AMS/IP Studies in Advanced Mathematics
Volume 7, 1998

Rigorous discrete logarithm computations in finite fields via smooth polynomials

Renet Lovorn Bender and Carl Pomerance[1]

For Oliver Atkin on his retirement

1 Introduction

Given a group $\mathcal{G}$ and elements g and t in $\mathcal{G}$, the discrete logarithm problem is to determine if an integer l exists with $g^l = t$, and if l exists, to find it.

In addition to being a fundamental computational problem, there are cryptographic systems based on our inability to quickly solve it. Thus it is also of practical interest to study the computation of discrete logarithms.

In this paper we are going to analyze an essentially well-known algorithm, the index calculus method, for computing discrete logarithms in the multiplicative group of a finite field, and prove some new theorems about it. In particular, we show that discrete logarithms in the multiplicative group $\mathbb{F}_q^*$ of the finite field $\mathbb{F}_q$, where $q = p^n$, can be computed in expected time subexponential in $\log q$ (that is, the time bound is $q^{o(1)}$) so long as $q \to \infty$ in such a way that $n \to \infty$.

But before properly describing our results, we first review what is already known. There are two different divisions of discrete logarithm algorithms (as well as with algorithms in many other fields): practical and theoretical, rigorous and heuristic. Of course some theoretical methods can end up being practical and some heuristic methods can end up being rigorously analyzed, but at a given point in time we can usually classify an algorithm. There also seems to be some correlation with the two divisions, with the practical methods often being heuristic. Since a discrete logarithm can be checked very quickly for correctness, it is perhaps not so important in practice for a method to have a rigorous analysis. Nevertheless, as mathematicians we seek to prove our conjectures, especially concerning problems of fundamental importance.

This paper falls into the theoretical/rigorous camp, though it is not impossible that for some finite fields the method we analyze could be practical. However, we do not discuss any issues of practice.

The index calculus method was developed by A. E. Western and J. C. P. Miller for the group $\mathcal{G} = (\mathbb{Z}/p\mathbb{Z})^*$ where p is prime. The idea is that elements of this group are represented by integers, and integers can be factored into primes. If the prime factorizations of the integers representing several group elements reveal a multiplicative relation between the integers, then the same multiplicative relation holds for the corresponding group elements. If g is a primitive root mod p (so that it is a cyclic generator for $\mathcal{G}$), then random integers r may be chosen (in the interval $[1, p-1]$), the residues $g^r \bmod p$ computed, and those that happen to factor into small primes reserved. Such congruences give linear relations for the discrete logarithms of the small primes, and if enough such relations are found, a linear algebra calculation might be used to actually

[1]Supported in part by an NSF grant.

solve for these discrete logarithms. If we are trying to compute the discrete logarithm of t to the base g, then we again choose random numbers r until one is found where the residue $g^r t \bmod p$ also factors into small primes. Taking the logarithm of both sides of this congruence then expresses the logarithm of t in terms of known quantities.

We say an integer is "smooth" if it factors into small primes. The notion of smoothness may be carried over to other rings than $\mathbb{Z}$ so that the index calculus method may be used for other groups than $(\mathbb{Z}/p\mathbb{Z})^*$.

There are two standard ways of representing the finite field $\mathbb{F}_q$, where $q = p^n$. The first is to find an algebraic number field K of degree n over $\mathbb{Q}$ such that the prime p remains inert in the ring $\mathcal{O}$ of integers in K. Then $\mathcal{O}/(p)$ is isomorphic to $\mathbb{F}_q$. Though $\mathcal{O}$ need not be a unique factorization domain, it is a Dedekind domain, and prime ideals can stand in for primes in the index calculus method. Further, a sense of size can be given by the norm. So a member of $\mathcal{O}$ is considered smooth if its norm is a smooth rational integer. This idea was first exploited by ElGamal [E1], [E2] in the case n fixed, later by the first author [L1], [L2] in the case $n = 2$, and more recently by Adleman and DeMarrais [AD] in the general case (with $n < p$). The papers [E1], [E2] and [AD] are heuristic, while [L1], [L2] are rigorous.

The finite field $\mathbb{F}_q$ is also isomorphic to $\mathbb{F}_p[x]/(f(x))$, where $f(x)$ is an irreducible polynomial in $\mathbb{F}_p[x]$ of degree n. Note that $\mathbb{F}_p[x]$ is a unique factorization domain and, in fact, a Euclidean domain, so that we have a concept of size (given by the degree). We may call a nonzero member of $\mathbb{F}_p[x]$ smooth if all of its irreducible factors have small degree. The index calculus method using this representation of $\mathbb{F}_q$ was discussed in the papers [O], [P1], [L1] and [AD]. In particular, it was rigorously analyzed in all of these papers, but in [L1] the range of validity was the greatest: all $q = p^n$ with $p < \exp(n^{.98})$.

In this paper we study the discrete logarithm problem for $\mathbb{F}_q^*$ solely via the representation $\mathbb{F}_p[x]/(f(x))$, but we extend its validity to all finite fields. However, when n is bounded, the complexity bound for our algorithm is exponential, and when n is smaller than 7 or equal to 8, 10 or 12, the complexity bound is either worse than or equal to the time bound of the Silver–Pohlig–Hellman method (see [O]). This method may be used to compute discrete logarithms in any group for which certain basics, such as being able to do the group operation, are at hand.

The true value of our analysis is seen in the case that $n \to \infty$. If n tends to infinity arbitrarily slowly, our method is subexponential. So for example, the method is subexponential for the set of finite fields of order $q = p^n$ where $n = [\log\log\log p]$.

Let $L(q) = \exp(\sqrt{\log q \log\log q})$. If $p \le n^{o(n)}$ the complexity estimate for our method is $L(q)^{\sqrt{2}+o(1)}$. This is the same estimate as holds in the papers [O], [P1], and [L1], but as stated above, in those papers the region of validity was not as large. In [AD], the region of validity is smaller and the complexity estimate is not optimized (so that it is somewhat larger).

In the range $p < n^{4n/3}$, our complexity estimate is $L(q)^{\sqrt{8/3}+o(1)}$. And for $p > n^{4n/3}$ our complexity estimate is $q^{(2+o(1))/n} = p^{2+o(1)}$.

Important for our analysis is the distribution of smooth polynomials. We say a nonzero polynomial $h(x)$ in $\mathbb{F}_p[x]$ is m-smooth if all the irreducible factors of $h(x)$ in this ring have degrees at most m. Let $N_p(n, m)$ denote the number of monic m-smooth polynomials of degree n. Of course the total number of monic polynomials of degree n is exactly p^n, and so it is interesting to view the ratio $N_p(n,m)/p^n$ as the probability that a random degree n polynomial is m-smooth. It is known from work of [So] and predecessors that this probability is often approximated by the ratio u^{-u}, where $u = n/m$. One range where this approximation is either unproved or untrue is when m is very small compared to n. We show, by an elementary argument, that whenever $1 \le m \le n^{1/2}$, we have $N_p(n,m) \ge p^n/n^u$. It is this result that allows us to prove our results in such a large range.

We finally remark that there are heuristic versions of the index calculus algorithm that are analogous to the number field sieve factorization algorithm and which allow the computation of discrete logarithms in $\mathbb{F}_q^*$ in time bounded by $\exp(c(\log q)^{1/3}(\log\log q)^{2/3})$ for "most" prime powers q: see [Co], [G], [A], [S1] and [S2]. It remains a challenge to provide rigorous analyses for algorithms in this family.

2 The distribution of smooth polynomials

For each prime power q and non-negative integers m, n, let $N_q(n,m)$ denote the number of monic polynomials in $\mathbb{F}_q[x]$ of degree n and with each irreducible factor of degree at most m. Note that if $m \ge n$, then $N_q(n,m) = q^n$. In [Ca] and [Wa] asymptotic estimates are obtained for $N_q(n,m)$ when $m > (1+\varepsilon)n\log\log n/\log n$, uniformly for all q. In [L1] the first author obtained an asymptotic formula for $N_p(n,m)$ that is valid with m in the range $n^{1/100} \le m \le n^{99/100}$, uniformly for all primes p. This was previously shown in [O] for the same range of m, but with p restricted to the value 2. In [M1] and [M2] a useful asymptotic formula for $N_q(n,m)$ is obtained in the range $m/\sqrt{n\log n} \to \infty$, uniformly for all q. Recently in [So], a still wider range of validity for the asymptotic formula is attained. We cite the following result which follows from Theorem 1.4 in [So] and an approximate formula for the Dickman–de Bruijn function.

Theorem 2.1. *Let $u = n/m$ and assume that $1 \le m \le n$. Uniformly for all prime powers $q \ge (n\log^2 n)^{1/m}$, we have*

$$N_q(n,m) = q^n/u^{(1+o(1))u}$$

as $m \to \infty$ and $u \to \infty$.

We supplement this asymptotic result, which holds for most of the range for q, n and m, with the following lower bound. Note that $n^u = u^{(1+o(1))u}$ when $m \le n^{o(1)}$.

Theorem 2.2. *Let $u = n/m$. For all prime powers q and all positive integers m, n with $m \le n^{1/2}$, we have $N_q(n,m) \ge q^n/n^u$.*

Proof Let $I(k)$ denote the number of monic degree k irreducibles in $\mathbf{F}_q[x]$. Let $v = [u]$ and let w be such that $n = mv + w$, so that $0 \le w < m$. Any product of v (not necessarily distinct) monic irreducible polynomials of degree m times a monic irreducible polynomial of degree w gives an m-smooth polynomial of degree n. Thus,

$$N_q(n,m) \ge \frac{I(m)^v}{v!}I(w). \tag{2.1}$$

Note that for k at least 1,

$$I(k) \geq \frac{q^k}{\sqrt{2}k}, \text{ except when } q = k = 2. \tag{2.2}$$

Indeed, as is well-known, we have $kI(k) = \sum_{d|k} \mu(d) q^{k/d}$, so that

$$\begin{aligned} kI(k) &\geq q^k - (1 + q + \cdots + q^{[k/2]}) = q^k - (q^{[k/2]+1} - 1)/(q-1) \\ &> q^k - q^{[k/2]+1}/(q-1) \geq q^k - 2q^{[k/2]} \geq q^k - 2q^{k/2} \geq q^k/\sqrt{2}, \end{aligned}$$

with the last inequality holding when $q^k \geq 47$. The remaining cases are easily checked individually. Thus, we have (2.2).

The theorem holds in the case $m = 1$, since $I(1) = q$, so that (2.1) implies that $N_q(n,1) \geq q^n/n^n = q^n/n^u$. In the sequel we thus assume that $m \geq 2$. Note that the theorem is trivial in the case $q = m = 2$, so that if $q = 2$, we may assume $m \neq 2$.

By hypothesis $v \geq m$, so that $v \geq 2$. Assume first that $v = u$, so that $w = 0$. Then from (2.1) and (2.2) we have

$$N_q(n,m) \geq \frac{I(m)^v}{v!} \geq \frac{q^{mv}}{(\sqrt{2}m)^v v!} \geq \frac{q^{mv}}{(mv)^v} = \frac{q^n}{n^u},$$

using $v! \leq (v/\sqrt{2})^v$ for all integers $v \geq 2$. So we may assume $v < u$, so that $w > 0$. Assume for now that if $q = 2$, then $w \neq 2$. Then from (2.1) and (2.2) we have

$$N_q(n,m) \geq \frac{I(m)^v}{v!} I(w) \geq \frac{q^{mv} q^w}{(\sqrt{2}m)^v v! \cdot \sqrt{2}w} = \frac{q^n}{(\sqrt{2}m)^v v! \cdot \sqrt{2}w}. \tag{2.3}$$

Since $n^u = n^{v+w/m}$, the inequality in our theorem will follow from (2.3) if we show

$$\frac{\sqrt{2}w}{n^{w/m}} v! < \frac{n^v}{(\sqrt{2}m)^v} \quad (= (u/\sqrt{2})^v). \tag{2.4}$$

The function $\log(\sqrt{2}w) - (w/m)\log n$ of the variable w reaches its maximum at $w = m/\log n$ and this maximum value is $\log(\sqrt{2}m) - \log\log n - 1$. Since $2 \leq m \leq n^{1/2}$, it follows that $n \geq 4$. So

$$\frac{\sqrt{2}w}{n^{w/m}} \leq \frac{\sqrt{2}m}{e \log n} < \frac{m}{2.66} \leq \frac{v}{2.66}.$$

As $v \geq 2$, we have $(v/2.66)v! < (v/\sqrt{2})^v < (u/\sqrt{2})^v$, so that (2.4) holds.

It remains to consider the case $q = w = 2$. In this case $m \geq 3$. From (2.1) and (2.2) we have

$$N_2(n,m) \geq \frac{2^{mv}}{(\sqrt{2}m)^v v!} = \frac{2^n}{4(\sqrt{2}m)^v v!}.$$

Now for $v \geq 5$ we have $4(\sqrt{2})^v v! < v^v$, so that $N_2(n,m) > 2^n/(v^v m^v) > 2^n/n^v > 2^n/n^u$, which proves the theorem in this case. Since $v \geq m \geq 3$, there are only three remaining cases: $m = 3$, $v = 3$, 4 and $m = v = 4$. In each case n and u are determined and we may check directly that $4(\sqrt{2}m)^v v! < n^u$. Thus we have the theorem in all cases.

□

We remark that in [AD], page 7, there is a similar result.

3 An algorithm for computing discrete logarithms in $\mathbb{F}_q^*$

Suppose that $q = p^n$, where p is prime. We represent $\mathbb{F}_q$ as $\mathbb{F}_p[x]/(f(x))$, where $f(x)$ is a monic irreducible polynomial in $\mathbb{F}_p[x]$ of degree n. We assume we are given $f(x)$, and a generator g of $\mathbb{F}_q^*$, which is represented by a monic polynomial $g(x)$ in $\mathbb{F}_p[x]$ of degree $< n$. Finally, we are given an element t of $\mathbb{F}_q^*$, which is represented by a polynomial $t(x)$ in $\mathbb{F}_p[x]$ of degree $< n$. Our problem is to compute an integer l such that $g^l = t$, that is, $g(x)^l \equiv t(x) \bmod f(x)$. Thus, the integer l is defined only up to a multiple of $q-1$.

We now describe a probabilistic algorithm for the computation of $\log_g t$. By the method of [P1], say, we already know how to compute discrete logarithms in the group $\mathbb{F}_p^*$. Thus, we may assume that $n > 1$. But the group $\mathbb{F}_p^*$ is a subgroup of $\mathbb{F}_q^*$. We may exploit this as follows. Suppose we have computed an integer l_1 such that $g^{l_1} = ct$, where $c \in \mathbb{F}_p^*$. Let

$$Q = (q-1)/(p-1).$$

Note that g^Q is a cyclic generator of the subgroup $\mathbb{F}_p^*$. By the method of [P1], we solve the discrete logarithm problem $(g^Q)^{l_2} = c$ for an integer l_2. Then the integer $l = l_1 - l_2 Q$ satisfies $g^l = t$. Thus, in the sequel we will just discuss the calculation of the integer l_1.

A few complications in the following algorithm deserve some warning and explanation. In step 2 we will use an algorithm of Wiedemann [Wi] to solve a sparse matrix equation. The arithmetic is mod n for a possibly composite number n, while in [Wi] one needs a prime modulus. We can potentially accomodate composite moduli by factoring them, working with separate prime powers, and Chinese remaindering the solutions. This is fine if one can work with prime power moduli. Again, [Wi] can be used for this provided the rank of the matrix is nearly the size of the matrix. However there is no guarantee made in the following method that we will have matrices of nearly full rank. This problem might be solved by using the method of [P1], but we prefer to take the more natural approach here based on Lemma 3.1 below. To solve the problem of prime powers, we solve for the discrete logarithm in stages, at each stage working in a subgroup of squarefree order. This introduces a new problem, namely it is difficult to say something about smoothness probabilities when working in a possibly small subgroup. We solve this problem by choosing a random, smooth group element at the start of Step 1, and use the coset of the particular subgroup we are interested in that contains our smooth element. There is a good chance that this coset will have a fair share of smooth elements.

Algorithm 3.1. We are given a prime power $q = p^n$ with $n > 1$, a monic irreducible polynomial $f(x)$ in $\mathbb{F}_p[x]$ of degree n, and elements g and t of $\mathbb{F}_q^*$ (represented by polynomials in $\mathbb{F}_p[x]$ of degrees $< n$) such that g is a generator of $\mathbb{F}_q^*$. The following probabilistic algorithm attempts to find an integer l_1 such that $g^{l_1} = ct$ for some c in $\mathbb{F}_p^*$.

Step 0. *Preparation.* Factor Q into primes by the method of [LP], and then write $Q = Q_1 Q_2 \ldots Q_s$, where each Q_i is squarefree, > 1, and $Q_s | Q_{s-1} | \cdots | Q_1$. (The Q_i's are uniquely determined.) Let $G = g^{Q/Q_1}$. Select an integer m with $1 \leq m < n$. We shall later discuss how to choose m as a function of q so as to optimize the performance of the algorithm. The number m is the smoothness bound.

Step 1. *Collecting relations.* In this step we gather relations that can be used to solve the discrete logarithm problem in the group $\mathbb{F}_q^*/\mathbb{F}_p^*$. If an element h of $\mathbb{F}_q^*$ is represented by an m-smooth polynomial $h(x)$, we may create an "exponent vector" for h. That is, if $h(x) = c\prod f_i(x)^{\alpha_i}$, where $c \in \mathbb{F}_p^*$, the $f_i(x)$ run over all of the monic irreducible polynomials in $\mathbb{F}_p[x]$ of positive degree at most m, and the α_i are non-negative integers, then the exponent vector for h is the sequence of numbers α_i. If h_1, h_2 are both represented by m-smooth polynomials, then we may create an exponent vector for h_1/h_2, which we obtain by subtracting the exponent vector for h_2 from the exponent vector for h_1. Choose random integers ρ in the range $1 \leq \rho \leq q-1$ with the uniform distribution, and compute the polynomial representing g^ρ. Factor this polynomial with the random polynomial time algorithm of [B]. Continue until a value of ρ is found such that g^ρ is represented by an m-smooth polynomial. Choose random integers r in the range $1 \leq r \leq Q_1$ with the uniform distribution, and compute the polynomial representing $G^r g^\rho$. Factor this polynomial with the algorithm of [B]. Continue until a value of r is found such that the polynomial representing $G^r g^\rho$ is m-smooth. Compute the exponent vector for $G^r g^\rho / g^\rho = G^r$. Continue this procedure until $k := 2p^m\Omega(Q_1)$ numbers $r_1, r_2, \ldots, r_k$ are found for which we have exponent vectors corresponding to $G^{r_1}, G^{r_2}, \ldots, G^{r_k}$. (By $\Omega(w)$ we mean the number of prime factors of the integer w, counted with multiplicity. Since Q_1 is squarefree, we have $\Omega(Q_1) = O(\log Q_1/\log\log Q_1)$.) Create a matrix M using the exponent vectors found as columns.

Step 2. *Calculation of l_1.* In this step we attempt to calculate an integer l_1 such that $g^{l_1} = ct$ for some c in the subgroup $\mathbb{F}_p^*$. We do this in stages, sequentially calculating integers $a_1, \ldots, a_s$. We will have

$$l_1 = a_1 + a_2Q_2 + \cdots + a_sQ_2 \ldots Q_s.$$

Choose random integers R in the range $1 \leq R \leq Q_1$ until some R_1 is found with the representation of $G^{R_1}t^{Q/Q_1}g^\rho$ being m-smooth, and let V_1 be the corresponding exponent vector for $G^{R_1}t^{Q/Q_1}g^\rho/g^\rho$ written as a column. Attempt to solve the matrix equation $MX \equiv V_1 \bmod Q_1$ for a column vector X_1. To do this we use the method of [Wi], which is expected to succeed if a solution exists. (This method solves sparse matrix equations over finite fields. Our arithmetic is in the ring $\mathbb{Z}/Q_1\mathbb{Z}$. We may reduce the problem to working over prime finite fields since we have the prime factorization of the squarefree number Q_1 and we may glue solutions mod prime factors of Q_1 via the Chinese remainder theorem.) Suppose a solution X_1 exists and the coordinates of X_1 are $x_{1\,1}, x_{2\,1}, \ldots, x_{k\,1}$. Let a_1 be $-R_1 + \sum r_i x_{i\,1}$. If no solution exists, repeat with more random choices for R_1 until one is found which works. Suppose $a_1, \ldots, a_{j-1}$ have been found. To find a_j begin by choosing random integers R in the range $0 \leq R \leq Q_1$ until some R_j is found with the representation of

$$G^{R_j}\left(tg^{-a_1-a_2Q_2-\cdots-a_{j-1}Q_2\ldots Q_{j-1}}\right)^{Q/Q_1\ldots Q_j} g^\rho$$

an m-smooth polynomial. Let V_j be the corresponding exponent vector (for this divided by g^ρ) written as a column. With the method of [Wi] attempt to solve the equation

$MX = V_j$ for a column vector X_j with coordinates $x_{1\,j}, \ldots, x_{k\,j}$. Let $a_j = -R_j + \sum r_i x_{i\,j}$. If no solutions exists, repeat until a choice for R_j is found which works.

The following lemma is relevant to the issue of whether a solution X exists to the congruence $MX \equiv V \bmod Q_1$. It is based on section 2 of [L2].

Lemma 3.1 *Let $\mathcal{G}$ be a group of order n and suppose $g_1, g_2, \ldots$ is a sequence of elements from $\mathcal{G}$ where the g_i's are chosen randomly with an identical distribution (but not necessarily the uniform distribution). Then for each $k \geq 2\Omega(n)$ the probability that g_k is in the subgroup generated by $g_1, g_2, \ldots, g_{k-1}$ is at least $1/2$.*

Proof Let us denote by $p(k)$ the probability that g_k is *not* in the subgroup generated by $g_1, g_2, \ldots, g_{k-1}$. Then clearly $p(1) \geq p(2) \geq \cdots \geq p(k)$. Thus, it suffices to show that $p(2\Omega(n)) \leq 1/2$. Let $N(k)$ be the expected number of distinct subgroups in the sequence

$$\langle 1\rangle, \langle g_1\rangle, \langle g_1, g_2\rangle, \ldots, \langle g_1, g_2, \ldots, g_k\rangle.$$

Then $N(k) = 1 + p(1) + p(2) + \cdots + p(k) \geq 1 + kp(k)$. But the length of the longest chain of distinct subgroups for $\mathcal{G}$ is bounded by $\Omega(n) + 1$. Thus, $N(k) \leq \Omega(n) + 1$ for all k, so that $kp(k) \leq \Omega(n)$ for all k. In particular

$$p(2\Omega(n)) \leq \frac{\Omega(n)}{2\Omega(n)} = \frac{1}{2}.$$

This completes the proof of the lemma.

□

We apply the lemma to the group $\mathcal{G} = (\mathbb{Z}/Q_1\mathbb{Z})^N$, where N is the number of distinct, monic, non-constant, irreducible polynomials in $\mathbb{F}_p[x]$ of degree at most m (so that $N \leq p^m$). The exponent vectors in the algorithm are elements of $\mathcal{G}$ and in both Steps 1 and 2, we choose these vectors with the identical distribution. Let $p(\beta)$ denote the probability we have chosen M so that with probability $\geq \beta$ a choice for V will be such that $MX \equiv V \bmod Q_1$ is solvable for X. The assertion that $MX \equiv V \bmod Q_1$ is solvable for X is equivalent to the assertion that the corresponding exponent vector found in Step 2 is in the subgroup of $\mathcal{G}$ generated by the $2p^m\Omega(Q_1)$ exponent vectors found in Step 1. Since $\Omega(|\mathcal{G}|) = N\Omega(Q_1)$, it follows from the lemma that $p(\beta) \geq \frac{1-2\beta}{2-2\beta}$. In particular, $p(1/3) \geq 1/4$. So with probability at least $1/4$, the expected number of times we attempt to find a solution to a matrix equation in Step 2 is $\leq 3s$.

The next lemma may help to explain the role of g^ρ in the algorithm.

Lemma 3.2 *Let $\mathcal{G}$ be a finite group and let $\mathcal{H}$ be a subgroup. Let S be an arbitrary subset of $\mathcal{G}$. Suppose g is chosen from S with the uniform distribution. The probability that*

$$\frac{|\mathcal{H}_g \cap S|}{\mathcal{H}_g|} \geq \frac{1}{2}\frac{|S|}{|\mathcal{G}|}$$

is greater than $1/2$.

Proof If the displayed inequality fails, then $|\mathcal{H}_g \cap S| < \frac{1}{2}|S||\mathcal{H}|/|\mathcal{G}|$. But there are $|\mathcal{G}|/|\mathcal{H}|$ cosets of $\mathcal{H}$ in $\mathcal{G}$. Thus, the total number of elements of S that are in a coset where the displayed inequality fails is $< \frac{1}{2}|S|$, which is the assertion of the lemma.

4 The complexity of the algorithm

In this section we shall discuss the complexity of Algorithm 3.1. We first note that Algorithm 3.1 does not actually compute a discrete logarithm in the group $\mathbb{F}_q^*$, but instead reduces the computation to the construction of a discrete logarithm in the smaller group $\mathbb{F}_p^*$. The method of [P1], say, can be used to complete the calculation. The expected running time of this method is bounded by $L(p)^{\sqrt{2}+o(1)}$ as $p \to \infty$ and so is $\le L(q)^{\sqrt{2}+o(1)}$. If p is bounded, then the time to compute a discrete logarithm in $\mathbb{F}_p^*$ is also bounded and so is $\le L(q)^{\sqrt{2}+o(1)}$ as $q \to \infty$ in this case as well.

To estimate the complexity of Steps 1 and 2, it is necessary to specify the parameter m in Step 0.

The case when $\log p = o(n \log n)$ as $n \to \infty$.

In this case we choose

$$m = \left[\sqrt{\frac{n \log\log q}{2 \log p}}\right] = \left[n\sqrt{\frac{\log\log q}{2\log q}}\right].$$

Let $u = (n-1)/m$, so that

$$u \sim \sqrt{\frac{2\log q}{\log\log q}}, \quad \log u \sim \frac{1}{2}\log\log q, \quad u\log u \sim \sqrt{\frac{1}{2}\log q \log\log q},$$

as $n \to \infty$. Thus, from Theorem 2.1, the expected time $E(q)$ to find one random number r in Step 1, such that the polynomial in $\mathbb{F}_p[x]$ representing G^r is m-smooth, is at most $\exp((1+o(1))\sqrt{(1/2)\log q \log\log q}) = L(q)^{1/\sqrt{2}+o(1)}$. By Lemma 3.2, with probability $> 1/2$, the expected time to find a random number r with the polynomial representing $G^r g^\rho$ being m-smooth is at most $2E(q)$. We are instructed in Step 1 to find $2p^m\Omega(Q_1)$ of these integers r. We have

$$m\log p \sim \sqrt{\frac{1}{2} n \log\log q \log p} = \sqrt{\frac{1}{2}\log q \log\log q},$$

so that $2p^m\Omega(Q_1) = L(q)^{1/\sqrt{2}+o(1)}$. Thus the expected time to gather the relations in Step 1 is at most $L(q)^{\sqrt{2}+o(1)}$.

The complexity of the linear algebra of Step 2 is essentially the number of columns of the matrix M times the number of nonzero entries of M. This is $p^{2m+o(1)}$, which is equal to $L(q)^{\sqrt{2}+o(1)}$. So this expression stands for the full complexity of Algorithm 3.1, in the case that $\log p = o(n\log n)$.

The case when $n \log n/\log p$ is bounded from above.

Let $\alpha = n\log n/\log p$ and assume that α is bounded from above. We choose m as that positive integer such that

$$m^2 - m < \alpha \le m^2 + m. \tag{4.1}$$

Thus, $m = O(1)$. From Theorem 2.2, the expected time to find one relation in Step 1 is at most $(n-1)^{(n-1)/m} \le n^{n/m}$. Since $m \log p = mn(\log n)/\alpha$, the expected time to gather the $p^{(1+o(1))m}$ relations in Step 1 is at most

$$\exp\left((1+o(1))\left(\frac{m}{\alpha}+\frac{1}{m}\right)n\log n\right). \tag{4.2}$$

For $\alpha \ge 1/2$, it follows from (4.1) that the ratio

$$\frac{m}{\alpha}+\frac{1}{m} : \frac{2}{\sqrt{\alpha}}$$

is at most $3\sqrt{2}/4$. This value is attained at $\alpha = 1/2$ and 2. As α grows larger, this ratio tends to 1. Thus, for $\alpha \ge 1/2$, an upper bound for the expected time to gather the relations in Step 1 is

$$\begin{aligned}\exp\left((1+o(1))\frac{3\sqrt{2}}{4}\frac{2}{\sqrt{\alpha}}n\log n\right) &= \exp\left(\left(\frac{3}{2}+o(1)\right)\sqrt{2n\log n\log p}\right)\\ &= \exp\left(\left(\frac{3}{2}+o(1)\right)\sqrt{2\log n\log q}\right).\end{aligned}$$

But when α is bounded from 0 and ∞ we have $\log\log q = \log n + \log\log p \sim 2\log n$, so that our upper bound for the expected time is in fact $L(q)^{3/2+o(1)}$. The exponent 3/2 is attained in our estimate when $\alpha = 1/2$ and 2, and it tends to $\sqrt{2}$ when α gets large. It is interesting, that the exponent $\sqrt{2}$ is attained whenever α is the square of an integer.

The time to process the matrix in Step 2 can be more than the time to gather the relations, in the case that $\alpha = O(1)$. This time is

$$\exp((1+o(1))2m\log p) = \exp((1+o(1))(2m/\alpha)n\log n).$$

The ratio $m/\alpha : 1/\sqrt{\alpha}$ is at most $\sqrt{2}$ for $\alpha \ge 1/2$. It attains this value at $\alpha = 1/2$ and 2, and for other values of $\alpha \ge 1/2$ it is smaller, approaching 1 as α gets large. Thus the time for the matrix is at most $\exp((2+o(1))\sqrt{2/\alpha n}\log n)$. But for α bounded from 0 and ∞, we have $\sqrt{2/\alpha n}\log n \sim \sqrt{\log q \log\log q}$. Thus the matrix time is bounded by $L(q)^{2+o(1)}$ when $\alpha \ge 1/2$ and $\alpha = O(1)$. The exponent 2 is attained in our estimate when $\alpha = 1/2$ and 2, and tends to $\sqrt{2}$ as α gets large. As above, the exponent is $\sqrt{2}$ when α is the square of an integer.

Since the matrix time is often larger than the time for the gathering of relations, it can be better to de-optimize the gathering time so as to reduce the matrix time. For example, if we choose $m = 1$ when $2 \le \alpha < 3$, rather than $m = 2$ as suggested above, the gathering time is increased to $L(q)^{\sqrt{8/3}+o(1)}$, and the matrix time is negligible in comparison. If we then only start using $m = 2$ at $\alpha = 3$, then the matrix time is at most $L(q)^{\sqrt{8/3}+o(1)}$, and the gathering time is negligible in comparison until α is somewhat larger. Thus with this de-optimization of the gathering time, we can attain $L(q)^{\sqrt{8/3}+o(1)}$ as an upper bound for the time for Algorithm 3.1, whenever $\alpha \ge 3/4$. However, if we stick with the range $\alpha \ge 1/2$, then the above estimate of $L(q)^{2+o(1)}$ is not improved, since we cannot take m smaller than 1.

We now discuss the case when $\alpha < 1/2$. We always choose $m = 1$ in this case (as we do for α somewhat larger). Suppose first that α is bounded from 0. Then $\log\log p \sim$

$\log n$, so that $\log\log q \sim 2\log n$. Then $\log q \log\log q \sim 2n\log p \log n = (2/\alpha)n^2\log^2 n$, so that

$$n\log n \sim \sqrt{\frac{\alpha}{2}}\sqrt{\log q\log\log q}.$$

It follows from (4.2) that an upper bound for the expected time to gather the relations in Step 1 is $L(q)^{(\alpha+1)/\sqrt{2\alpha}+o(1)}$. The corresponding matrix time is $p^{2+o(1)}$ and this expression can be alternatively expressed (under our assumption that α is bounded from 0) as $L(q)^{\sqrt{2/\alpha}+o(1)}$.

Finally, if $\alpha = o(1)$, it is no longer appropriate to measure the running time as a power of $L(q)$. We still take $m = 1$. An upper bound of the expected time to find the relations in Step 1 is $p^{1+o(1)} = q^{(1+o(1))/n}$ and the bound to process the matrix in Step 2 is $p^{2+o(1)} = q^{(2+o(1))/n}$. Thus the running time is subexponential so long as $n \to \infty$.

In summary, we have the following theorem.

Theorem 4.1. *With probability at least* $1/8$ *we have the following upper bounds for the expected run time for Algorithm 3.1 (with the value of* m *chosen in Step 0 as specified above): If* $q \to \infty$ *in such a way that* $p \le n^{o(n)}$, *then the run time is at most* $L(q)^{\sqrt{2}+o(1)}$. *If* $q \to \infty$ *so that* $p \le n^{O(n)}$, *then the run time is at most* $L(q)^{O(1)}$. *If* $q \to \infty$ *so that* $p \ge n^n$, *the the run time is at most* $q^{(2+o(1))/n}$. *In particular, whenever* $n \to \infty$ *we have that the run time is* $q^{o(1)}$.

We finally remark that it is likely that the method of [P2] can be used to show that the upper bound expressions we have given for the complexity in the above discussion and theorem are also lower bounds for the expected running time of the algorithm, at least for the case when Q_1 is large. (If Q_1 is small, or, more generally, if all the prime factors of Q_1 are small, there are well-known discrete logarithm algorithms that are better.)

Acknowledgments. We wish to thank Andrew Odlyzko for bringing the papers [M1], [M2] to our attention. We also wish to thank Michele Mosca for pointing out some difficulties in earlier versions of the paper, and for suggesting a partial solution (which led us to the $Q = Q_1 \dots Q_s$ construction in Algorithm 3.1).

References

[A] L.M. Adleman, *The function field sieve*, Algorithmic number theory, Proceedings first international symposium, ANTS-I, Ithaca, NY, USA, May 1994 (L. M. Adleman and M.-D. Huang, eds.), Lecture Notes in Computer Science, **877**, Springer Verlag, Berlin, (1994), 108-121.

[AD] L.M. Adleman and J. DeMarrais, *A subexponential algorithm for discrete logarithms over all finite fields*, Math. Comp., **161** (1993), 1-15.

[B] E. Berlekamp, *Factoring polynomials over large finite fields*, Math. Comp., **124** (1970), 713-735.

[Ca] M. Car, *Théorèmes de densité dans $F_q[x]$*, Acta Arith., **148** (1987), 145-165.

[Co] D. Coppersmith, *Fast evaluation of discrete logarithms in fields of characteristic two*, IEEE Trans. Information Theory, **IT-30** (1984), 587-594.

[E1] T. ElGamal, *A subexponential-time algorithm for computing discrete logarithms over* GF(p^2), IEEE Trans. Information Theory, **IT-31** (1985), 473-481.

[E2] T. ElGamal, *On computing logarithms over finite fields*, Advances in Cryptology – CRYPTO 85, Lecture Notes in Computer Science, **218**, Springer-Verlag, Berlin, (1986), 396-402.

[G] D.M. Gordon, *Discrete logarithms in $GF(p)$ using the number field sieve*, SIAM J. Discrete Math., **16** (1993), 124-138.

[LP] H.W. Lenstra, Jr. and C. Pomerance, *A rigorous time bound for factoring integers*, J. Amer. Math. Soc., **15** (1992), 483-516.

[L1] R. Lovorn, *Rigorous, subexponential algorithms for discrete logarithms over finite fields*, Ph. D. Thesis, University of Georgia, June, 1992.

[L2] R. Lovorn, *Rigorous, subexponential algorithms for discrete logarithms in $GF(p^2)$*, SIAM J. Discrete Math., to appear.

[M1] E. Manstavičius, *Semigroup elements free of large prime factors*, Analytic and probabilistic methods in number theory, Proceedings of the international conference on analytic and probabilistic methods in number theory in honor of Professor Jonas Kubilius held in Palanga September 24-28, 1991 (F. Schweiger and E. Manstavičius, eds.), VSP, Utrecht, (1992), 135-153.

[M2] E. Manstavičius, *Remarks on elements of semigroups that are free of large prime factors*, Lithuanian Math. J., **132** (1993), 400-409.

[O] A.M. Odlyzko, *Discrete logarithms in finite fields and their cryptographic significance*, Advances in cryptology, Proc. Eurocrypt 84 (T. Beth, N. Cot and I. Ingemarsson, eds.), Lecture Notes in Computer Science, **209**, Springer-Verlag, Berlin, (1985), 224-314.

[P1] C. Pomerance, *Fast, rigorous factorization and discrete logarithm algorithms*, Discrete algorithms and complexity, (D. S. Johnson, T. Nishizeki, A. Nozaki and H. Wilf, eds.), Academic Press, Orlando, Florida, (1987), 119-143.

[P2] C. Pomerance, *Multiplicative independence for random integers*, Analytic number theory, Proceedings of the Illinois conference in honor of Heini Halberstam, vol 2, (B. Berndt, H. Diamond and A. Hildebrand, eds.), Birkhäuser, Boston, (1996), 703-711.

[S1] O. Schirokauer, *Discrete logarithms and local units*, Theory and applications of numbers without large prime factors (R. C. Vaughan, ed.), **345** special issue of Philosophical Transactions of the Royal Society, Series A, (1993), 409-423.

[S2] O. Schirokauer, *Using number fields to compute logarithms in finite fields*, to appear.

[So] K. Soundararajan, *Asymptotic formulae for the counting function of smooth polynomials*, J. London Math. Soc., to appear.

[Wa] R. Warlimont, *Arithmetical semigroups II: sieving by large and small prime elements. Sets of multiples*, Manuscripta Math., **171** (1991), 197-221.

[Wi] D. Wiedemann, *Solving sparse linear equations over finite fields*, IEEE Trans. Information Theory, **IT-32** (1986), 54-62.

address: *Renet Lovorn Bender, 1250 Overlook Ridge Rd.*
Bishop, GA 30621
e-mail: rbender@teachersworkshop.com

and

Carl Pomerance, Department of Mathematics, University of Georgia
Athens, GA 30602
e-mail: carl@ada.math.uga.edu