International Journal of Computer Mathematics
Vol. 82, No. 12, December 2005, 1481–1495

On the cubic sieve method for computing discrete logarithms over prime fields[†]

ABHIJIT DAS† and C. E. VENI MADHAVAN*‡

†Department of Mathematics, Indian Institute of Technology, Kanpur 208 016, India
‡Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India

(Received 13 December 2002)

In this paper, we report efficient implementations of the linear sieve and the cubic sieve methods for computing discrete logarithms over prime fields. We demonstrate through empirical performance measures that for a special class of primes the cubic sieve method runs about two times faster than the linear sieve method even in cases of small prime fields of the size about 150 bits. We also provide a heuristic estimate of the number of solutions of the congruence $X^3 \equiv Y^2 Z \pmod p$ that is of central importance in the cubic sieve method.

Keywords: Cryptography; Cryptanalysis; Finite field; Discrete logarithm; Sieve method

C.R. Categories: F.2.2

1. Introduction

The security of many cryptographic protocols is based on the apparent intractability of the discrete logarithm problem over prime fields. Since no conclusive complexity-theoretic results concerning the real difficulty of this problem are available, practical performance measures of the algorithms known for solving this problem appear to be of utmost interest in cryptography.

Computation of discrete logarithms over a finite field $\mathbb{F}_q$ is a difficult problem. No algorithms are known that solve the problem in time polynomially bounded by the size of the field (*i.e.*, $\lg q$).[‡] The index calculus method [1–5] is currently the best known algorithm for this purpose and has a sub-exponential expected running time given by

$$L\langle q,\ \gamma,\ c\rangle = \exp((c + \mathrm{o}(1))(\ln q)^{\gamma}(\ln\ln q)^{1-\gamma})$$

for some positive constant c and for some real number γ, $0 < \gamma < 1$. For practical applications, one typically uses prime fields or fields of characteristic 2. In this paper, we focus on prime fields only.

*Corresponding author. Email: cevm@csa.iisc.ernet.in
†A preliminary version of this paper appeared in ISAAC99 [6].
‡We denote $\log x = \log_{10} x$, $\ln x = \log_e x$ and $\lg x = \log_2 x$.

Let $\mathbb{F}_p$ be a prime field of cardinality p. For an element $a \in \mathbb{F}_p$ we denote by $\bar{a}$ the representative of a in the set $\{0, 1, \ldots, p-1\}$. Let g be a primitive element of $\mathbb{F}_p$ (*i.e.*, a generator of the cyclic multiplicative group $\mathbb{F}_p^*$). Given an element $a \in \mathbb{F}_p^*$, there exists a unique integer $0 \le x \le p-2$ such that $a = g^x$ in $\mathbb{F}_p$. This integer x is called the *discrete logarithm* or *index* of a in $\mathbb{F}_p$ with respect to g and is denoted by $\mathrm{ind}_g(a)$. The determination of x from the knowledge of p, g and a is referred to as the *discrete logarithm problem.* In general, one need not assume g to be a primitive element and is supposed to compute x from a and g, if such an x exists (*i.e.*, if a belongs to the cyclic subgroup of $\mathbb{F}_p^*$ generated by g). In this paper, we always assume for simplicity that g is a primitive element of $\mathbb{F}_p$.

In what follows we denote by $L(p,\ c)$ any quantity that satisfies

$$L(p,c) - L\left\langle p, \frac{1}{2}, c\right\rangle = \exp((c+\mathrm{o}(1))\sqrt{\ln p \ln\ln p}),$$

where c is a positive constant. When p is understood from the context, we write $L[c]$ for $L(p,c)$. In particular, $L[1]$ is denoted simply by L.

The basic index calculus method [4, section 6.6.2] for the computation of discrete logarithms over prime fields and the adaptations of this method take time $L[c]$ for c between 1.5 and 2 and are not useful in practice for prime fields $\mathbb{F}_p$ with $p > 2^{100}$. Coppersmith *et al.* [1] proposed three variants of the index calculus method that run in time $L[1]$ and are practical for $p \le 2^{250}$. A subsequent paper [2] by LaMacchia and Odlyzko reports implementation of two of these three variants, namely the linear sieve method and the Gaussian integer method. They were able to compute discrete logarithms in $\mathbb{F}_p$ with p of length about 200 bits.

The paper [1] also describes a cubic sieve method due to Reyneri for the computation of discrete logarithms over prime fields. The cubic sieve method has a heuristic running time of $L[\sqrt{2\alpha}]$ for some α, $1/3 \le \alpha < 1/2$, and is, therefore, asymptotically faster than the linear sieve method (and the other $L[1]$ methods described in ref. [1]). However, the authors of ref. [1] conjectured that the theoretical asymptotics do not appear to take over for p in the range of practical interest (a few hundred bits). A second problem associated with the cubic sieve method is that it requires a solution of the Diophantine equation

$$X^3 \equiv Y^2 Z \pmod{p}, \quad X^3 \ne Y^2 Z, \quad X,\ Y,\ Z = \mathrm{O}(p^{\alpha}) \ \text{ for some } \alpha,\ \frac{1}{3} \le \alpha < \frac{1}{2}.$$

It is not known how to find a solution of this Diophantine equation in the general case. Forcertain special primes p a solution arises naturally, for example, when p is *close* to a perfect cube.

Recently, a new variant of the index calculus method based on number field sieves (NFS) has been proposed [7] and has a conjectured heuristic running time of

$$L\left\langle p,\ \frac{1}{3},\ c\right\rangle = \exp((c+\mathrm{o}(1))(\ln p)^{1/3}(\ln\ln p)^{2/3}).$$

Weber *et al.* [8–10] have implemented and proved the practicality of this method. Currently the NFS-based methods are known to be the fastest algorithms for solving the discrete logarithm problem over prime fields.

In this paper we report efficient implementations of the linear sieve and the cubic sieve methods. To the best of our knowledge ours is the first large-scale implementation of the cubic sieve method. In our implementation we employ ideas similar to those used in the quadratic sieve method for integer factorization [11–13]. Our experiments seem to reveal that the equation collection phase of the cubic sieve method, whenever applicable, runs faster than that in the linear sieve method. We also propose a heuristic modification of the cubic sieve method, that allows us to build a larger database with only a nominal increase in the running time.

We heuristically estimate that the number of solutions of the Diophantine equation mentioned above is of the order of $p^{3\alpha-1}$. We enumerate the number of solutions for some small primes and show that our estimates tally with the actual numbers reasonably closely. These estimates imply that in practice one expects a solution of the Diophantine equation for some α slightly larger than $1/3$. Thus the best running time $L[\sqrt{2/3}]$ of the cubic sieve method seems to be achievable in general.

In the next two sections we briefly describe the linear sieve and the cubic sieve methods. Performance of our implementation and comparison of the two methods for a randomly chosen prime field are presented in sections 4 and 5. Our emphasis is not to set a record on the computation of discrete logarithms, but to point out that our heuristic principles work in practical situations. We, therefore, experimented with a *small* prime (of length 150 bits). Even for this field we get a performance gain (of the cubic sieve method over the linear sieve method) of nearly two. For larger prime fields this performance improvement is expected to get accentuated. In section 6, we derive our analytic estimates of the number of solutions of the congruence mentioned above. We conclude the paper in section 7.

2. The linear sieve method

The first stage for the computation of discrete logarithms over a prime field $\mathbb{F}_p$ using the currently known subexponential methods involves calculation of discrete logarithms of elements of a given subset of $\mathbb{F}_p$, called the *factor base*. To this end, a set of linear congruences are solved modulo $p-1$. Each such congruence is obtained by checking the factorization of certain integers computed deterministically or randomly. For the linear sieve method, the congruences are generated in the following way.

Let $H = \lfloor\sqrt{p}\rfloor + 1$ and $J = H^2 - p$. Then $J \leq 2\sqrt{p}$. For *small* integers c_1, c_2 the right side of the following congruence (henceforth denoted as $T(c_1, c_2)$)

$$(H + c_1)(H + c_2) \equiv J + (c_1 + c_2)H + c_1c_2 \pmod{p} \tag{1}$$

is of the order of $\sqrt{p}$. If the integer $T(c_1, c_2)$ is smooth with respect to the first t primes $q_1, q_2, \ldots, q_t$, that is, if we have a factorization like $J + (c_1 + c_2)H + c_1c_2 = \prod_{i=1}^{t} q_i^{\alpha_i}$, then we have a relation

$$\mathrm{ind}_g(H + c_1) + \mathrm{ind}_g(H + c_2) = \sum_{i=1}^{t} \alpha_i \,\mathrm{ind}_g(q_i). \tag{2}$$

For the linear sieve method the factor base comprises primes less than $L[1/2]$ (so that by the prime number theorem $t \approx L[1/2]/\ln(L[1/2])$ which is again $L[1/2]$) and integers $H + c$ for $-M \leq c \leq M$. The bound M on c is chosen such that $2M \approx L[1/2 + \varepsilon]$ for some small positive real ε. Once we check the factorization $T(c_1, c_2)$ for all values of c_1 and c_2 in the indicated range, we are expected to get $L[1/2 + 3\varepsilon]$ relations like (2) involving the unknown indexes of the factor base elements. If we further assume that the primitive element g is a small prime which itself is in the factor base, then we get a 'free' relation $\mathrm{ind}_g(g) = 1$. The resulting system with asymptotically more equations than unknowns is expected to be of full rank and is solved to compute the discrete logarithms of elements in the factor base.

In order to check the smoothness of the integers $T(c_1, c_2) = J + (c_1 + c_2)H + c_1c_2$ for c_1, c_2 in the range $-M, \ldots, M$ sieving techniques are used. First one fixes a c_1 and initializes to zero an array $\mathfrak{U}$ indexed $-M, \ldots, M$. One then computes for each prime power q^h (q is a small prime in the factor base and h is a small positive exponent) a solution for c_2 of the congruence $(H + c_1)c_2 + (J + c_1 H) \equiv 0 \pmod{q^h}$. If the $\gcd(H + c_1, q) = 1$, *i.e.*, if $H + c_1$ is not a multiple of q, then the solution is given by $d \equiv -(J + c_1 H)(H + c_1)^{-1} \pmod{q^h}$. The inverse in the last equation can be calculated by running the extended gcd algorithm on $H + c_1$ and q^h. Then for each value of c_2 ($-M \leq c_2 \leq M$) that is congruent to d modulo q^h the quantity $\lg q$ is added[†] to the corresponding array locations $\mathfrak{U}_{c_2}$. On the other hand, if $q^{h_1} \| (H + c_1)$ with $h_1 > 0$, we compute $h_2 \geq 0$ such that $q^{h_2} \| (J + c_1 H)$. If $h_1 > h_2$, then for each value of c_2 the expression $T(c_1, c_2)$ is divisible by q^{h_2} and by no higher powers of q. So we add the quantity $h_2 \ln q$ to $\mathfrak{U}_{c_2}$ for all $-M \leq c_2 \leq M$. Finally, if $h_1 \leq h_2$, then we add $h_1 \ln q$ to $\mathfrak{U}_{c_2}$ for all $-M \leq c_2 \leq M$ and for $h > h_1$ solve the congruence as $d \equiv -((J + c_1 H)/q^{h_1})$ $((H + c_1)/q^{h_1})^{-1} \pmod{q^{h-h_1}}$.

Once the above procedure is carried out for each small prime q in the factor base and for each small exponent h,[‡] we check for which values of c_2, the entry of $\mathfrak{U}$ at index c_2 is *sufficiently close* to the value $\lg(T(c_1, c_2))$. These are precisely the values of c_2 such that for the given c_1 the integer $T(c_1, c_2)$ factors smoothly over the small primes in the factor base.

In an actual implementation, one might choose to vary c_1 in the sequence $-M, -M + 1, -M + 2, \ldots$, and, for each c_1, consider only the values of c_2 in the range $c_1 \leq c_2 \leq M$. The criterion for ‘sufficient closeness’ of the array element $\mathfrak{U}_{c_2}$ and $\lg(T(c_1, c_2))$ goes like this. If $T(c_1, c_2)$ factors smoothly over the small primes in the factor base, then it should differ from $\mathfrak{U}_{c_2}$ by a small positive or negative value. On the other hand, if $T(c_1, c_2)$ is not smooth, it would have a factor at least as small as q_{t+1}, and hence the difference between $\lg(T(c_1, c_2))$ and $\mathfrak{U}_{c_2}$ would not be too less than $\lg q_{t+1}$. In other words, this means that the values of the difference $\lg(T(c_1, c_2)) - \mathfrak{U}_{c_2}$ for smooth values of $T(c_1, c_2)$ are well-separatedfrom those for non-smooth values and one might choose for the criterion a check whether the absolute value of the above difference is less than $0.1 \lg q_{t+1}$.

This completes the description of the equation collection phase of the first stage of the linear sieve method. This is followed by the solution of the linear system modulo $p - 1$. The second stage of the method involves computation of discrete logarithms of arbitrary elements of $\mathbb{F}_p^*$ using the database of logarithms of factor base elements. We do not deal with these steps in this paper, but refer the reader to [1, 2, 14] for details.

3. The cubic sieve method

Let us assume that we know a solution (in integers) of the Diophantine equation, henceforth denoted as the *cubic sieve congruence* (CSC).

$$\begin{aligned} X^3 &\equiv Y^2 Z \pmod{p} \\ X^3 &\neq Y^2 Z \end{aligned} \tag{3}$$

with X, Y, Z of the order of p^α for some $1/3 \leq \alpha < 1/2$. Then we have the congruence

$$(X + AY)(X + BY)(X + CY) \equiv Y^2[Z + (AB + AC + BC)X + (ABC)Y] \pmod{p} \tag{4}$$

for all triples (A, B, C) with $A + B + C = 0$. If the bracketed expression on the right side of the above congruence, henceforth denoted as $R(A, B, C)$, is smooth with respect to the first

[†]More precisely, some approximate value of $\lg q$, for example, the integer $\lfloor 1000\, lgq \rfloor$.

[‡]The exponent h can be chosen in the sequence $1, 2, 3, \ldots$, until one finds an h for which none of the integers between $-M$ and M is congruent to d.

t primes $q_1, q_2, \ldots, q_t$, that is, if we have a factorization $R(A, B, C) = \prod_{i=1}^{t} q_i^{\beta_i}$, then we have a relation like

$$\mathrm{ind}_g(X + AY) + \mathrm{ind}_g(X + BY) + \mathrm{ind}_g(X + CY) \equiv \mathrm{ind}_g(Y^2) + \sum_{i=1}^{t} \beta_i \mathrm{ind}_g(q_i) (\mathrm{mod}\, p - 1). \tag{5}$$

If A, B, C are *small* integers, then $R(A, B, C)$ is of the order of p^{α}, since each of X, Y and Z is of the same order. This means that we are now checking integers smaller than $\mathrm{O}(p^{1/2})$ for smoothness over first t primes. As a result, we are expected to get relations like (5) more *easily* than relations like (2) as in the linear sieve method.

This observation leads to the formulation of the cubic sieve method as follows. The factor base comprises primes less than $L[\sqrt{\alpha/2}]$ (so that $t \approx L[\sqrt{\alpha/2}]/\ln(L[\sqrt{\alpha/2}])$ which is again $L[\sqrt{\alpha/2}]$), the integer Y^2 and the integers $X + AY$ for $0 \leq |A| \leq M$, where M is of the order of $L[\sqrt{\alpha/2}]$. The integer $R(A, B, C)$ is, therefore, of the order of $p^{\alpha} L[\sqrt{3\alpha/2}]$ and hence the probability that it is smooth over the first t primes selected as above, is about $L[-\sqrt{\alpha/2}]$. As we check the smoothness for $L[\sqrt{2\alpha}]$ triples (A, B, C) (with $A + B + C = 0$), we expect to obtain $L[\sqrt{\alpha/2}]$ relations like (5).

In order to check for the smoothness of $R(A, B, C) = Z + (AB + AC + BC)X + (ABC)Y$ over the first t primes sieving techniques are employed. We maintain an array $\mathfrak{A}$ indexed $-M, \ldots, M$ as in the linear sieve method. At the beginning of each sieving step we fix C, initialize the array $\mathfrak{A}$ to zero and let B vary. The relation $A + B + C = 0$ allows us to eliminate A from $R(A, B, C)$ as $R(A, B, C) = -B(B + C)(X + CY) + (Z - C^2X)$. For a fixed C we try to solve the congruence

$$-B(B + C)(X + CY) + (Z - C^2X) \equiv 0 (\mathrm{mod}\, q^h), \tag{6}$$

where q is a small prime in the factor base and h is a small positive exponent. This is a quadratic congruence in B. If $X + CY$ is invertible modulo q^h (*i.e.*, modulo q), then the solution for B is given by

$$B \equiv -\frac{C}{2} + \sqrt{(X + CY)^{-1}(Z - C^2X) + \frac{C^2}{4}} (\mathrm{mod}\, q^h), \tag{7}$$

where the square root is modulo q^h. If the expression inside the radical is a quadratic residue modulo q^h, then for each solution d of B in (7) the quantity $\lg q$ is added to those indexes of $\mathfrak{A}$ which are congruent to d modulo q^h. On the other hand, if the expression under the radical is a quadratic non-residue modulo q^h, we have no solutions for B in (6). Finally, if $X + CY$ is non-invertible modulo q, we compute $h_1 > 0$ and $h_2 \geq 0$ such that $q^{h_1} \| (X + CY)$ and $q^{h_2} \| (Z - C^2X)$. If $h_1 > h_2$, then $R(A, B, C)$ is divisible by q^{h_2} and by no higher powers of q for each value of B (and for the fixed C). We add $h_2 \lg q$ to $\mathfrak{A}_i$ for each $-M \leq i \leq M$. On the other hand, if $h_1 \leq h_2$, we add $h_1 \lg q$ to $\mathfrak{A}_i$ for each $-M \leq i \leq M$ and try to solve the congruence $-B(B + C)((X + CY)/q^{h_1}) + ((Z - C^2X)/q^{h_1}) \equiv 0 (\mathrm{mod}\, q^{h-h_1})$ for $h > h_1$. Since $(X + CY)/q^{h_1}$ is invertible modulo q^{h-h_1}, this congruence can be solved similar to (7).

Once the above procedure is carried out for each small prime q in the factor base and for each small exponent h, we check for which values of B, the entry of $\mathfrak{A}$ at index B is *sufficiently close* to the value $\lg(R(A, B, C))$. These are precisely the values of B for which $R(A, B, C)$ issmooth over the first t primes for the given C. The criterion of 'sufficient closeness' of $\mathfrak{A}_B$ and $\lg(R(A, B, C))$ is the same as described in connection with the linear sieve method.

In order to avoid duplication of effort we should examine the smoothness of $R(A, B, C)$ for $-M \leq A \leq B \leq C \leq M$. It can be easily shown that under this condition C varies from 0 to M and for a fixed C the integer B varies in the range $-C/2 \leq B \leq \min(C, M - C)$. Though we do not use the value of A directly in the sieving procedure described above, it is useful[§] to note that A varies from $\max(-2C, -M)$ to $-C/2$ for a fixed C. In particular, A is always negative.

After sufficient number of relations are available, the resulting system is solved modulo $p-1$ and the discrete logarithms of the factor base elements are stored for computation of individual discrete logarithms. We refer the reader to [1, 2, 14] for details on the solution of sparse linear systems and on the computation of individual discrete logarithms with the cubic sieve method.

Attractive as it looks, the cubic sieve method has several drawbacks which impair its usability in practical situations.

1. It is currently not known how to solve the congruence (3) for a general p. And even when it is solvable, how large can α be? For practical purposes α should be as close to 1/3 as possible. No non-trivial results are known to the authors, that can classify primes p according to the smallest possible values of α they are associated with.
2. Because of the quadratic and cubic expressions (in A, B and C) as coefficients of X and Y in $R(A, B, C)$ the integers $R(A, B, C)$ tend to be as large as $p^{1/2}$, even when α is equal to 1/3. If we compare this scenario with that for $T(c_1, c_2)$ (see equation (1)), we see that the coefficient of H is a linear function of c_1 and c_2 and as such, the integers $T(c_1, c_2)$ are larger than $p^{1/2}$ by a small multiplicative factor. This shows that though the integers $R(A, B, C)$ are asymptotically smaller than the integers $T(c_1, c_2)$, the formers are, in practice, only nominally smaller than the latter ones, even when α assumes the most favourable value (namely, 1/3). In other words, when one wants to use the cubic sieve method, one should use values of t (*i.e.*, the number of small primes in the factor base) much larger than the values prescribed by the asymptotic formula for t.
3. The second stage of the cubic sieve method, *i.e.*, the stage that involves computation of individual logarithms, is asymptotically as slow as the equation collection stage. For the linear sieve method, on the other hand, individual logarithms can be computed much faster than the equation collection phase.

In what follows we report an efficient implementation of the cubic sieve method for the case $\alpha = 1/3$, that runs faster than the linear sieve method for the same prime. Our experimentation tends to reveal that the cubic sieve method, when applicable, outperforms the linear sieve method, even when the cardinality of the ground field is around 150 bits long.

4. An efficient implementation of the linear sieve method

Before we delve into the details of the comparison of the linear and cubic sieve methods, we describe an efficient implementation of the linear sieve method. The tricks that help us speed up the equation collection phase of the linear sieve method are very similar to those employed in the quadratic sieve method for integer factorization. See refs. [11–13] for details.

We first recall that at the beginning of each sieving step we find a solution for c_2 modulo q^hin the congruence $T(c_1, c_2) \equiv 0 \pmod{q^h}$ for every small prime q in the factor baseandfor a set of small exponents h. The costliest operation that need to be carried out for each such solution is the computation of a modular inverse (namely, that of $H + c_1$ modulo q^h). As described in ref. [2] and as is evident from our experiments too, calculations of these inverses take more than half of the CPU time needed for the entire equation collection stage. Any trick that reduces the number of computations of the inverses speeds up the algorithm.

[§]For a reason that will be clear in section 5.

One way to achieve this is to solve the congruence every time only for $h = 1$ and ignore all higher powers of q. That is, for every q (and c_1) we check which of the integers $T(c_1, c_2)$ are divisible by q and then add lg q to the corresponding indexes of the array $\mathfrak{U}$. If some $T(c_1, c_2)$ is divisible by a higher power of q, this strategy fails to add lg q the required number of times. As a result, this $T(c_1, c_2)$, even if smooth, may fail to pass the 'closeness criterion' described in section 2. This is, however, not a serious problem, because we may increase the cut-off from a value smaller than $\lg q_t$ to a value $\zeta \lg q_t$ for some $\zeta \geq 1$. This means that some non-smooth $T(c_1, c_2)$ will pass through the selection criterion in addition to some smooth ones that could not, otherwise, be detected. This is reasonable, because the non-smooth ones can be later filtered out from the smooth ones and one might use even trial divisions to do so. Forprimes p of less than 200 bits values of $\zeta \leq 2.5$ work quite well in practice [11, 13].

The reason why this strategy performs satisfactorily is as follows. If q is small, for example $q = 2$, we should add *only* 1 to $\mathfrak{U}_{c_2}$ for every power of 2 dividing $T(c_1, c_2)$. On the other hand, if q is much larger, say $q = 1{,}299{,}709$ (the 10^5th prime), then $\lg q \approx 20.31$ is *large*. But $T(c_1, c_2)$ would not be, in general, divisible by a *high* power of this q. The approximate calculation of logarithm of the smooth part of $T(c_1, c_2)$, therefore, leads to a situation where the probability that a smooth $T(c_1, c_2)$ is actually detected as smooth is quite high. A few relations would be still missed out even with the modified 'closeness criterion', but that is more than compensated by the speed-up gained by the method.

The above strategy helps us in a way other than by reducing the number of modular inverses. For values of p of practical interest the small primes in the factor base are usually single-precision ones. As a result the computation of d (see section 2) can be carried out using single-precision operations only.

We now compare the performance of the modified strategy with that of the original strategy for a value of p of length 150 bits. This prime is chosen as a random one satisfying the conditions (i) $(p - 1)/2$ is also a prime and (ii) p is close to a perfect cube. This second condition is necessary, because for these primes the cubic sieve method is known to be applicable, so that we can compare the performance of the two sieve methods for these primes. Our experiments are based on the CCRYPTO Library, a commercial cryptography toolkit developed by the authors, and are carried out on a 1 GHz Pentium-III machine running Linux version 2.4.18-3 and having 128 Mb RAM. The GNU C Compiler version 2.96 is used.[¶]

In table 1, we compare the performance of the 'exact' version of the algorithm (where all relations are made available by choosing values of $h \geq 1$) with that of the 'approximate' version of the algorithm (in which exponents $h > 1$ are neglected). The CPU times listed in the table include the times for filtering out the 'spurious' relations obtained in the approximate version.[‖] It is evident from the table that the performance gain obtained using the heuristic variant is between 1.5 and 2 for values of ζ between 1.0 and 1.5. These values of ζ clearly suffice for fields of this size. Larger values of ζ (say, 2 or more) imply a very liberal selection criterion which increases the number of subsequent trial divisions considerably, so that we lose the benefits of approximate sieving.

[¶]The timings reported in ref. [6] are based on the Galois Field Library (GFL) [15] developed by the authors and were obtained on a 200 MHz Pentium processor running Linux version 2034 and having the GNU C compiler version 2.7. For this paper we opted for changing the core mathematical library, because CCRYPTO is more stable and efficient than GFL. One can readily observe that the relative performances of the two sieve methods are almost identical under both the libraries.

[‖]The timings reported in ref. [6] do *not* include the filtering overhead and hence are less realistic.

Table 1. Performance of the linear sieve method.

Method	ζ	No. of relations ($\bar{\rho}$)	No. of variables ($\bar{\nu}$)	$\bar{\rho}/\bar{\nu}$	CPU time (seconds)
Exact	0.1	108,637	67,001	1.6214	15,297
Approximate	1.0	108,213	67,001	1.6151	7605
	1.5	108,624	67,001	1.6212	10,455
	2.0	108,636	67,001	1.6214	18,456
	2.5	108,637	67,001	1.6214	45,064

Note: $p = 1{,}320{,}245{,}474{,}656{,}309{,}183{,}513{,}988{,}729{,}373{,}583{,}242{,}842{,}871{,}683$; $t = 7000$; $M = 30{,}000$.

5. An efficient implementation of the cubic sieve method

For the cubic sieve method we employ strategies similar to those described in the last section. That is, we solve the congruence $R(A, B, C) \equiv 0 \pmod q$ for each small prime q in the factor base and ignore higher powers of q that might divide $R(A, B, C)$. As before we set the cut-off at $\zeta \lg q_t$ for some $\zeta \geq 1$. We are not going to elaborate the details of this strategy and the expected benefits once again in this section. We concentrate on an additional heuristic modification of the equation collection phase instead.

We recall from section 3 that we check the smoothness of $R(A, B, C)$ for $-M \leq A \leq B \leq C \leq M$. With this condition C varies from 0 to M. For each value of C we have to execute the entire sieving operation once. For each such sieving operation (that is, for a fixed C) the sieving interval for B is (*i.e.*, the admissible values of B are) $-C/2 \leq B \leq \min(C, M - C)$. Correspondingly $A = -(B + C)$ varies from $\max(-2C, -M)$ to $-C/2$. Itis easy to see that in this case total number of triples (A, B, C) for which the smoothness of $R(A, B, C)$ is examined is $\tau = \sum_{C=0}^{M}(1 + \lfloor C/2 \rfloor + \min(C, M - C)) \approx M^2/2$. The number of unknowns, that is, the size of the factor base, on the other hand, is $\nu \approx 2M + t$.

If we remove the restriction $A \geq -M$ and allow A to be as negative as $-\lambda M$ for some $1 < \lambda \leq 2$, then we profit in the following way. As before we allow C to vary from 0 to M keeping the number of sieving operations fixed. Since A can now assume values smaller than $-M$, the sieving interval increases to $-C/2 \leq B \leq \min(C, \lambda M - C)$. As a result the total number of triples (A, B, C) becomes $\tau_\lambda = \sum_{C=0}^{M}(1 + \lfloor C/2 \rfloor + \min(C, \lambda M - C)) \approx (M^2/4)(4\lambda - \lambda^2 - 1)$, whereas the size of the factor base increases to $\nu_\lambda \approx (\lambda + 1)M + t$. (Note that with this notation the value $\lambda = 1$ corresponds to the original algorithm and $\tau = \tau_1$ and $\nu = \nu_1$.) The ratio τ_λ/ν_λ is approximately proportional to the number of smooth integers $R(A, B, C)$ generated by the algorithm divided by the number of unknowns. Therefore, λ should be set at a value for which this ratio is maximum. If one treats t and M as constants, then the maximum is attained at $\lambda^* = -U + \sqrt{U^2 + 4U + 1}$, where $U = (M + t)/M = 1 + (t/M)$. As we increase U from 1 to ∞ (or, equivalently the ratio t/M from 0 to ∞), the value of λ^* increases monotonically from $\sqrt{6} - 1 \approx 1.4495$ to 2. In table 2 we summarize the variation of τ_λ/ν_λ for some values of U. These values of U correspond from left to right to $t \ll M$, $t \approx M/2$, $t \approx M$ and $t \approx 2M$ respectively. The corresponding values of λ^* are respectively 1.4495, 1.5414, 1.6056 and 1.6904. It is clear from the table that for practical ranges of values of U the choice $\lambda = 1.5$ gives performance quite close to the optimal.

We note that this scheme keeps M and the range of variation of C constant and hence does not increase the number of sieving steps and, in particular, the number of modular inverses and square roots. It is, therefore, advisable to apply the trick (with, say, $\lambda = 1.5$) instead of increasing M.

Table 2. Variation of τ_λ/ν_λ with λ.

	τ_λ/ν_λ (approx.)			
λ	$U = 1$	$U = 1.5$	$U = 2$	$U = 3$
1	$0.2500M$	$0.2000M$	$0.1667M$	$0.1250M$
1.5	$0.2750M$	$0.2292M$	$0.1964M$	$0.1527M$
2	$0.2500M$	$0.2143M$	$0.1875M$	$0.1500M$
λ^*	$0.2753M$	$0.2293M$	$0.1972M$	$0.1548M$

Now we report the performance of the cubic sieve method for various values of the parameters ζ and λ. We continue to work in the prime field $\mathbb{F}_p$ with

$$p = 1{,}320{,}245{,}474{,}656{,}309{,}183{,}513{,}988{,}729{,}373{,}583{,}242{,}842{,}871{,}683$$

as in the last section. For this prime we have

$$X = \lfloor \sqrt[3]{p} \rfloor + 1 = 1{,}097{,}029{,}305{,}312{,}372, \quad Y = 1, \quad Z = 31{,}165$$

as a solution of (3), which corresponds to $\alpha = 1/3$.

To start with we fix $\lambda = 1.5$ and examine the variation of the performance of the equation collection stage with ζ. We did not implement the 'exact version of this method in which one tries to solve (6) for exponents $h > 1$ of q. Table 3 lists the experimental details for the 'approximate method. As in table 1 the CPU times include the times for filtering out thespurious relations available by the more generous closeness criterion for the approximate method. For the cubic sieve method values of ζ around 1.0 works quite well for our prime p.

In table 4 we fix ζ at 1.5 and list the variation of the performance of the cubic sieve method for some values of λ. It is clear from the table that among the cases observed the largest value of the ratio $\bar{\rho}/\bar{\nu}$ is obtained at $\lambda = 1.5$. (The theoretical maximum is attained at $\lambda \approx 1.6$.) We also note that changing the value of λ incurs variation in the running time by at most 11%. Thus our heuristic allows us to build a larger database at very little extra cost. On the other hand, if one fixes $\lambda = 1.0$ as in the original method and increases M in order to obtain a database of size same as that for the modified method with $\lambda = 1.5$, the running time typically increases by 20% or more. As an example, taking $M = 12{,}500$, $t = 10{,}000$, $\zeta = 1.5$ and $\lambda = 1$ gave us 61,079 relations in 35,001 variables in a total time of 6271 seconds. This amounts to about 22% increase in the running time over the parameter settings $M = 10{,}000$, $t = 10{,}000$, $\zeta = 1.5$ and $\lambda = 1.5$.

5.1 ***Performance comparison with linear sieve***

The speed-up obtained by the cubic sieve method over the linear sieve method is about 2 for the field of size around 150 bits. For larger fields this speed-up is expected to be more. It is,

Table 3. Performance of the cubic sieve method of various values of ζ.

ζ	No. of relations ($\bar{\rho}$)	No. of variables ($\bar{\nu}$)	$\bar{\rho}/\bar{\nu}$	CPU time (seconds)
1.0	54,805	35,001	1.5658	3626
1.5	54,865	35,001	1.5675	5148
2.0	54,868	35,001	1.5676	8742

Note: $p = 1{,}320{,}245{,}474{,}656{,}309{,}183{,}513{,}988{,}729{,}373{,}583{,}242{,}842{,}871{,}683$; $t = 10{,}000$; $M = 10{,}000$; $\lambda = 1.5$.

Table 4. Performance of the cubic sieve method of various values of λ.

λ	No. of relations ($\bar{\rho}$)	No. of variables ($\bar{\nu}$)	$\bar{\rho}/\bar{\nu}$	CPU time (seconds)
1.0	43,434	30,001	1.4478	4808
1.5	54,865	35,001	1.5675	5148
1.6	56,147	36,001	1.5596	5270
2.0	58,234	40,001	1.4558	5340

Note: $p = 1{,}320{,}245, 474{,}656{,}309{,}183{,}513{,}988{,}729{,}373{,}583{,}242{,}842{,}871{,}683$; $t = 10{,}000$; $M = 10{,}000$; $\zeta = 1.5$.

therefore, evident that the cubic sieve method, at least for the case $\alpha = 1/3$, runs faster than the linear sieve counterpart for the practical range of sizes of prime fields.

6. Number of solutions of the cubic sieve congruence

Now we derive some estimates on the number of solutions of the CSC. For the sake of convenience we quote the following well-known result [16, chapter 3].

Euler summation formula: If $f\colon \mathbb{R} \to \mathbb{R}$ has a continuous derivative f' in the closed interval $[a, b]$, $0 < a < b$, then

$$\sum_{a<n\le b} f(n) = \int_a^b f(t)\,\mathrm{d}t + \int_a^b (t - \lfloor t \rfloor) f'(t)\,\mathrm{d}t + f(b)(\lfloor b \rfloor - b) - f(a)(\lfloor a \rfloor - a).$$

The following estimates are easy consequences of the Euler summation formula:

$$\sum_{1\le n\le x} \frac{1}{n} = \ln x + \gamma + \mathrm{O}\left(\frac{1}{x}\right). \tag{8}$$

$$\sum_{1\le n\le x} \frac{1}{n^s} = \frac{x^{1-s}}{1-s} + \zeta(s) + \mathrm{O}(x^{-s}) \quad \text{if } s > 0,\ s \ne 1. \tag{9}$$

$$\sum_{1\le n\le x} n^s = \frac{x^{s+1}}{s+1} + \mathrm{O}(x^s) \quad \text{if } s \ge 0. \tag{10}$$

Here γ is the Euler constant defined by $\gamma = \lim_{n\to\infty}(1/1 + 1/2 + \cdots + 1/n - \ln n) = 0.57721566\ldots$ and $\zeta(s)$ is the Riemann zeta function defined for all real $s > 0$, $s \ne 1$, as

$$\zeta(s) = \begin{cases} \displaystyle\sum_{n=1}^{\infty} \frac{1}{n^s} & \text{if } s > 1, \\ \displaystyle\lim_{x\to\infty}\left(\sum_{n=1}^{\infty} \frac{1}{n^2} - \frac{x^{1-s}}{1-s}\right) & \text{if } 0 < s < 1. \end{cases}$$

We define for $n \in \mathbb{N}$ the integer $d(n)$ to be the total number of (positive integral) divisors of n. By the Euler summation formula it follows that for a real $x \ge 1$ we have

$$\sum_{n\le x} d(n) = x \ln x + (2\gamma - 1)x + \mathrm{O}(\sqrt{x}). \tag{11}$$

Let us now introduce a few notations related to the set of solutions of the CSC.

$$S = \{(X, Y, Z) \mid X^3 \equiv Y^2 Z \pmod p, \quad 1 \le X, Y, Z < p\}.$$
$$S_= = \{(X, Y, Z) \in S \mid X^3 = Y^2 Z\}$$
$$S_{\neq} = \{(X, Y, Z) \in S \mid X^3 \neq Y^2 Z\}.$$
$$S_\alpha = \{(X, Y, Z) \in S_{\neq} \mid 1 \le X, Y, Z \le p^\alpha\}.$$

For the cubic sieve method we are not interested in solutions of the CSC in $S_=$. However, it is easy to estimate the cardinality of $S_=$. This, in turn, gives the cardinality of $S_{\neq}$. The sets S_α for $1/3 \le \alpha < 1/2$ are extremely important for the cubic sieve method. The smallest possible value of α for which S_α is non-empty, determines the best running time of the cubic sieve method.

6.1 *Cardinality of S*

For each value of $X, Y \in \mathbb{F}_p^*$, we have a unique solution for $Z \in \mathbb{F}_p^*$ satisfying the CSC. Therefore,

$$\#S = (p-1)^2 = \Theta(p^2). \tag{12}$$

6.2 *Cardinality of $S_=$*

Choose $1 \le X < p$ and a solution $(X,\ Y,\ Z) \in S_=$. Let the prime factorization of X be $X = p_1^{\beta_1} p_2^{\beta_2} \cdots p_r^{\beta_r}$, where p_i are distinct primes and $\beta_i > 0$. Therefore, $Y^2 Z = X^3 = p_1^{3\beta_1} p_2^{3\beta_2} \cdots p_r^{3\beta_r}$. Since $Y^2 | X^3$, for each $i = 1, \ldots, r$ the exponent h such that p_i^h divides Y must be one of $0,\ 1,\ 2, \ldots,\ \lfloor 3\beta_i/2 \rfloor$. Some choices of these exponents may lead to $Y > p$. We neglect this for the time being and see that for the given X the total number of choices for Y (and hence for Z) is $\le \prod_{i=1}^r (1 + \lfloor 3\beta_i/2 \rfloor) \le \prod_{i=1}^r (1 + 3\beta_i/2) \le (3/2)^r \prod_{i=1}^r (1+\beta_i) = (3/2)^r d(X) \le p^{\lg(3/2)} d(X) \le p^{0.585} d(X)$. If we sum this quantity over all $X,\ 1 \le X < p$, and use (11), we get

$$\begin{aligned}\#S_= &\le p^{0.585} \sum_{1 \le X < p} d(X) = p^{0.585}((p-1)\ln(p-1) + (2\gamma - 1)(p-1) + \mathrm{O}(\sqrt{p}))\\ &= \mathrm{O}(p^{1.585} \ln p).\end{aligned} \tag{13}$$

Next we derive a lower bound for $S_=$. First we note that each $X = Y = Z \in \mathbb{F}_p^*$ is in $S_=$ and hence $\#S_= \ge p - 1$. We can determine a bound slightly better than this. To do so we first fix Y. Then $Y^2 \le X^3 < Y^2 p$, since $1 \le Z < p$. Let the values of X that satisfy $Y^2 \le X^3 < Y^2 p$ be $X_1,\ X_2, \ldots,\ X_s$ where $s = (Y^2 p)^{1/3} - (Y^2)^{1/3} + \mathrm{O}(1)$ and $X_i = X_1 + i - 1$. Since $Y < p$, it is clear that each X_i above is less than p. We consider only those values of X_i for which $Y^2 | X_i^3$. We see that if $Y | X_i$, then $Y^2 | X_i^3$. Hence for a fixed Y the total number of solutions $(X,\ Y,\ Z) \in S_=$ is greater than or equal to $((Y^2 p)^{1/3} - (Y^2)^{1/3} + \mathrm{O}(1))/Y$. If we sum this over all Y, we get applying the formulas (8) and (9)

$$\begin{aligned}\#S_= &\ge \sum_{1 \le Y < p} \frac{(Y^2 p)^{1/3} - (Y^2)^{1/3} + \mathrm{O}(1)}{Y} = (p^{1/3} - 1) \sum_{1 \le Y < p} \frac{1}{Y^{1/3}} + \mathrm{O}(\ln p)\\ &= (p^{1/3} - 1)\left[\frac{(p-1)^{1-1/3}}{1 - 1/3} + \zeta(1/3) + \mathrm{O}(p^{-1/3})\right] + \mathrm{O}(\ln p) = \frac{3}{2} p + \mathrm{O}(p^{2/3}),\end{aligned} \tag{14}$$

where $\zeta(1/3) = -0.97336024\ldots$. In particular, $\#S_= = \Omega(p)$.

6.3 *Cardinality of $S_{\#}$*

Since S is the disjoint union of $S_=$ and $S_{\neq}$, equations (12)–(14) give

$$(p-1)^2 - p^{1.585}\ln(p-1) + \mathrm{O}(p) \le \#S_{\#} \le (p-1)^2 - \frac{3}{2}p + \mathrm{O}(p^{2/3}). \tag{15}$$

In particular, $\#S_{\#} = \Theta(p^2)$.

6.4 *Heuristic estimate of $\#S_\alpha$*

In this section, we count the number of solutions of the CSC with $X,\ Y,\ Z \le p^\alpha,\ X^3 \neq Y^2Z$. Since the cubic sieve method demands $1/3 \le \alpha < 1/2$, we consider α only in this range, though our argument is valid for any $0 \le \alpha \le 1$.

We first fix Y and write $X^3 = Y^2Z + kp$ for some $k \in \mathbb{Z}\backslash\{0\}$. We then see that $X^3 \equiv kp \pmod{Y^2}$. This implies that k must be chosen such that kp is a cubic residue modulo Y^2. We are interested only in the cubic residues $1^3,\ 2^3, \ldots,\ \lfloor p^\alpha \rfloor^3$ modulo Y^2.

Claim 1 *Irrespective of whether the $\lfloor p^\alpha \rfloor$ cubic residues $1^3,\ 2^3, \ldots,\ \lfloor p^\alpha \rfloor^3$ are distinct modulo Y^2 or not, for any n distinct random values of kp, we expect $n\lfloor p^\alpha \rfloor / Y^2$ distinct solutions for $(X,\ Y,\ Z)$ with $X \le p^\alpha$.*

Proof This is because if $X_1^3 \equiv X_2^3 \equiv kp \pmod{Y^2}$ for some k with $X_1 \neq X_2$, then we get two solutions $(X_1,\ Y_1,\ Z_1)$ and $(X_2,\ Y_2,\ Z_2)$. In particular, if the *good* cubic residues $1^3,\ 2^3, \ldots,\ \lfloor p^\alpha \rfloor^3$ assumes m distinct values modulo Y^2, then from the n given values of kp, we expect nm/Y^2 values of k to correspond to the set of these cubic residues. Each such residue, on the other hand, is associated, on an average, with $\lfloor p^\alpha \rfloor / m$ solutions with $X \le p^\alpha$. Hence the expected number of solutions $(X,\ Y,\ Z)$ corresponding to the given n random values of kp is $(nm/Y^2)(\lfloor p^\alpha \rfloor / m) = n\lfloor p^\alpha \rfloor / Y^2$. ■

Now we allow k to vary in the range

$$\frac{1}{p} - p^{\alpha-1}Y^2 \le k \le p^{3\alpha-1} - \frac{Y^2}{p}. \tag{16}$$

This corresponds to a total of $p^{3\alpha-1} - Y^2/p + p^{\alpha-1}Y^2 + \mathrm{O}(1)$ values of $k \neq 0$. Since $kp = X^3 - Y^2Z$ and $1 \le X,\ Z \le p^\alpha$, for the fixed value of Y chosen above we have $1 - p^\alpha Y^2 \le kp \le p^{3\alpha} - Y^2$ which implies (16). Note, however, that the converse is not true, that is, all values of k prescribed by (16) do not lead to values of $1 \le X,\ Z \le p^\alpha$. We will force $1 \le X \le p^\alpha$ and consider only those solutions for which $1 \le Z \le p^\alpha$.

Now we make the following *heuristic assumption*:

Assumption 1 *As k varies in the range given by* (16), *the integers kp behave as random integers modulo Y^2.*

This is a reasonable assumption since the $\gcd(Y^2,\ p) = 1$. This assumption together with Claim 1 guarantees an expected number of approximately

$$\left(p^{3\alpha-1} + \mathrm{O}(1) + \left(p^{\alpha-1} - \frac{1}{p}\right)Y^2\right)\frac{p^\alpha}{Y^2} \tag{17}$$

solutions $(X,\ Y,\ Z)$ with the given Y. All these solutions correspond to $1 \le X \le p^\alpha$, but not necessarily to $1 \le Z \le p^\alpha$ as told before. The inequalities (16) together with $X^3 = Y^2Z + kp$

show that the range of variation of Z is

$$1 - \frac{p^{3\alpha} - X^3}{Y^2} \le Z \le p^\alpha + \frac{X^3 - 1}{Y^2}. \tag{18}$$

At this point we make the *second heuristic assumption*:

ASSUMPTION 2 *All these values of Z are equally likely to occur.*

For any $1 \le X \le p^\alpha$ the inequalities (18) prescribe $p^\alpha - 1 + (p^{3\alpha} - 1)/Y^2 + \mathrm{O}(1)$ non-zero values for Z including the values $1 \le Z \le p^\alpha$. Therefore, by Assumption 2, the probability that Z lies in the range $1 \le Z \le \lfloor p^\alpha \rfloor$ is

$$\approx \frac{p^\alpha}{p^\alpha - 1 + (p^{3\alpha} - 1)/Y^2} = \frac{p^\alpha Y^2}{(p^\alpha - 1)Y^2 + p^{3\alpha} - 1} > \frac{Y^2}{Y^2 + p^{2\alpha}} \ge \frac{Y^2}{2p^{2\alpha}}. \tag{19}$$

This probability multiplied by (17) gives the expected number of solutions in S_α with the given fixed Y to be greater than or equal to $(1/2p^\alpha)(p^{3\alpha-1} + \mathrm{O}(1) + (p^{\alpha-1} - 1/p)Y^2)$. We finally vary Y with $1 \le Y \le p^\alpha$ and use (10) to obtain:

$$\begin{aligned}\text{Expected cardinality of } S_\alpha &\ge \frac{1}{2p^\alpha}\Bigg(p^{3\alpha-1}p^\alpha + \mathrm{O}(p^\alpha) \\ &\quad + \left(p^{\alpha-1} - \frac{1}{p}\right)\left(\frac{(p^\alpha)^3}{3} + \mathrm{O}((p^\alpha)^2)\right)\Bigg) \\ &= \frac{2}{3}p^{3\alpha-1} + \mathrm{O}(\max(1,\ p^{2\alpha-1})) = \Omega(p^{3\alpha-1}). \end{aligned} \tag{20}$$

For sufficiently large p the term $(2/3)p^{3\alpha-1}$ dominates and one might expect to get a solution if $(2/3)p^{3\alpha-1} \gg 1$, say, for example, if $(2/3)p^{3\alpha-1} \ge 1000$, *i.e.*, if

$$\alpha \ge \frac{1}{3} + \frac{\ln(1500)}{3\ln p}.$$

For example, if $p \approx 2^{500}$, then $\alpha = 0.34037$ is expected to make S_α non-empty.

We have noted that Assumption 1 is reasonable and gives a good picture of the average situation. Assumption 2, on the other hand, is difficult to justify mathematically. We assumed an average scenario to get an estimate of $\#S_\alpha$. As we pointed out earlier, our aim is not to prove the non-emptiness or otherwise of S_α, but to compute an approximate value of its cardinality with the hope that this behaviour is general enough to portray the average situation. We shortly show that up to a constant factor our estimates are quite close to the experimental values we obtained from a set of small scale experiments. These experimental results together with our theoretical estimate tempt us to make the following conjecture:

CONJECTURE 6.1 *The expected cardinality of S_α is asymptotically equal to $\chi\, p^{3\alpha-1}$ for all $0 \le \alpha \le 1$ and for some constant $\chi \approx 1$. (Note that* (15) *demands* $\chi = 1$.)

Few primes of special forms might not obey the conjectured estimates. But we do not see any such special form – both experimentally and theoretically. The bulk of the derivation of (20) is based on the cubic residues module Y^2 for integers $Y = 1, 2, 3, \ldots$. The prime p does not seem to play an important role in connection with Assumption 1. The second assumption, however, can be influenced by the choice of p and may lead to situations we failed to visualize.

6.5 *Experimental evidence*

We experimented with randomly generated primes of size around 30 bits. We actually enumerated all the solutions of the CSC for various values of α in the range $0.33 \le \alpha \le 0.50$. We tabulate these experimental values together with the theoretical estimates obtained as $\#S_\alpha = \lfloor (2/3) p^{3\alpha-1} \rfloor$. We also list the conjectured values given by $\#S_\alpha = \lfloor \chi\, p^{3\alpha-1} \rfloor$ with $\chi = 1$.

Table 5 gives the data for a random 30-bit prime $p = 1{,}034{,}302{,}223$. Though we have experimented with many primes of this size, we give the values of $\#S_\alpha$ only for a typical value. This is because we get exactly similar pattern of variation of $\#S_\alpha$ with α for all of our test primes. Thus a single representative is sufficient to reflect the scenario.

The table clearly shows that apart from constant factors the experimental, estimated and conjectured values exhibit the same pattern of variation of $\#S_\alpha$ with α. For α close to 0.33 the relation between these values is little erratic. As α increases, say $\alpha \ge 0.40$, the ratio of the estimated value to the experimental value and the ratio of the conjectured value to the experimental value tend to approach constant values. In particular, the conjectured values are quite close to the experimental values. It remains unsettled if this pattern continues to hold for general primes of larger sizes, say for primes of size $\le$1000 bits. Since at present no algorithms are known to solve the CSC in time polynomial in $\lg p$, we cannot experiment with higher values of p. In addition, even if such an algorithm exists, one should spend $O(p^{2\alpha})$ time for enumerating all the solutions in S_α. This makes it infeasible to continue the experimental study with primes of practical interest. These small-scale experiments give us some confidence about the theoretical estimates derived in this section.

In spite of all these theoretical and experimental exercises the question of existence or otherwise of a solution of the CSC for some $1/3 \le \alpha < 1/2$ continues to remain unanswered. It is believed that a solution exists [1, 17]. Our analysis only strengthens the belief in favor of a solution and to that effect is stronger than the argument presented in ref. [17].

Table 5. $\#S_\alpha$ for $p = 1{,}034{,}302{,}223$ (A 30-bit prime).

	Values of $\#S_\alpha$				
α	(a)	(b)	(c)	(b)/(a)	(c)/(a)
0.33	0	0	0	–	–
0.34	1	1	1	1.00	1.00
0.35	1	1	2	1.00	2.00
0.36	2	3	5	1.50	2.50
0.37	5	6	9	1.20	1.80
0.38	9	12	18	1.33	2.00
0.39	23	22	34	0.96	1.48
0.40	53	42	63	0.79	1.19
0.41	98	78	118	0.80	1.20
0.42	185	147	220	0.79	1.19
0.43	368	274	411	0.74	1.17
0.44	695	511	766	0.74	1.10
0.45	1363	952	1429	0.70	1.05
0.46	2475	1776	2664	0.72	1.08
0.47	4646	3310	4965	0.71	1.07
0.48	8815	6170	9256	0.70	1.05
0.49	16,615	11,502	17,253	0.69	1.04
0.50	31,451	21,440	32,160	0.68	1.02

Note: (a) experimental, (b) estimated, (c) conjectured.

7. Conclusion

In this paper we have described various practical aspects of efficient implementation of the linear and the cubic sieve methods for the computation of discrete logarithms over prime finite fields. We have also compared the performances of these two methods and established the superiority of the latter method over the former for the cases when p is close to a perfect cube. It, however, remains unsettled whether the cubic sieve method performs equally well for a general prime p. We have provided some heuristic estimates on the number of solutions of a cubic congruence which is at the heart of the cubic sieve method. These estimates tend to corroborate the fact that for random primes the performance of the cubic sieve method is expected not to deteriorate much (compared to the case for special primes we studied). However, designing an algorithm for computing a solution of this cubic congruence continues to remain an open problem and stands in the way of the general acceptance of the cubic sieve method. Last but not the least, we need performance comparison of the cubic sieve method with the number field sieve method.

References

[1] Coppersmith, D., Odlyzko, A.M. and Schroeppel, R., 1986, Discrete logarithms in *GF*(p). *Algorithmica*, **1**, 1–15.

[2] LaMacchia, B.A. and Odlyzko, A.M., 1991, Computation of discrete logarithms in prime fields. *Designs, Codes, and Cryptography*, **1**, 46–62.

[3] McCurley, K.S., 1990, The discrete logarithm problem. Cryptology and computational number theory. *Proceeding of Symposia in Applied Mathematics*, **42**, 49–74.

[4] Menezes, A.J. (Ed.), 1993, *Applications of Finite Fields* (Dordnecht: Kluwer Academic Publishers).

[5] Odlyzko, A.M., 1985, Discrete logarithms and their cryptographic significance. Advances in Cryptology: Proceedings of Eurocrypt84, LNCS No. 209, pp. 224–314.

[6] Das, A. and Veni Madhavan, C.E., 1999, Performance comparison of linear sieve and cubic sieve algorithms for discrete logarithms over prime fields. *Proceedings of ISAAC99*, LNCS No. 1741, pp. 295–306.

[7] Gordon, D.M., 1993, Discrete logarithms in *GF*(p) using the number field sieve. *SIAM Journal of Discrete Mathematics*, **6**, 124–138.

[8] Schirokauer, O., Weber, D. and Denny, T., 1996, Discrete logarithms, the effectiveness of the index calculus method. *Proceedings of ANTS II*, LNCS no. 1122, pp. 337–361.

[9] Weber, D., 1996, Computing discrete logarithms with the general number field sieve. Procceedings of ANTS II, LNCS no. 1122, pp. 99–114.

[10] Weber, D. and Denny, T., 1998, The solution of McCurleys discrete log challenge. Procceedings of Crypto98, LNCS no. 1462, pp. 458–471.

[11] Bressoud, D.M., 1989, *Factorization and Primary Testing*, Undergraduate Text in Mathematics (Berlin: Springer-Verlag).

[12] Gerver, J., 1983, Factoring large numbers with a quadratic sieve. *Mathematics of Computation*, **41**, 287–294.

[13] Silverman, R.D., 1987, The multiple polynomial quadratic sieve. *Mathematics of Computation*, **48**, 329–339.

[14] LaMacchia, B.A. and Odlyzko, A.M., 1991, Solving large sparse linear systems over finite fields. Advances in Cryptology – CRYPTO90, LNCS #537, pp. 109–133.

[15] Das, A. and Veni Madhavan, C.E., 1998, Galois field library. Reference manual. Technical Report No IISc-CSA-98-05, Department of Computer Science and Automation, Indian Institute of Science.

[16] Apostol, T.M., 1976, *Introduction to Analytic Number Theory*, Undergraduate Text in Mathematics (Berlin: Springer-Verlag).

[17] Lenstra, A.K. and Lenstra, H.W., 1990, Algorithms in number theory. In: J. van Leeuwen, (Ed.) *Handbook of Theoretical Computer Science*, pp. 675–715.

[18] Cohen, H., 1993, *A Course in Computational Algebraic Number Theory*, Graduate Text in Mathematics No. 138, (Berlin: Springer-Verlag).