A New Special-Purpose Factorization Algorithm

Qi Cheng[*]

Abstract

In this paper, a new factorization algorithm is presented, which finds a prime factor p of an integer n in time $(D\log n)^{O(1)}$, if $4p-1 = Db^2$ where D and b are integers. Hence this algorithm will factor a number efficiently, if it has a prime factor p such that $4p-1$ is a product of a small integer and a square. Such primes should be avoided when we select the RSA secret keys. Some generalizations of the algorithm are discussed in the paper as well.

Classification of Topics: Cryptography, Integer factorization.

1 Introduction

Integer factorization is a classical problem in computer science and number theory. It has been studied for centuries and been intensively investigated in the last four decades. Although remarkable progresses have been achieved, especially in the last thirty years, this problem is still considered difficult. Several cryptographic systems based on the hardness of factorization or analogical problems have been proposed. Among them, the RSA system is the most famous and widely used. So far, the fastest general-purpose factorization algorithm is the number field sieve (NFS), which has a heuristic time complexity $O(e^{c(\log n)^{1/3}(\log\log n)^{2/3}})$ to factor an integer n, where $c \approx 1.923$. We refer to [3] for a survey on the current knowledge about factoring general integers.

Other than the general-purpose factorization algorithms, some algorithms are very efficient at finding a prime factor of special form, even though, the performance of those algorithms is sometimes worse than that of the exhaustive search if we try to apply them on general integers. Those algorithms include:

1. Pollard's $p-1$ method [16] finds a prime factor p efficiently if $p-1$ is smooth. More precisely, if the largest prime factor of $p-1$ is r, then it takes time $(r\log n)^{O(1)}$ for the algorithm to find p.

2. Hugh Williams's $p+1$ method [19] works well when $p+1$ is smooth.

[*]School of Computer Science, the University of Oklahoma, Norman, OK 73019. Email: qcheng@cs.ou.edu.

3. The Bach-Shallit cyclotomic polynomial method [1] extends the ideas in the $p \pm 1$ algorithms. It finds a prime factor p of n efficiently if $\phi_k(p)$ is smooth, where ϕ_k is the k-th cyclotomic polynomial. This algorithm provides a unified presentation of a class of factorization algorithms, including the $p \pm 1$ methods. But its practical application is limited because when $k > 2$, $\phi_k(p)$ is much bigger than p, hence unlikely to be smooth.

4. Other than integers with special form prime factors, integers with certain prime power can also be efficiently factored. For example, Boneh and etc. [2] proposed an algorithm, which factors $n = p^r q$ in polynomial time if p and q are primes and r is close to $\log p$.

To implement RSA cryptosystem, two large primes need to be selected and kept secret. The product of these two primes is made public. The security of this cryptosystem is destroyed if the adversary can factor the product. In order to avoid the $p-1$ factorization, we should make sure that $p-1$ contains at least one large prime factor, or better yet, $p-1 = 2q$ with q a prime. Traditionaly, a prime p is called *safe*, if $\frac{p-1}{2}$ is also a prime.

We call n a RSA integer, if it is the product of two different primes. Given a prime p, if any of the $p-1$, $p+1$ or $\phi_k(p)$ (k is small) is smooth, then a RSA integer with p as its prime factor can be factored efficiently. These primes are unsafe and should be avoided when we select RSA secret primes. In this paper, we report a new factorization algorithm and a new class of unsafe primes. Our main result is

Theorem 1 *Let integer $n = pm$ with p a prime and m an integer. There exists a random algorithm finding p from n in time $(D \log n)^{O(1)}$ if p has form $(Db^2+1)/4$ with b and D integers.*

Note that it must hold that $D \equiv 3 \pmod 8$. The algorithm is called $4p-1$ method in this paper.

Remark 1 If a prime factor of n is known to have special form $\frac{1+Db^2}{4}$, then factorization of n amounts to finding the integer solutions of the multivariate equation:

$$(1 + Dx^2)y - 4n = 0.$$

In his seminal paper [7], Coppersmith proposed a lattice reduction technique to solve certain kinds of integral multivariate equations. However, it can be verified that his algorithm does not work here.

2 Overview of the algorithm

Our algorithm can be viewed as a variant of the elliptic curve factorization algorithm invented by Lenstra [13]. Let $R = \mathbf{Z}/n\mathbf{Z}$. In his algorithm, a random elliptic curve E/R with a point P on that curve is chosen. A large smooth number k is computed. Since the smooth bound B is usually set to be subexponential, computing k alone takes subexponential time. The order of $E(\mathbf{F}_p)$ for some $p|n$ is B-smooth with subexponential probability. In this case, computing kP usually reveals p. The idea in the algorithm originates from the $p-1$ method. As in the $p-1$ method, smoothness plays an important role in Lenstra's algorithm. But the latter is a general-purpose factorization algorithm as oppose to the $p-1$ method.

In our algorithm, we fix the set of elliptic curves and use n *itself* instead of a large smooth integer k as the multiplier. Our algorithm outputs a prime factor p of n, if $E(\mathbf{F}_p)$ has order exactly p. Since given an arbitrary elliptic curve, it is usually difficult to find a point on the curve modulo a composite number, it is important that we find a way to avoid working with points explicitly. Instead of computing a product of n and a point, we evaluate the n-th division polynomial on a randomly chosen integer x, which we hope is an x-coordinate of an $\mathbf{F}_p$-point on the E. A random integer becomes such an integer with probability about 1/2, which is an easy consequence of Hasse's Lemma. Computing the g.c.d. of n and the value of the division polynomial modulo n gives us the factorization of n.

We first consider the case when the set of elliptic curves is the rational elliptic curves with complex multiplications. For any curve E in this set, the primes p such that $|E(\mathbf{F}_p)| = p$ can be described. They include every prime p such that $4p-1$ is a product of D and a square, where $D \in \{3, 11, 19, 43, 67, 163\}$. We then extend ideas to work on the general D. The idea is to use elliptic curve with j-invariant

$$j = X \pmod{H_D(X), n},$$

where $H_D(X)$ is the Hilbert class polynomial for the field with discriminant $-D$. A RSA integer with prime factor of one of these forms can be factored efficiently by our algorithm if D is small.

We can consider using the rational elliptic curves with small j-invariants and the hyperelliptic curves with small genus g as well. A hyperelliptic curve with Jacobian group of order p^g over $\mathbf{F}_p$ can be used to factor any integer with prime factor p. Several interesting questions in number theory are raised: (1) Given a prime p, what are the (hyper)elliptic curves with Jacobian group of order $p(p^g)$ over $\mathbf{F}_p$? The question can help us pick good primes free of $4p-1$ attack. (2) Given a curve C/Q with genus g, how many primes p are there such that the reduction of C at p has Jacobian group of order p^g over $\mathbf{F}_p$? In the elliptic curve case, the problem has been studied. We will review some results in this paper. However, we are not aware of any results on the similar question about hyperelliptic curves.

The novelties of our algorithm includes (1) We use n as the multiplier. Using integers closely related to n is another possibility. (2) We avoid finding a point on the curve. This is very important since we need to work with a curve and its quadratic twist. Finding points on both curves is usually a very difficult problem. If one is satisfied with random polynomial time, then it is not necessary to know the y-coordinate of a point in order to factor an integer. We can evaluate the n-th division polynomial on a random integer. (3) Although our algorithm is derived from the elliptic curve factorization algorithm, it factors numbers with special form prime factors in polynomial time, without assuming any number theory conjecture. The time complexity of the algorithm doesn't rely on the abundance of smooth numbers, which is quite different from the classical factorization algorithm.

2.1 Comparison of $p-1$ method and $4p-1$ method

How many primes are vulnerable to $4p-1$ attack? For simplicity, we consider using rational elliptic curves. Given a prime p, the number of $\mathbf{F}_p$-points is a random integer (almost) uniformly distributed between $p+1-2\sqrt{q}$ and $p+1+2\sqrt{p}$ for a random elliptic curve $\mathcal{E}/\mathbf{Q}$. Hence heuristically, given an elliptic curve $\mathcal{E}/\mathbf{Q}$, for a random prime p, $|\mathcal{E}_p(\mathbf{F}_p)| = p$ happens with probability $O(1/\sqrt{p})$, where $\mathcal{E}_p$ is the reduction of $\mathcal{E}$ at p. Let $\pi_{\mathcal{E}}(x)$ denote the number of primes p less than x such that $|\mathcal{E}_p(\mathbf{F}_p)| = p$ for the elliptic curve $\mathcal{E}/\mathbf{Q}$. The above heuristic gives us

$$\pi_{\mathcal{E}}(x) = O(\frac{x}{\log x}\frac{1}{\sqrt{x}}) = O(\frac{\sqrt{x}}{\log x}).$$

In fact, it was conjectured by Lang and Trotter [11] that $\pi_{\mathcal{E}}(x) \approx \frac{c\sqrt{x}}{\log x}$. Note that c could be 0, for example when $\mathcal{E}$ has non-trivial torsions.

This problem has been studied by Serre [17]. Assuming GRH, the upper bound of $x^{4/5}(\log x)^{-1/5}$ has been proved by Murty etc. [15]. They also showed that the curve tends to have the number of points far away from the median $p+1$ as p varies. Hence the RSA integers which can be efficiently factored by our algorithm are rare. However, some cautions need to be taken when we design RSA system, especially when we generate special form RSA moduli [12]. Note that for a fixed small D, The most time-consuming part of $4p-1$ method is to evaluate the n-th division polynomial modulo n, whose time complexity is roughly equal to computing a multiplication of a point by the number n.

The $p-1$ method works if $p-1$ is a smooth number. We say an integer is l-smooth, if all its prime factors are less than l. If we choose the smooth bound to be $l = (\log n)^c$, $c > 1$, then there are about $n^{1-1/c}$ l-smooth number less than n [8]. Again using the heuristical argument, we can see that about $\frac{n^{1-1/c}}{\log n}$ primes are vulnerable to $p-1$ attack. In order to make $p-1$ method competitive to $4p-1$ method, we have to choose $c > 2$. The time complexity of the attack is equivalent to computing s-power of a integer modulo n, where s is about $(\sqrt{n})^{\log^c n}$.

Hence in limited time, $4p-1$ method will factor more numbers than $p-1$ method does. When we increase the time limitation, then $p-1$ will outperform the $4p-1$ method.

Lenstra's algorithm also provides a set of easily-factored integers, namely, those which contain the primes p such that the number of $\mathbf{F}_p$-points on a pre-fixed elliptic curve is $\log^c n$-smooth. The situation is similar to the $p-1$ method.

3 Elliptic curves

An elliptic curve is a smooth cubic curve. Let k be a field. If the characteristic of k is neither 2 nor 3, we may assume that the elliptic curve is given by an equation of the form

$$y^2 = x^3 + ax + b, \qquad a, b \in k.$$

The discriminant of this curve is defined as $\Delta = -16(4a^3 + 27b^2)$, which is non-zero as the curve is smooth. For detailed information about elliptic curves, we refer to Silverman's book [18].

The j-invariant of the curve $y^2 = x^3 + ax + b$ is defined as $j = 1728\frac{4a^3}{4a^3+27b^2}$. Two elliptic curves with a same j-invariant are isomorphic over the algebraic closed field. For elliptic curves defined over a prime finite field $\mathbf{F}_p$ with $p > 3$, two curves with a same j-invariant may not be isomorphic. If $j \neq 0$ or 1728, there are exactly two isomorphic classes which have the same j-invariant, one can be represented by $E_1 : y^2 = x^3 + kx + k$ and the other by $E_c : y^2 = x^3 + c^2kx + c^3k$, where $k = \frac{27j}{4(1728-j)}$ and c is a quadratic nonresidue modulo p. The latter curve E_c is called the quadratic twist of the former one. It is not hard to see that $|E_1(\mathbf{F}_p)| + |E_c(\mathbf{F}_p)| = 2p + 2$. There are at most 6 isomorphic classes with $j = 0$, and at most 4 isomorphic classes with $j = 1728$.

The set of points on an elliptic curve consists of the solution set of the definition equation plus a point at infinity. These points form an abelian group with the infinity point as the identity. We call a point *torsion* if it has a finite order in the group. The x-coordinates of the torsions of order $n > 3$ are the solutions of $P_n(x)$, the n-th division polynomial of $\mathcal{E}$. The $P_n(x)$ can be evaluated using only $O(\log n)$ arithmetic operations (additions, subtractions and multiplications) from a, b and x, just like that nP can be computed using only $O(\log n)$ point additions. The observation is implicitly stated in several places, we refer to [5] for the formal proof (of a stronger version).

Proposition 1 *For any integer $n(> 0)$, $P_n(x)$ can be computed by $O(\log n)$ ring operations from a, b and x, where P_n is the n-th division polynomial of $\mathcal{E} : y^2 = x^3 + ax + b$.*

Assume that $a, b \in \mathbf{Z}$. Even when n is very large, we can still carry out the computation of $P_n(x)$ if we do every operation modulo an integer m. The result can be used to factor m. The prime factors of $P_n(x)$ forms a subset of all the primes such that the reduction curves at those primes have order dividing n over the prime finite field. The next proposition follows easily from the definition of torsion points.

Proposition 2 *Let $\mathcal{E} : y^2 = x^3 + ax + b$ be an elliptic curve defined over $\mathbf{Z}$. Assume that $\mathcal{E}$ has a good reduction E at a prime p. If x is an integer and*

1. *it is the x-coordinate of a point on $E(\mathbf{F}_p)$,*

2. *the point $(x, \sqrt{x^3 + ax + b})$ is not a torsion on $\mathcal{E}$,*

then $P_l(x) \neq 0$ and $p|P_l(x)$, where l is any non-zero multiple of $|E(\mathbf{F}_p)|$.

The number of torsions is very small [9, 10]. A random integer x has the properties described in the above proposition with probability about 1/2.

D	j_D	The form of p
3	0	$4p-1=3b^2$
11	$(-2^5)^3$	$4p-1=11b^2$
19	$(-2^5*3)^3$	$4p-1=19b^2$
43	$(-2^5*3*5)^3$	$4p-1=43b^2$
67	$(-2^5*3*5*11)^3$	$4p-1=67b^2$
163	$(-2^6*3*5*23*29)^3$	$4p-1=163b^2$

Table 1: The primes of special forms

4 Proof of the main theorem

Let p be a prime greater than 3. A non-supersingular elliptic curve $E/\mathbf{F}_p$ has a complex multiplication by an order of a quadratic field $K = \mathbf{Q}(\sqrt{-D})$. We are interested in the curves which have exactly p $\mathbf{F}_p$-points. Similar problem has been studied in [14]. If $|E(\mathbf{F}_p)| = p$, then its quadratic twist has $p+2$ $\mathbf{F}_p$-points. First we consider the curves defined over $\mathbf{Q}$. See Table 1 for the list of integers D, the corresponding j-invariants of the curves whose complex multiplications are the maximal order in $\mathbf{Q}(\sqrt{-D})$, and the forms of the primes p such that at least one of the isomorphic classes of the curves has exactly p $\mathbf{F}_p$-points.

If p has one of the special forms in Table 1, we can easily construct an elliptic curve $E/\mathbf{F}_p$ with exactly p $\mathbf{F}_p$-points. See [14] for the algorithm to decide the right isomorphic classes. When it comes to the factorization, p is unknown. It is impossible to check whether an integer is a quadratic residue modulo p or not. Fortunately the j-invariants of the curves do not depend on p, and one half of the integers are quadratic residues modulo p, the other half are quadratic non-residues modulo p. Hence we can still construct the right curves with probability about 1/2.

Now we study the case when D is not in the table 1. Suppose n contains a prime factor p and $4p-1$ is a product of D and a square. The Hilbert polynomial $H_D(x)$ is the minimum polynomial for the j-invariant of the elliptic curve whose endomorphism ring is the maximal order of $\mathbf{Q}(\sqrt{-D})$. It can be computed in time $D^{O(1)}$ [6, page 415]. We can use the curve with j-invariant $j = X \pmod{H_D(X), n}$. (For better time complexity, we may use Weber polynomials and compute j by simple algebraic operations.) The $P_n(x)$ can still be computed for any random integer x. Let $g(X) = P_n(x) \in \mathbf{Z}/(n)[X]$. When modulo p, $g(X)$ has a common root with $H_D(X)$ with probability around 1/2 for random c. If q is another prime factor of n, it is almost certain that $gcd(H_D(X) \pmod{q}, g(X) \pmod{q}) = 1$. We can factor n efficiently according to the following lemma.

Lemma 1 *Given an integer n and two monic polynomial $f(x), g(x) \in \mathbf{Z}/(n)[x]$ with maximum degree d. If n has two prime factors p and q, and*

1. *$gcd(f(x) \pmod{p}, g(x) \pmod{p}) \neq 1$;*

2. *$gcd(f(x) \pmod{q}, g(x) \pmod{q}) = 1$,*

then n can be factored in time $(d \log n)^{O(1)}$.

Proof: Apply Euclidean algorithm on $f(x), g(x)$. During the execution of the algorithm, if we find a zero-divisor, n is factored as a consequence. Now assume that the algorithm is completed. The output should be a constant $a \in \mathbf{Z}/(n)$ since $gcd(f(x) \pmod{q}, g(x) \pmod{q}) = 1$. In this case $p|gcd(n,a)$ and $q \not| gcd(n,a)$. □

5 Algorithm description and example

We now describe the algorithm. In the following algorithm description, we assume D is known. There are a little difference between $D = 3$ and $D \neq 3$, so we treat them separately. First we consider the case when $D \neq 3$. In the following algorithm, it suffices to set $B_1 = B_2 = 10$.

```
compute $H_D(X)$;
let $j = X \pmod{H_D(X), n}$;
compute $a(X) = \frac{j}{1728-j}$;
randomly select $B_1$ integers $c_1, c_2, \cdots, c_{B_1}$;
randomly select $B_2$ integers $x_1, x_2, \cdots, x_{B_2}$;
for each $c \in \{c_1, c_2, \cdots, c_{B_1}\}$
    for each $x \in \{x_1, x_2, \cdots, x_{B_2}\}$
        compute $z(X) = P_n(x)$ where $P_n$ is the $n$-th division
        polynomial of the ellipitic curve $y^2 = x^3 + 3a(X)c^2x + 2a(X)c^3$;
        compute $\alpha = gcd(z(X), H_D(X) \pmod{n})$;
        if the Euclidean algorithm can not process, a zero-divisor in
        $\mathbf{Z}/(n)$ must been found, output the factor of $n$ and exit;
        if $gcd(\alpha, n)$ is non-trivial, output the results and exit;
    endfor
endfor
```

This algorithm factors the following 98-digit number in the matter of seconds on a 1GHz PC.

```
n = 2673244416506417435728194307912316746792104124799
    7589999975598149848254595650256312445340591487269
```

The Hilbert polynomial at discriminant -35 is

$$H_{35}(X) = X^2 + 117964800X - 134217728000.$$

Let $\mathcal{E}$ be the elliptic curve with $j = X \pmod{n, X^2 + 117964800X - 134217728000}$ and $c = 1$ and $P_n(x)$ be its n-th division polynomial. Evaluating $P_n(2)$ gives us $z(X)$

```
9835574879806685785089618088376686675536348034816708
114592982090289489337979148083794880998750685*X
+ 17884831546461983409826598629611855714964241177958
06465489635634750028292682696385406645184555639.
```

Computing $gcd(z(X), X^2 + 117964800X - 134217728000 \pmod{n})$ yields

```
2655409412619398519588238788594643694821968607782
2481810088900062744578960275375216911557353776836
```

which contains the prime factor of n

```
p = 1394116698586249968612479056968729556521399688429.
```

Indeed, $4p - 1 = 35 \times 399158643518553190536057^2$. The other factor of n is

```
q = 19175183965713265819619376872762949381791719783961.
```

Note that $p \pm 1$ methods will not factor n in reasonable time, since the prime factorizations of $p \pm 1$ and $q \pm 1$ are

$$\begin{aligned}
p - 1 &= 2^2 * 3 * 223283 * 520310061889414617552791396631453341171443 \\
p + 1 &= 2 * 5 * 3596009 * 177737796426323039 * 218121546208842242073293 \\
q - 1 &= 2^3 * 5 * 1423 * 336879549643592161272301069444183931514260713 \\
q + 1 &= 2 * 3 * 17 * 173 * 1254682349 * 866082924064902690008192697814996903
\end{aligned}$$

None of the general-purpose factorization algorithm can factor n without hours of computation on a single 1GHz PC.

When $D = 3$, we should use the curve with $j = 0$, namely, $y^2 = x^3 + a$. There are at most six isomorphic classes, depending on the sixth power residue classes that a belongs to. If randomly choose a, then with probability 1/6, we will have the right curve E with $|E(\mathbf{F}_p)| = p$. The algorithm in this case is as follows. We can set $B_1 = 20$.

```
randomly select $B_1$ integers $a_1, a_2, \cdots, a_{B_1}$;
randomly select $B_2$ integers $x_1, x_2, \cdots, x_{B_2}$;
for each $a \in \{a_1, a_2, \cdots, a_{B_1}\}$
    for each $x \in \{x_1, x_2, \cdots, x_{B_2}\}$
        compute $z = P_n(x) \pmod{n}$ where $P_n$ is the $n$-th division
        polynomial of elliptic curve $y^2 = x^3 + a$;
        compute $gcd(z, n)$;
        if the gcd is non-trivial, output the result and exit;
    endfor
endfor
```

We can certainly use other rational elliptic curves without complex multiplications.

```
for $j$ from $-B_3$ to $B_3$
    compute $a = \frac{j}{1728-j} \pmod{n}$;
    randomly select $B_1$ integers $c_1, c_2, \cdots, c_{B_1}$;
    randomly select $B_2$ integers $x_1, x_2, \cdots, x_{B_2}$;
    for each $c \in \{c_1, c_2, \cdots, c_{B_1}\}$
        for each $x \in \{x_1, x_2, \cdots, x_{B_2}\}$
            compute $z = P_n(x) \pmod{n}$ where $P_n$ is the $n$-th division polynomial of the elliptic curve $y^2 = x^3 + 3ac^2x + 2ac^3$;
            compute $gcd(z, n)$;
            if the gcd is non-trivial, output the result and exit;
        endfor
    endfor
endfor
```

In the algorithm, the bound B_3 may be set accordingly. The time complexity is $B_3(\log n)^{O(1)}$

6 Conclusion and open problems

We present a new special-purpose factorization algorithm, which splits n in time $(D \log n)^{O(1)}$, if it has a prime factor of form $(Db^2 + 1)/4$. As in the elliptic curve factorization algorithm, this method relies on the fact that the order of an elliptic curve group over $\mathbf{F}_p$ is uniformly distributed between $p + 1 - 2\sqrt{p}$ and $p + 1 + 2\sqrt{p}$, hence could be p. If we use the multiplicative group of finite field, we can not obtain such an algorithm.

From the past experiences, we know the algorithms of factoring integers and solving the discrete logarithm over finite fields are usually coupled with each other. For example, when $p - 1$ is smooth, the discrete logarithm over $\mathbf{F}_p$ admits efficient algorithm too. It is interesting to see whether the discrete logarithm problem on $\mathbf{F}_p$ with p of the special forms has polynomial time algorithm or not. It is well-known that the discrete logarithm problem on $E/\mathbf{F}_p$ where $|E(\mathbf{F}_p)| = p$ can be efficiently solved.

References

[1] Eric Bach and Jeffrey Shallit. Factoring with cyclotomic polynomials. *Math. Comp.*, 52(185):201–219, 1989.

[2] Dan Boneh, Glenn Durfee, and Nick Howgrave-Graham. Advances in cryptology. In *Proc. of Crypto'99*, volume 1666 of *Lecture Notes in Computer Science*, 1999.

[3] Richard P. Brent. Recent progress and prospects for integer factorisation algorithms. In *Proc. of COCOON 2000*, volume 1858 of *Lecture Notes in Computer Science*. Springer-Verlag, 2000.

[4] David G. Cantor. On the analogue of the division polynomials for hyperelliptic curves. *J. Reine Angew. Math.*, 447:91–145, 1994.

[5] Qi Cheng. Some remarks on the l-conjecture. In *Proc. of the 13th Annual International Symposium on Algorithms and Computation(ISAAC)*, Lecture Notes in Computer Science. Springer-Verlag, 2002.

[6] Henri Cohen. *A Course in Computational Algebraic Number Theory*. Springer-Verlag, 1993.

[7] D. Coppersmith. Small solutions to polynomial equations, and low exponent rsa vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997.

[8] Adolf Hildebrand and Gerald Tenenbaum. Integers without large prime factors. *J. Theor. Nombres Bordeaux*, 5(2):411–484, 1993.

[9] S. Kamienny. Torsion points on elliptic curves and q-coefficients of modular forms. *Inventiones Mathematicae*, 109:221–229, 1992.

[10] M. Kenku and F. Momose. Torsion points on elliptic curves defined over quadratic fields. *Nagoya Mathematical Journal*, 109:125–149, 1988.

[11] Serge Lang and Hale Trotter. *Frobenius distributions in* GL_2*-extensions*. Springer-Verlag, Berlin, 1976.

[12] A.K. Lenstra. Generating RSA moduli with a predetermined portion. In *Asiacrypto'98*, volume 1514 of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.

[13] H. W. Lenstra. Factoring integers with elliptic curves. *Annals of Mathematics*, 126:649–673, 1987.

[14] A. Miyaji. Elliptic curves over $\mathbf{F}_p$ suitable for cryptosystems. In *Advances in Cryptology, AUSCRYPT92*, volume 718 of *Lecture Notes in Computer Science*, pages 479–491. Springer-Verlag, 1993.

[15] M. Ram Murty, V.K. Murty, and N. Saradha. Modular forms and the Chebotarev density theorem. *Amer. J. Math.*, 110(2):253–281, 1988.

[16] J.M. Pollard. Theorems on factorization and primality testing. *Proc. Camb. Phil. Soc.*, 76(2):521–528, September 1974.

[17] Jean-Pierre Serre. Quelques applications du théorème de densité de Chebotarev. *Inst. Hautes Études Sci. Publ. Math.*, (54):323–401, 1981.

[18] J.H. Silverman. *The arithmetic of elliptic curves*. Springer-Verlag, 1986.

[19] H.C. Williams. A $p+1$ method of factoring. *Mathematics of Computation*, 39(159):225–234, 1982.