}_q)$ into subsets $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$ accordingly. We aim to distinguish the particular subset containing E by identifying relations that hold in some $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$ but not others.

Our main tool is the action of the ideal class group $\mathrm{cl}(u^2D_K)$ of $\mathcal{O}(u^2D_K)$ (the order of K with conductor u) on the set $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$. Here we rely on standard results from the theory of complex multiplication, and the Deuring lifting theorem.

Theorem 1. *With q, t, v, and D_K as in* (1)*, let u be a divisor of v and $\mathfrak{a}$ an ideal of $\mathcal{O}(u^2D_K)$ with prime norm ℓ. Then $\mathfrak{a}$ acts on the set $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$ via an isogeny of degree ℓ, and this defines a faithful group action by $\mathrm{cl}(u^2D_K)$.*

For a proof, see Theorems 10.5, 13.12, and 13.14 in [26], or Chapter 3 of [24]. For additional background, we also recommend [11] and [34, Chapter II].

Theorem 1 implies that the cardinality of $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$ is a multiple of the class number $h(u^2D_K)$, and in fact these values are equal [32]. In general, the curves ℓ-isogenous to E need not belong to $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$. However, when ℓ does not divide v, we have the following result of Kohel [24, Proposition 23]:

Theorem 2. *Let ℓ be a prime not dividing v. There are exactly $1 + (D_E \mid \ell)$ isogenies of degree ℓ starting from E, and they all lead to curves with endomorphism ring isomorphic to $\mathcal{O}(E)$.*

The notation $(D_E \mid \ell)$ is the Kronecker symbol. Note that $(D_E \mid \ell) = (D_K \mid \ell)$, so we can compute it without knowing D_E. We are primarily interested in the case $(D_E \mid \ell) = 1$, where the prime ℓ splits into distinct prime ideals of norm ℓ in $\mathcal{O}(E)$, and these ideals lie in inverse ideal classes α and α^{-1} in $\mathrm{cl}(D_E)$ (if ℓ splits into principal ideals, then $\alpha = \alpha^{-1} = 1$). By Theorem 1, the orbit of E under the action of α corresponds to a cycle of ℓ-isogenies whose length is equal to the order of α in $\mathrm{cl}(D_E)$. Additional details on the structure of the isogeny graph can be found in [24] and, in a more concise way, in [14].

2.2. Explicit computation

We implement class group computations via binary quadratic forms. For a negative discriminant D, the ideals in $\mathcal{O}(D)$ correspond to primitive, positive-definite, binary quadratic forms $ax^2 + bxy + cy^2$ with discriminant $D = b^2 - 4ac$. We use the triple (a, b, c) to denote such a form, where a is equal to the norm of the corresponding ideal. Ideal classes in $\mathrm{cl}(D)$ are uniquely represented by reduced forms. As typically implemented, the group operation has complexity $O(\log^2 |D|)$, see [5].[1]

To navigate the isogeny graph, we rely on the classical modular polynomial $\Phi_\ell(X, Y)$, which parametrizes pairs of ℓ-isogenous elliptic curves. This is a symmetric polynomial with integer coefficients. For a prime ℓ not dividing q, two elliptic curves E_1 and E_2 defined over $\mathbb{F}_q$ are connected by an isogeny of degree ℓ if and only if $\Phi_\ell(j(E_1), j(E_2)) = 0$, by [37, Theorem 12.19].[2]

The polynomial Φ_ℓ has size $O(\ell^3 \log \ell)$, by [10], and may be computed in time $O(\ell^{3+\varepsilon})$ using [7] or [13]. When ℓ is small we use precomputed $\Phi_\ell \in \mathbb{Z}[X, Y]$, but for larger ℓ we directly compute $\Phi_\ell/\mathbb{F}_q$, the polynomial Φ_ℓ reduced modulo the characteristic of $\mathbb{F}_q$. As a practical optimization, one may also consider alternative modular polynomials that are sparser and have smaller coefficients than Φ_ℓ.

To find the curves that are ℓ-isogenous to E, we compute the roots of the univariate polynomial $f(X) = \Phi_\ell(X, j(E))$ in $\mathbb{F}_q$. We may restrict ourselves to primes $\ell \nmid v$ with $(D_E \mid \ell) = 1$, so that $f(X)$ has exactly two roots, by Theorem 2. We find these roots by computing $\gcd(f, X^q - X)$ and solving the resulting quadratic, using an expected $O(\mathsf{M}(\ell) \log q)$ operations in $\mathbb{F}_q$ (this is the time to compute $X^q \bmod f$). Given $\Phi_\ell/\mathbb{F}_q$, we use $O(\ell^2)$ operations in $\mathbb{F}_q$ to construct $f(X) = \Phi_\ell(X, j(E))$. For $\ell \gg \log q$ this dominates the time to find the roots of $f(X)$ and bounds the cost of taking a single step in the ℓ-isogeny graph.

2.3. Relations

Let us suppose that $\alpha \in \mathrm{cl}(D_E)$ contains an ideal of prime norm $\ell \nmid v$, and has order $e = |\alpha|$. In this situation we say that the relation $\alpha^e = 1$ holds in $\mathrm{cl}(D_E)$. We cannot actually compute α^e in $\mathrm{cl}(D_E)$, since we do not yet know D_E, but we may apply Theorem 1 to compute the action of either α^e or α^{-e} on E by walking a distance e along the cycle of ℓ-isogenies, starting from $j = j(E)$.

Algorithm. WalkCycle(j, ℓ, e):

```
1. Set $j_0 \leftarrow j$.
2. Let $j_1$ be one of the two roots of $\Phi_\ell(X, j_0)$.
3. For $s$ from 1 to $e - 1$:
4.     Let $j_{s+1}$ be the root of $\Phi_\ell(X, j_s)/(X - j_{s-1})$.
5. Return $j_e$.
```

The symmetry of $\Phi_\ell(X, Y)$ implies that j_{s-1} must be a root of $\Phi_\ell(X, j_s)$ in Step 4. The roots of $\Phi_\ell(X, j_s)$ are typically distinct (any exceptions require $|D_E| \leqslant 4\ell^2$, by [14, Theorem 2.2]), but the algorithm applies in any case.

The choice of j_1 in Step 2 is arbitrary, it may correspond to the action of either α or α^{-1}. Nevertheless, since $e = |\alpha| = |\alpha^{-1}|$, we have $j_e = j_0$ in either case. A difficulty arises when we consider a relation that is not unary, say $\alpha_1^{e_1} \alpha_2^{e_2} = 1$, where α_i contains an ideal of prime norm ℓ_i with $\ell_1 \neq \ell_2$. Starting from $j(E)$, we walk e_1 steps along the ℓ_1-isogeny cycle, then walk e_2 steps along the ℓ_2-isogeny cycle. We make two arbitrary choices here, and may compute the action of $\alpha_1^{e_1} \alpha_2^{e_2}$, $\alpha_1^{e_1} \alpha_2^{-e_2}$, $\alpha_1^{-e_1} \alpha_2^{e_2}$ or $\alpha_1^{-e_1} \alpha_2^{-e_2}$. The actions of these four elements are almost certainly not identical; even if $\alpha_1^{e_1} \alpha_2^{e_2} = 1$ in $\mathrm{cl}(D_E)$, it is unlikely that $\alpha_1^{e_1} \alpha_2^{-e_2}$ will fix $j(E)$.

[1] The algorithm of [30] has complexity $O(\log^{1+\varepsilon} |D|)$, but we do not make use of it.

[2] This isogeny is necessarily cyclic, since it has prime degree.

To address this situation, we formally define a *relation* R as a pair of vectors $(\ell_1, \ldots, \ell_k)$ and $(e_1, \ldots, e_k)$, where each ℓ_i is prime, $\ell_i \nmid v$ and $(D_K \mid \ell_i) = 1$, and each e_i is a positive integer.[3] The integer k is the *arity* of the relation. Given a discriminant $D = u^2 D_K$ with $u \mid v$, choose ideal classes $\alpha_1, \ldots, \alpha_k \in \mathrm{cl}(D)$ so that α_i contains an ideal of norm ℓ_i. This ideal need not be the reduced representative of α_i, and may be principal (implying $\alpha_i = 1$); this depends on D. We now define

$$\#R/D := \#\left\{\tau \in \{\pm 1\}^{\{1,\ldots,k\}}\colon \prod_{i=1}^{k} \alpha_i^{\tau_i e_i} = 1\right\} \tag{3}$$

as the *cardinality* of the relation R in $\mathrm{cl}(D)$. When $\#R/D > 0$, we say R *holds* in $\mathrm{cl}(D)$. The integer $\#R/D$ is independent of the choice of the α_i. It has even parity, since if τ belongs to the set in (3), so does $-\tau$.

To compute $\#R/D_E$, we enumerate the 2^k possible walks we may take in the isogeny graph, starting from $j(E)$, considering all possible sign vectors τ (these walks typically form a tree in which each path from root to leaf has k binary branch points). By the symmetry noted above, we may fix $\tau_1 = 1$.

Algorithm. CountRelation(E, R):

```
1. Compute $j_0 \leftarrow$ WalkCycle($j(E), \ell_1, e_1$) and let $J$ be the list $(j_0)$.
2. For $i$ from 2 to $k$:
3.     Set $J' \leftarrow J$ and then set $J$ to the empty list.
4.     For each $j_0 \in J'$:
5.         Let $j'_0 = j_0$ and let $j_1$ and $j'_1$ be the two roots of $\Phi_{\ell_i}(X, j_0)$.
6.         For $s$ from 1 to $e_i - 1$:
7.             Let $j_{s+1}$ be the root of $f(X) = \Phi_{\ell_i}(X, j_s)/(X - j_{s-1})$.
8.             Let $j'_{s+1}$ be the root of $f(X) = \Phi_{\ell_i}(X, j'_s)/(X - j'_{s-1})$.
9.         Append $j_{e_i}$ and $j'_{e_i}$ to $J$.
10. Return $2n$, where $n$ counts the occurrences of $j(E)$ in $J$.
```

Given $\Phi_\ell/\mathbb{F}_q$, the complexity of Algorithm CountRelation is dominated by

$$\sum_{i=1}^{k} 2^{i-1} e_i T(\ell_i), \tag{4}$$

where $T(\ell)$ is the time to take a single step in the ℓ-isogeny graph, which for large ℓ is bounded by $O(\ell^2)$ operations in $\mathbb{F}_q$, as noted above. Our algorithms rely on *smooth* relations in which k, ℓ_i, and e_i are all rather small: in the first example of Section 5 we have $k = 10$, $\ell_i \leqslant 600$, and $e_i \leqslant 2000$. As a practical optimization, we order the couples (ℓ_i, e_i) to minimize (4), using an estimate of $T(\ell)$.

Computing $\#R/D$ in $\mathrm{cl}(D)$ (where D is known) is straightforward: one computes the set in (3) by evaluating products of powers in $\mathrm{cl}(D)$. A total of $O(2^k + \sum \log e_i)$ operations in the class group suffice (regardless of the ℓ_i).

Remark. Provided $\mathbb{Z}[\pi]$ is maximal at ℓ (meaning $\ell \nmid v$), there is a way to distinguish which of the two directions on an ℓ-isogeny cycle corresponds to the action of a particular prime ideal of norm ℓ in $\mathcal{O}(E)$. This involves techniques adapted from the SEA algorithm, and is described in [15, §3] (and also [6, §5]). As an optimization, this approach could reduce the time to test relations, and would eliminate the need to work with the notion of cardinality we define above, at the cost of a more intricate implementation.

[3] In practice, we relax the constraint $\ell_i \nmid v$ for very small ℓ_i (especially $\ell_i = 2$), see [36, Algorithm 4.2].

However, this does not impact the overall complexity of our two main algorithms, which are dominated by the tasks of finding suitable relations and computing the polynomials $\Phi_\ell/\mathbb{F}_q$. This is true both asymptotically (see Section 4) and in practice (see Section 5). The main benefit of such an optimization would likely be a speedup in the time to verify a certificate in Algorithm Verify (defined in Section 3).

2.4. Probing class groups

We now consider how we may distinguish class groups of orders in K by computing the cardinality of suitable relations. We rely on the following lemma.

Lemma 3. *Suppose $\mathcal{O}(D_1) \subseteq \mathcal{O}(D_2)$. Then for every relation R we have*

$$\#R/D_1 \leqslant \#R/D_2.$$

Proof. Let $\mathfrak{a}$ be an $\mathcal{O}(D_1)$-ideal with norm prime to the conductor of D_1. The map

$$\mathfrak{a} \mapsto \mathfrak{a}\mathcal{O}(D_2)$$

induces a natural morphism of class groups. It preserves norms (see [11, Proposition 7.20] for a proof in the case D_2 is fundamental, from which one easily derives the general case) and therefore transports relations from $\mathrm{cl}(D_1)$ to $\mathrm{cl}(D_2)$. □

Corollary 4. *Let p^k be a prime power dividing v, and let $D_1 = (v/p^j)^2 D_K$ and $D_2 = p^{2k} D_K$, where $j = \nu_p(v) - k + 1$. Suppose $\#R/D_1 > \#R/D_2$ for some relation R, and let $D = u^2 D_K$ where $u \mid v$. Then $p^k \mid u$ if and only if $\#R/D < \#R/D_1$.*

Provided we have a suitable relation R for each prime-power p^k dividing v, we can apply the corollary to $D = D_E$ to determine the prime-power factorization of u, and hence the endomorphism ring of E. The computations of $\#R/D_1$ and $\#R/D_2$ are performed in the class groups $\mathrm{cl}(D_1)$ and $\mathrm{cl}(D_2)$, but the computation of $\#R/D_E$ takes place in the isogeny graph via the CountRelation algorithm. Notice that we may replace v in the corollary by any multiple of u dividing v.

Proposition 5. *For all primes $p > 3$, there are infinitely many relations satisfying the assumptions of Corollary 4.*

Proof. Consider unary relations with $e_1 = 1$ and $\ell_1 = \ell$, and denote them R_ℓ. The relation R_ℓ holds in $\mathrm{cl}(D)$ precisely when ℓ splits into principal ideals in $\mathcal{O}(D)$. For $i \in \{1, 2\}$, let S_i be the set of primes ℓ such that R_ℓ holds in $\mathrm{cl}(D_i)$. It suffices to show that $S_1 \setminus S_2$ is infinite.

The set S_i is equal to the set of primes that split completely in the ring class field L_i of $\mathcal{O}(D_i)$, which is a Galois extension of $\mathbb{Q}$ (see [11, Lemma 9.3]). The Chebotarev density theorem [11, Theorem 8.19] implies that S_1 and S_2 are both infinite, and $S_1 \setminus S_2$ is finite if and only if $L_1 \subseteq L_2$.

But L_1 cannot be contained in L_2, for $\mathcal{O}(D_1)$ is not contained in $\mathcal{O}(D_2)$. Indeed, p^k divides the conductor of $\mathcal{O}(D_2)$ but not that of $\mathcal{O}(D_1)$, which for $p > 3$ implies that p^k divides the conductor of L_2 but not that of L_1 (see [11, Exercises 9.20–9.23]). □

Many relations other than those considered in the proof above may also satisfy the requirements of Corollary 4. Empirically, relations R holding in $\mathrm{cl}(D_1)$ satisfy $\#R/D_1 > \#R/D_2$ with probability converging to 1 as p grows. We will not attempt to prove this statement, but as a heuristic assume that this probability is at least bounded above zero, and furthermore that this applies to relations that are smooth (as defined in Section 4). Note that, independent of this assumption, the above proposition guarantees that our algorithms are always able to terminate.

3. Algorithms

3.1. Computing $\mathcal{O}(E)$ from above

We now describe our first algorithm to compute u, the conductor of the order $\mathcal{O}(E)$ isomorphic to $\mathrm{End}(E)$. We rely on Algorithm FindRelation(D_1, D_2), described in Section 3.4, to obtain relations to which Corollary 4 may be applied.

For small primes p dividing v, say all $p \leqslant B$ for some B, we can efficiently determine the largest prime power p^k dividing u by isogeny climbing, as described in [24, Section 4.2] and [36, Section 4.1]. This yields an isogenous curve E' for which the conductor of $\mathcal{O}(E')$ is $u' = u/p^k$, using $O(kp^2 \log q)$ operations in $\mathbb{F}_q$ (given $\Phi_p/\mathbb{F}_q$).

For simplicity, the algorithm below assumes that v is not divisible by the square of a prime larger than B. The modification to handle large primes whose square divides v is straightforward but unlikely to be needed in practice.

Algorithm 1. $(E/\mathbb{F}_q)$:

```
1. Let Schoof's algorithm compute the trace $t$ of $E$,
   then determine $D_K$, $v$, and the prime factors of $v$, by factoring $v^2 D_K = t^2 - 4q$.
2. Select a bound $B$ and set $u \leftarrow 1$.
3. For each prime $p \leqslant B$ dividing $v$:
4.     Determine the largest power of $p$ dividing $u$ by isogeny climbing,
       then set $E \leftarrow E'$, remove all powers of $p$ from $v$, and update $u$.
5. For each prime $p > B$ that divides $v$:
6.     Set $R \leftarrow$ FindRelation$(D_1, D_2)$, where $D_1 = (v^2/p^2)D_K$ and $D_2 = p^2 D_K$.
7.     Determine whether $p$ divides $u$ by checking if $\#R/D_E < \#R/D_1$,
       then update $u$ appropriately.
8. Return $u$.
```

The correctness of Algorithm 1 follows from Corollary 4. Its running time depends on the choice of B and the complexity of FindRelation. We obtain in Section 4 (Corollary 7) a heuristic bound of

$$L[1/2, \sqrt{3}/2](q)$$

on the expected running time of Algorithm 1, using $L[1/2, 1/\sqrt{3}](q)$ space.

Note that the relations computed in Algorithm 1 only depend on the Frobenius trace t of E, not its endomorphism ring, hence they may be reused to compute the endomorphism ring of any curve in the same isogeny class. These relations also provide a means to subsequently verify the computation of u, but for this purpose we may wish to specialize the relations to u, a task we now consider.

3.2. Certifying u

Let us suppose that a particular value u is claimed as the conductor of $\mathcal{O}(E)$. This may arise in a situation where u is actually known, either via Algorithm 1 or from the construction of E (say, by the CM method), but may also occur when one wishes to test a provisional value of u, as we will do in Algorithm 2. We first give an algorithm to construct a *certificate* that may be used to efficiently check whether a given curve with trace t in fact has endomorphism ring $\mathcal{O}(E)$ with conductor u. Equivalently, it tests whether an element of $\mathrm{Ell}_t(\mathbb{F}_q)$ lies in $\mathrm{Ell}_{t,u}(\mathbb{F}_q)$.

The construction of this certificate depends only on u, v, and D_K and does not require an elliptic curve as input. Small prime factors of u and v may be removed by isogeny climbing prior to calling Certify.

Algorithm. Certify(u, v, D_K):

```
1. For each prime factor $p$ of $v/u$:
2.     Set $R_p \leftarrow$ FindRelation($D_1, D_2$), where $D_1 = u^2 D_K$ and $D_2 = p^2 D_K$.
3. For each prime factor $p$ of $u$:
4.     Set $R_p \leftarrow$ FindRelation($D_1, D_2$), where $D_1 = (u^2/p^2) D_K$ and $D_2 = u^2 D_K$.
5. Return $C = (u, v, D_K, \{R_p\}_{p|v})$.
```

The relations computed in Step 2 may verify that the actual value of u divides the claimed value, whereas the relations computed in Step 4 may verify that the claimed value of u is not a proper divisor of u, as shown by Algorithm Verify.

Algorithm. Verify($E/\mathbb{F}_q, C$):

```
1. For each prime factor $p$ of $v/u$, verify that $\#R_p/E > \#R_p/p^2 D_K$.
2. For each prime factor $p$ of $u$, verify that $\#R_p/(u^2/p^2) D_K > \#R_p/E$.
3. Return true if all verifications succeed and false otherwise.
```

In addition to the verification of u above, one may also wish to verify that v and D_K are correct. This may be accomplished in polynomial time if the trace t and the factorizations of v and D_K are included in the certificate. One may additionally wish to certify the primes in these factorizations [2], or the verifier may apply a polynomial-time primality test [1]. Assuming these values are correct, the conductor of $\mathcal{O}(E)$ is equal to u if and only if Verify(E, C) returns `true`. This statement does not depend on any unproven hypotheses.

The size of the certificate is unconditionally bounded by $O(\log^3 q)$, and under heuristic assumptions we obtain an $O(\log^{2+\varepsilon} q)$ bound (Corollary 8). Within this bound, certificates for primes dividing v or D_K can be included, as each certificate requires $O(\log^{1+\varepsilon} q)$ space and there are $O(\log q)$ such primes.

The expected running times of Certify and Verify depend on a smoothness parameter μ used by FindRelation. This parameter may be chosen to balance the cost of certification and verification, as in Algorithm 2 below, or one may reduce the verification time by increasing the certification time. See Proposition 9 and the discussion following for an analysis of this trade-off.

3.3. *Computing $\mathcal{O}(E)$ from below*

We now present a second algorithm to compute u, which may be much faster than Algorithm 1 if u is small compared to v, and is in general only slightly slower. Our basic strategy is to examine each of the divisors u_i of v in order, attempting to prove that $u = u_i$ is the conductor of $\mathcal{O}(E)$, by constructing a certificate and verifying it. This only requires finding relations in class groups with discriminants whose absolute value is at most $|u^2 D_k|$.

Typically, v has few divisors (almost always $O(\log^{\log 2} v)$, by [19, p. 265]), in which case this basic strategy is quite effective. However, in order to improve performance in the worst case, we apply isogeny climbing to effectively remove prime factors from v as we go, thereby reducing the number of u's we must consider. As above, we suppose v is square-free for the sake of presentation.

The algorithm uses an auxiliary variable w to bound the complexity of isogeny climbing using a known lower bound for u that increases as the algorithm proceeds. Initially we have no information about u so we use the cost of the factorization computed in Step 1 to select w.

Algorithm 2. $(E/\mathbb{F}_q)$:

```
1. Let Schoof's algorithm compute the trace $t$ of $E$, then determine $D_K$, $v$, and the prime factors of $v$ by factoring $v^2 D_K = t^2 - 4q$.
2. Set $x \leftarrow 0$.
3. Set $w \leftarrow \max(1/3, x/2 + 1/\log q)$.
4. For primes $p < \exp(\log^w v)$:
5.     Test whether $p \mid u$ by isogeny climbing, then set $E \leftarrow E'$, $v \leftarrow v/p$.
6. For divisors $u$ of $v$ less than $\exp(\log^{2w} v)$:
7.     If Verify($E$, Certify($u, v, D_k$)) returns true:
8.         Return the product of $u$ and the primes determined in Step 5.
9. Set $x \leftarrow 2w$ and go to Step 3.
```

The running time of Algorithm 2 is analyzed in Section 4, where the bound

$$L\big[1/2 + o(1), 1\big]\big(|D_E|\big) + L[1/3, c_f](q)$$

is obtained under suitable heuristic assumptions. The same assumptions yield a space bound of $L[1/2 + o(1), 2/3](|D_E|)\log q$.

3.4. *Finding relations*

Given negative discriminants D_1 and D_2, we seek a relation R satisfying $\#R/D_1 > \#R/D_2$. We find such an R by searching for a relation that holds in $\mathrm{cl}(D_1)$ and then testing this inequality. As noted at the end of Section 2, this test almost always succeeds, but if not we search for another relation.

To find relations that hold in $\mathrm{cl}(D_1)$, we adapt an algorithm of McCurley [18,29]. Let us fix a smoothness bound B. For each prime $\ell \leqslant B$ with $(D_1 \mid \ell) \neq -1$, let f_ℓ denote the *primeform* with norm ℓ. By this we mean the binary quadratic form (ℓ, b_ℓ, c_ℓ) of discriminant D_1 with $b_\ell \geqslant 0$, which may be constructed via [8, Algorithm 3.3].

We now generate reduced forms by computing products

$$(a, b, c) = \prod_\ell f_\ell^{x_\ell}, \tag{5}$$

where the x_ℓ are randomly chosen subject to certain constraints (and mostly zero). If the prime factors of a are bounded by B, say $a = \prod_\ell \ell^{y_\ell}$, then we may decompose the form (a, b, c) as

$$(a, b, c) = \prod_\ell f_\ell^{\tau_\ell y_\ell}, \tag{6}$$

where for nonzero y_ℓ, $\tau_\ell = \pm 1$ is defined by $b = \tau_\ell b_\ell \bmod 2\ell$.

Recall that $n = \sqrt{|D_1|/3}$ is an upper bound on the norm of a reduced imaginary quadratic form [12, Exercise 5.14]. Provided that $\prod_\ell \ell^{|x_\ell|} > n$, the decompositions in (5) and (6) must be different, since $a \leqslant n$. This yields a non-trivial relation with exponents $e_\ell = x_\ell - \tau_\ell y_\ell$.

In order to minimize the cost of computing $\#R/D_E$ (via CountRelation) for the relations we obtain, in addition to bounding the primes ℓ, we must also bound the exponents e_ℓ, and especially the number of nonzero e_ℓ, which determines the arity k of R. To achieve this we require all but a constant number k_0 of the x_ℓ to be zero (we use $k_0 = 3$), and note that a random B-smooth integer $a \in [1, n]$ is expected to have approximately $2\log n/\log B$ distinct prime factors. In the unlikely event that k significantly exceeds this expected value, we seek a different relation.

Having bounded k, the complexity of CountRelation then depends on the products $e_\ell T(\ell)$ appearing in (4). For large ℓ we have $T(\ell) = O(\ell^2)$ operations in $\mathbb{F}_q$. To make the products $e_\ell T(\ell)$

approximately equal we may use the bound $|x_\ell| \leqslant (B/\ell)^2$. In practice we pick a parameter $\omega > 1$ and use a bound

$$|x_\ell| \leqslant (B/\ell)^\omega$$

that better reflects the cost of $T(\ell)$ for moderate values of ℓ (we use $\omega \approx 1.6$); this has no impact on our asymptotic analysis.

We heuristically assume[4] that the norms of the forms we generate are about as likely to be B-smooth as random integers in the interval $[1, n]$. This leads us to use a smoothness bound $B = L(n)^\mu$, where

$$L(n) = \exp(\sqrt{\log n \log\log n}),$$

and μ is a parameter to be chosen. By a theorem of Hildebrand [20, Theorem 1], for B of this form, the number of B-smooth integers in the interval $[1, n]$ is asymptotic to $\rho(z)n$, where $z = \log n/\log B$ and $\rho(z)$ is the Dickman function. We thus expect that for sufficiently large n, a uniformly random sample of

$$m = 1/\rho(\log n/\log B)$$

integers in $[1, n]$ will contain at least one B-smooth integer, with probability greater than $1 - \exp(-1) > 1/2$. See [17] for a survey of these and related results, including algorithms to compute $\rho(z)$ and a table of commonly used values.

To allow for the possibility that none of the primeforms generated according to our constraints have B-smooth norms (or that none of the relations we find are suitable), we increase the smoothness bound by a constant factor r slightly greater than 1 if we fail to find a suitable relation after testing $2m$ elements.

Algorithm. FindRelation(D_1, D_2):

```
1. Set $B \leftarrow L(n)^\mu$, where $n = \sqrt{|D_1|/3}$, and set $m \leftarrow 1/\rho(\log n/\log B)$.
2. Compute primeforms $f_\ell$ for $\ell \leqslant B$.
3. Repeat $2m$ times:
4.     Pick random integers $x_\ell$ with $|x_\ell| \leqslant (B/\ell)^\omega$
       such that at most $k_0$ of the $x_\ell$ are nonzero and $\prod_\ell \ell^{|x_\ell|} > n$.
5.     Compute the reduced form $(a, b, c) = \prod_\ell f_\ell^{x_\ell}$.
6.     If $a$ is $B$-smooth:
7.         Let $R$ be the relation with $e_\ell = |x_\ell - \tau_\ell y_\ell|$ where $a = \prod_\ell \ell^{y_\ell}$, and let $k$ be the arity of $R$.
8.         If $k < (2/\mu)\log^{1/2} n$ and $\#R/D_1 > \#R/D_2$, then return $R$.
9. Set $B \leftarrow rB$ and go to Step 2.
```

As a practical optimization, we may choose not to generate completely new values for x_ℓ every time Step 4 is executed, instead changing just one bit in some nonzero x_ℓ. This allows the form (a, b, c) to be computed in most cases with a single composition/reduction using a precomputed set of binary powers of the f_ℓ.

To implement Step 6 one may use the elliptic curve factorization method (ECM) to identify B-smooth integers in expected time $L[1/2, 2](B) = L[1/4, 2\mu](n)$, which effectively makes the cost of smoothness testing negligible within the precision of our subexponential complexity bounds. A faster approach uses Bernstein's algorithm, which identifies the smooth numbers in a given list in essentially linear time [4]. This does not change our complexity bounds, so for the sake of simplicity we use ECM in our analysis.

[4] This is known to be true for random forms, see [8, Proposition 11.4.3].

In practice, the bound B is quite small (under 1000 in both our examples), and very little time is spent on smoothness testing. In our implementation we use a combination of trial division and a restricted form of Bernstein's algorithm.

4. Complexity analysis

The complexity bounds derived below depend on the following heuristics:

(1) **Small primes.** We assume the GRH. The effective Chebotarev bounds of Lagarias and Odlyzko then imply that for all $x = \Omega(\log^{2+\varepsilon}|D_K|)$ there are $\Omega(x/\log x)$ primes less than x that split in $\mathcal{O}_K$, where the implied constants are all effectively computable [25, Theorem 1.1].

(2) **Random norms.** We assume that the norms of the reduced forms computed in Step 4 of FindRelation have approximately the distribution of random integers in $[1, n]$. We use this to estimate the probability of generating a form whose norm is B-smooth.

(3) **Random relations.** If $D_1 = u_1^2 D_K$ and $D_2 = u_2^2 D_K$ are sufficiently large discriminants with $u_2 \nmid u_1$, and R is a random relation for which $\#R/D_1 > 0$, with ℓ_i and e_i bounded as in FindRelation, then we assume that $\#R/D_1 > \#R/D_2$ with probability bounded above zero.

(4) **Integer factorization.** We assume that ECM finds a prime factor p of an integer n in expected time $L[1/2, 2](p)\log^2 n$, by [27], and that the expected running time of the number field sieve is $L[1/3, c_f](n)$, by [9].

In the propositions and corollaries that follow, we use the shorthand (**H**) to indicate that we are assuming Heuristics 1–4 above.

Proposition 6. (**H**) FindRelation(D_1, D_2) *has expected running time*

$$L\bigl[1/2, 1/(\sqrt{8}\mu)\bigr]\bigl(|D_1|\bigr) + L[1/2, 0]\bigl(|D_1|\bigr)\log^3|D_2|.$$

The output relation R has norms ℓ_i bounded by $L[1/2, \mu/\sqrt{2}](|D_1|)$, exponents e_i bounded by $L[1/2, \sqrt{2}\mu](|D_1|)$, and arity $k < (2/\mu)\log^{1/2}|D_1|$.

Proof. We have $B = L(n)^\mu = L[1/2, \mu/\sqrt{2}](|D_1|)$, where $n = \sqrt{D_1/3}$. Using the bound $1/\rho(z) = z^{z+o(1)}$ we find that $m = L[1/2, 1/(2\mu)](n)$. Under Heuristic 1, for sufficiently large B there are at least $\Omega(\log B)$ primes $\ell = O(\log^2 B)$ for which $(D_1 \mid \ell) = 1$. For these ℓ, the value of $|x_\ell|$ may range up to $B^{\omega-\varepsilon} > B$. There are thus $\Omega(B^{\log B}) \gg m$ distinct forms that may be generated in Step 4, and with high probability at least m are. So Heuristic 2 applies, and with probability at least 1/2 we generate at least one form with B-smooth norm each time Step 3 is executed.

Heuristic 2 also implies that the expected number k of nonzero exponents e_i is at most

$$k_0 + 2\log n/\log B = k_0 + \frac{1}{\mu}\log^{1/2}|D_1|\bigl(\log\log|D_1|\bigr)^{-1/2},$$

since we expect a random B-smooth integer in $[1, n]$ to have $(2 + o(1))\log n/\log B$ distinct prime factors (this may be proven with the random bisection model of [3]). This, together with Heuristic 3, ensures that when Step 8 is reached the algorithm terminates, with some constant probability greater than zero. Thus we expect to reach Step 9 just $O(1)$ times, and the total number of forms (a, b, c) generated by the algorithm during its execution is bounded by $L[1/2, 1/(2\mu)](n)$.

For each form (a, b, c), the algorithm tests whether a is B-smooth in Step 6. Applying ECM, under Heuristic 4 we identify a B-smooth integer in time $L[1/2, 2](B) = L[1/4, \sqrt{2\mu}](n)$, with high probability. This yields

$$L\bigl[1/2, 1/(2\mu)\bigr](n) = L\bigl[1/2, 1/(\sqrt{8}\mu)\bigr]\bigl(|D_1|\bigr)$$

as a bound on the expected time spent finding relations.

The bounds on k, the ℓ_i, and the e_i are immediate. We may bound the cost of computing $\#R/D_j$, for $j \in \{1,2\}$, by

$$O\left(2^k \log(\max e_i) \log^2|D_j|\right) = O\left(2^k \log^{5/2+\varepsilon}|D_j|\right) = L[1/2,0]\big(|D_1|\big)\log^3|D_j|.$$

The proposition follows. □

Corollary 7. (**H**) *Algorithm* 1 *has expected running time* $L[1/2,\sqrt{3}/2](q)$.

Proof. We may compute t in polynomial time with Schoof's algorithm, and under Heuristic 4 we factor $v^2D_K = t^2 - 4q$ in expected time $L[1/3, c_f](q)$.

We use $B = L(q)^{1/\sqrt{12}}$ in Algorithm 1, and set the parameter $\mu = 1/\sqrt{6}$ when calling FindRelation. The cost of isogeny climbing, the calls to FindRelation, and the calls to CountRelation to compute $\#R/D_E$ all have expected complexity $L[1/2,\sqrt{3}/2](q)$, including the cost of computing the required $\Phi_\ell/\mathbb{F}_q$. Only $O(\log q)$ iterations are required in Algorithm 1 (one for each $p \mid v$), which does not change the complexity bound. □

Corollary 8. (**H**) *Let* $D_1 = u^2D_K$ *and* $D_2 = v^2D_K$. *The expected running time of* Certify(u, v, D_K) *is within an* $O(\log v)$ *factor of the expected complexity of* FindRelation(D_1, D_2). *The output certificate* C *has size* $O(\log^{1+\varepsilon}|D_1|\log v)$.

Proof. Algorithm Certify makes fewer than $O(\log v)$ calls to FindRelation with $|D_1| \leqslant |u^2D_K|$ and $|D_2| \leqslant |v^2D_K|$. Applying the bounds of Proposition 6 for ℓ_i, e_i, and k, each relation has size $O(\log|D_1|\log\log|D_1|)$. □

Proposition 9. (**H**) *Given a certificate* C *produced by Algorithm* Certify *with parameter* μ *and an elliptic curve* $E/\mathbb{F}_q$, *Algorithm* Verify$(E/\mathbb{F}_q, C)$ *has expected running time*

$$L[1/2, 3\mu/\sqrt{2}]\big(\left|u^2D_K\right|\big)\log^{5/2} q.$$

Proof. The expected time to compute $\Phi_\ell/\mathbb{F}_q$ is $O(\ell^{3+\varepsilon}\log^{1+\varepsilon} q)$, see [7,13]. By Proposition 6, each relation in the certificate contains $O(\log^{1/2}|D_1|)$ distinct ℓ_i, each bounded by $L[1/2,\mu/\sqrt{2}](|u^2D_K|)$. There are at most $O(\log q)$ relations in the certificate, yielding a total time of

$$L[1/2, 3\mu/\sqrt{2}]\big(\left|u^2D_K\right|\big)\log^{5/2} q$$

to compute all the $\Phi_\ell/\mathbb{F}_q$ needed for verification. The total cost of all calls to CountRelation may be bounded by

$$L[1/2, \sqrt{2}\mu]\big(\left|u^2D_K\right|\big)\log^{2+\varepsilon} q,$$

using fast multiplication in $\mathbb{F}_q$, which is dominated by the bound above. □

To balance the costs of verification and certification, one uses $\mu = 1/\sqrt{6}$. The verification time may be reduced (and the certification time increased) by making μ smaller. For example, with $\mu = 1/\sqrt{18}$ the verification time is $L[1/2,1/2](|u^2D_K|)$ and the certification time is $L[1/2,3/2](|u^2D_K|)$, ignoring logarithmic factors in q.

Proposition 10. (**H**) *Algorithm* 2 *has expected running time*

$$L\big[1/2+o(1),1\big]\big(|D_E|\big)+L[1/3,c_f](q).$$

Proof. In Step 1 we compute t in polynomial time and factor $v^2D_K=t^2-4q$ in expected time $L[1/3,c_f](q)$, by Heuristic 4. Let $\mu=1/\sqrt{6}$ in all the calls to Certify, in order to balance the cost of Verify. The cost of each certification/verification performed in Step 7 is then bounded by $L[1/2,\sqrt{3}/2](|D_E|)\log^{5/2}q$, according to Proposition 9, since we never test a divisor of v that is greater than the conductor u of D_E. In Step 6, v can contain no prime factors less than $\exp(\log^w v)$. Thus the number of divisors is bounded by

$$\binom{\log^{1-w}v}{\log^{2w-w}v}\leqslant\big(\log^{1-w}v\big)^{\log^w v}=L[w,1](v)=L\big[1/2+o(1),1\big]\big(|D_E|\big).$$

In the rightmost equality we have used

$$\log|D_E|>\log u\geqslant\log^{2w-1/\log q}v\quad\Rightarrow\quad\log v\leqslant\log^{1/(2w-1/\log q)}|D_E| \tag{7}$$

to express the bound in terms of $|D_E|$, noting that

$$w/(2w-1/\log q)=1/2+1/(4w\log q-2),$$

where $w\geqslant 1/3$ and $q\to\infty$ as $|D_E|\to\infty$. The cost of Step 7 for all the divisors considered in a single execution of Step 6 is bounded by $L[1/2+o(1),1](|D_E|)\log^{5/2}q$. The algorithm may repeat Step 6 up to $\log q$ times, but the cost of each iteration dominates all prior ones, so we have bounded the total cost of Step 7.

The cost of isogeny climbing in Step 5 during the first iteration is bounded by

$$\exp\big(\big(3+o(1)\big)\log^{1/3}v\big)\log^{2+\varepsilon}q=L[1/3,c_f](q)$$

(for any c_f), and thereafter cannot exceed

$$\exp\big((3+\varepsilon)\log^w v\big)\log^{2+\varepsilon}q=L[w,1](v)\log^3 q=L\big[1/2+o(1),1\big]\big(|D_E|\big)\log^{2+\varepsilon}q.$$

Here we have again applied (7), and the choice of the constant 1 (or any constant) is justified by the fact that $3/(\log\log|D_E|)^{1-w}\to 0$ as $|D_E|\to\infty$.

To complete the proof, we note that if $L[1/2+o(1),1](|D_E|)\log^{5/2}q$ exceeds $L[1/3,c_f](q)$ we may incorporate the $\log^{5/2}q$ factor into the $o(1)$ term. Otherwise, the complexity is $L[1/3,c_f](q)$, and the proposition holds in either case. □

In both Algorithms 1 and 2, the space is dominated by the size of the polynomials $\Phi_\ell/\mathbb{F}_q$. Using the algorithm of [7] these can be computed in $O(\ell^{2+\varepsilon}\log q)$ space. Plugging in parameters from the complexity analysis above, and making the same heuristic assumptions, we obtain an $L[1/2,1/\sqrt{3}](q)$ space bound for Algorithm 1, and an $L[1/2+o(1),2/3](|D_E|)\log q$ space bound for Algorithm 2.

5. Examples

The rough timings we give here were achieved by a simple implementation running on a single 3.0 GHz AMD Phenom II core. Algorithm FindRelation was implemented using the GNU C Compiler [35] and the GMP library [16], while for CountRelation we used Shoup's NTL library [33]. We did not attempt to maximize performance; our purpose was simply to demonstrate the practicality of the algorithms on some large inputs. In a more careful implementation, constant factors would be improved and many steps could be parallelized.

5.1. First example

We consider the elliptic curve $E/\mathbb{F}_q$ with Weierstrass equation $Y^2 = X^3 - 3X + c_E$, where

$$c_E = 660897170071025494489036936911196131075522079970680898049528 \quad \text{and}$$

$$q = 1606938044258990275550812343206050075546550943415909014478299.$$

Its trace $t = 212$ is computed by the Schoof–Elkies–Atkin algorithm in a few seconds and, factoring $4q - t^2$, it is nearly instantaneous to retrieve $D_K = -7$ and

$$v = 2 \cdot 127 \cdot \underbrace{524287}_{p_1} \cdot \underbrace{7195777666870732918103}_{p_2}.$$

Let us apply Algorithm 1 to compute the conductor u of $\mathcal{O}(E)$. First, we use isogeny climbing to handle small prime factors p of v, those for which Φ_p can be computed quickly (or has already been precomputed); here, this means 2 and 127. It takes about two seconds to compute Φ_{127} and isogeny climbing itself takes less than two seconds. We find that none of these primes divide u; hence $E' = E$ and we may now assume $v = p_1 p_2$.

For p_1 we set $D_1 = (v/p_1)^2 D_K$ and $D_2 = p_1^2 D_K$ as in Corollary 4. To find a relation satisfying this corollary, we use Algorithm FindRelation(D_1, D_2) with the bound $B = 600$. Corollary 7 uses $B = L(q)^{1/\sqrt{12}} \approx 1900$, but, taking into account constant factors in the complexity estimates, we find experimentally that $B = 600$ better balances the running time of FindRelation with the time to compute Φ_ℓ for $\ell \approx B$. The iteration bound $2m = 6 \cdot 10^7$ is obtained from $m = 1/\rho(z)$, using Table 1 of [17] computed by Bernstein, where $z = \log n / \log B \approx 8$.

After about four minutes, FindRelation outputs the relation R with

$$\left(\ell_i^{e_i}\right) = \left(2^{1798}, 23^3, 29^1, 37^2, 53^{29}, 137^1, 149^1, 233^1, 263^2, 547^1\right),$$

for which $\#R/D_1 = 2$ and $\#R/D_2 = 0$. Note that, as suggested by Footnote 3, we make use of $\ell = 2$ even though it divides v (using the algorithm in [36, Section 4.2]). Now, to evaluate $\#R/D_E$ using Algorithm CountRelation(E, R), we need to compute the required modular polynomials. We use precomputed Φ_ℓ for $\ell < 100$, and for $\ell \geqslant 100$ apply the algorithm in [7]; this takes about four minutes, three of which are spent on Φ_{547}. Finally, $\#R/D_E = 0$ is evaluated in about a minute. Since $\#R/D_E < \#R/D_1$, we conclude from Corollary 4 that p_1 is a factor of u.

We now turn to p_2 and set $D_1 = (v/p_2)^2 D_K$ and $D_2 = p_2^2 D_K$ accordingly. The relation $R = (2^{23}, 11^5, 43^1, 71^2)$ is found by FindRelation(D_1, D_2), and we have $\#R/D_1 = 2$ and $\#R/D_2 = 0$. We then use CountRelation(E, R) to compute $\#R/D_E = 2$, proving that $p_2 \nmid u$ (since $\#R/D_E \not< \#R/D_1$). The processing of p_2 takes very little time, a few seconds at most.

All in all, we have found the conductor $u = p_1$ of the elliptic curve E defined over a 200-bit prime field in less than ten minutes of computation. The sizes of the primes p_1 and p_2 represent nearly a worst-case: if p_2 was five or six bits larger the remaining part of v would be small enough that one could more efficiently use a combination of isogeny climbing and Hilbert class polynomials to determine u.

We note that, in this example, we could have used the modular function $\gamma_2 = j^{1/3}$ (or other even more favorable functions [7,13]) in place of j, allowing us to use modular polynomials that can be computed much more quickly than Φ_ℓ. Doing so would let us increase the bound B (reducing the time to find relations), and lead to an overall improvement in the running time.

5.2. Second example

Consider now the elliptic curve $E : Y^2 = X^3 - 3X + c_E$ defined over the 255-bit prime field $\mathbb{F}_q$ where

$$\begin{aligned} c_E = {} & 142629578957837647429875247328211995 70\backslash \\ & 86024329300773553757502705145366349 4306 \quad \text{and} \end{aligned}$$

$$\begin{aligned} q = {} & 50272551883931021408091448710235646749\backslash \\ & 90466098049857668008669986543184356 8847. \end{aligned}$$

As above, we compute its trace $t = 1200$ via the SEA algorithm in about 10 seconds, and an easy factorization yields $D_K = -7$ and

$$v = 2 \cdot 127 \cdot \underbrace{582509}_{p_1} \cdot \underbrace{582511}_{p_2} \cdot \underbrace{852857}_{p_3} \cdot \underbrace{2305843009213693951}_{p_4}.$$

Let us run Algorithm 2 to compute $\mathcal{O}(E)$. We start with $w = 1/3$, and first remove the prime factors of v less than $\exp(\log^{1/3} v) \approx 85$. As in Example 1, the constant factors make this a slight underestimate, and we are happy to increase this bound to include both 2 and 127, which we handle by isogeny climbing. We find that neither of these divide u, and therefore $E' = E$, so we now assume that $v = p_1 p_2 p_3 p_4$.

We then reach Step 6 and consider divisors u of v less than $\exp(\log^{2w} v) \approx 4 \cdot 10^8$, namely p_1, p_2, and p_3. Starting with $u \leftarrow p_1$, the certificate C generated in Step 7 by $\textsc{Certify}(u, v, D_K)$ consists of

$$R_{p_4} = R_{p_3} = R_{p_2} = \left(2^{41}, 11^{31}, 37^1\right) \quad \text{and} \quad R_{p_1} = \left(11^1\right),$$

and takes negligible time to compute. The call to $\textsc{Verify}(E/\mathbb{F}_q, C)$ takes less than one second and returns `false`, proving that $u \neq p_1$.

Turning to $u = p_2$, $\textsc{Certify}(u, v, D_K)$ quickly outputs the certificate

$$R_{p_4} = R_{p_3} = R_{p_1} = \left(2^{85}, 11^2, 23^5, 29^3\right) \quad \text{and} \quad R_{p_2} = \left(11^1\right),$$

and $\textsc{Verify}(E/\mathbb{F}_q, C)$ returns `false` (again taking less than a second); so $u \neq p_2$.

We next consider $u = p_3$; the certificate used is

$$R_{p_4} = R_{p_2} = R_{p_1} = \left(2^{239}, 11^1, 37^3\right) \quad \text{and} \quad R_{p_3} = \left(11^1\right).$$

Computing and verifying this certificate takes about a second, and in this case the verification succeeds, proving that $u = p_3$.

The total running time is less than fifteen seconds, most of which is spent point-counting. For comparison, our implementation of Algorithm 1 takes the better part of a day when run on this example. This can be reduced to less than four hours using alternative modular polynomials, but Algorithm 2 is still much faster in this situation.

Acknowledgments

The authors express their gratitude to Pierrick Gaudry, Takakazu Satoh, Daniel J. Bernstein, and Tanja Lange for their careful reading and helpful comments on an early version of this paper; they also thank the anonymous referee for suggesting various improvements.

References

[1] Manindra Agrawal, Neeraj Kayal, Nitin Saxena, Primes is in P, Ann. of Math. 160 (2004) 781–793.
[2] A.O.L. Atkin, François Morain, Elliptic curves and primality proving, Math. Comp. 61 (1993) 29–68.
[3] Eric Bach, Analytic methods in the analysis and design of number-theoretic algorithms (ACM Distinguished Dissertation), MIT Press, 1985.
[4] Daniel J. Bernstein, How to find smooth parts of integers, http://cr.yp.to/papers#smoothparts, 2004.
[5] Ingrid Biehl, Johannes Buchmann, An analysis of the reduction algorithms for binary quadratic forms, in: Peter Engel, Halyna M. Syta (Eds.), Voronoi's Impact on Modern Science, Institute of Mathematics, Kiev, 1998, pp. 71–98, available at http://www.cdc.informatik.tu-darmstadt.de/reports/TR/TI-97-26.ps.gz.
[6] Reinier Bröker, A p-adic algorithm to compute the Hilbert class polynomial, Math. Comp. 77 (2008) 2417–2435.
[7] Reinier Bröker, Kristin Lauter, Andrew V. Sutherland, Modular polynomials via isogeny volcanoes, 2009, in preparation.
[8] Johannes Buchmann, Ulrich Vollmer, Binary Quadratic Forms: an Algorithmic Approach, Springer, 2007.
[9] Joseph P. Buhler, Hendrik W. Lenstra Jr., Carl Pomerance, Factoring integers with the number field sieve, in: Arjen K. Lenstra, Hendrik W. Lenstra Jr. (Eds.), The Development of the Number Field Sieve, in: Lecture Notes in Math., vol. 1554, Springer, 1993.
[10] Paula Cohen, On the coefficients of the transformation polynomials for the elliptic modular function, Math. Proc. Cambridge Philos. Soc. 95 (1984) 389–402.
[11] David A. Cox, Primes of the Form $x^2 + ny^2$, John Wiley & Sons, 1989.
[12] Richard Crandall, Carl Pomerance, Prime Numbers: A Computational Perspective, second ed., Springer, 2005.
[13] Andreas Enge, Computing modular polynomials in quasi-linear time, Math. Comp. 78 (2009) 1809–1824.
[14] Mireille Fouquet, François Morain, Isogeny volcanoes and the SEA algorithm, in: Claus Fieker, David R. Kohel (Eds.), Algorithmic Number Theory Symposium – ANTS, V, in: Lecture Notes in Comput. Sci., vol. 2369, Springer, 2002, pp. 276–291.
[15] Steven D. Galbraith, Florian Hess, Nigel P. Smart, Extending the GHS Weil descent attack, in: Advances in Cryptology – EUROCRYPT 2002, in: Lecture Notes in Comput. Sci., vol. 2332, Springer, 2002, pp. 29–44.
[16] Torbjörn Granlund, et al., GNU multiple precision arithmetic library, version 4.3.0, available at http://gmplib.org/, 2009.
[17] Andrew Granville, Smooth numbers: computational number theory and beyond, in: Joseph P. Buhler, Peter Stevenhagen (Eds.), Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography, in: Math. Sci. Res. Inst. Publ., vol. 44, 2008, pp. 267–323.
[18] James L. Hafner, Kevin S. McCurley, A rigorous subexponential algorithm for computing in class groups, J. Amer. Math. Soc. 2 (4) (1989) 837–850.
[19] Godfrey H. Hardy, Edward M. Wright, An Introduction to the Theory of Numbers, fifth ed., Oxford Science Publications, 1979.
[20] Adolf Hildebrand, On the number of positive integers $\leqslant x$ and free of prime factors $> y$, J. Number Theory 22 (3) (1986) 289–307.
[21] Dale Husemöller, Elliptic Curves, Springer, 1987.
[22] David Jao, Stephen D. Miller, Ramarathnam Venkatesan, Do all elliptic curves of the same order have the same difficulty of discrete log? in: Advances in Cryptology – ASIACRYPT 2005, in: Lecture Notes in Comput. Sci., vol. 3788, Springer, 2005, pp. 21–40.
[23] David Jao, Stephen D. Miller, Ramarathnam Venkatesan, Expander graphs based on GRH with an application to elliptic curve cryptography, J. Number Theory 129 (6) (2009) 1491–1504.
[24] David Kohel, Endomorphism rings of elliptic curves over finite fields, PhD thesis of the University of California at Berkeley, 1996.
[25] Jeffrey C. Lagarias, Andrew M. Odlyzko, Effective versions of the Chebotarev density theorem, in: Algebraic Number Fields: L-Functions and Galois Properties, Proceedings of a Symposium held at the University of Durham, 1975, Academic Press, 1977, pp. 409–464.
[26] Serge Lang, Elliptic Functions, second ed., Springer, 1987.
[27] Hendrik W. Lenstra Jr., Factoring integers with elliptic curves, Ann. of Math. 126 (1987) 649–673.
[28] Hendrik W. Lenstra Jr., Carl Pomerance, A rigorous time bound for factoring integers, J. Amer. Math. Soc. 5 (3) (1992) 483–516.
[29] Kevin S. McCurley, Cryptographic key distribution and computation in class groups, in: R.A. Mollin (Ed.), NATO ASI on Number Theory and Applications, C, vol. 265, Kluwer, 1989, pp. 459–479.
[30] Arnold Schönhage, Fast reduction and composition of binary quadratic forms, in: Stephen M. Watt (Ed.), International Symposium on Symbolic and Algebraic Computation – ISSAC'91, ACM Press, 1991, pp. 128–133.
[31] René Schoof, Counting points on elliptic curves over finite fields, J. Theor. Nombres Bordeaux 7 (1995) 219–254.
[32] Jean-Pierre Serre, Complex multiplication, in: John W.S. Cassels, Albrecht Frölich (Eds.), Algebraic Number Theory, Academic Press, 1967.
[33] Victor Shoup, NTL: a library for doing number theory, version 5.5.1, available at http://www.shoup.net/, 2009.

[34] Joseph H. Silverman, Advanced Topics in the Arithmetic of Elliptic Curves, Springer, 1999.
[35] Richard Stallman, et al., GNU compiler collection, version 4.2.4, available at http://gcc.gnu.org/, 2008.
[36] Andrew V. Sutherland, Computing Hilbert class polynomials with the Chinese Remainder Theorem, Math. Comp. (2009), in press; http://arxiv.org/abs/0903.2785.
[37] Lawrence C. Washington, Elliptic Curves: Number Theory and Cryptography, second ed., CRC Press, 2008.