ÉCOLE POLYTECHNIQUE

LIX

A General Framework for Subexponential Discrete Logarithm Algorithms

A. Enge and P. Gaudry

LIX/RR/00/04

23 juin 2000

LABORATOIRE D'INFORMATIQUE
Unité mixte CNRS n°7650
École polytechnique 91128 Palaiseau Cedex FRANCE

A General Framework for Subexponential Discrete Logarithm Algorithms

Andreas Enge
Lehrstuhl für Diskrete Mathematik,
Optimierung und Operations Research
Universität Augsburg
86135 Augsburg
Deutschland
enge@math.uni–augsburg.de

Pierrick Gaudry
LIX
École polytechnique
91128 Palaiseau
France
gaudry@lix.polytechnique.fr

June 23, 2000

Abstract

We describe a generic algorithm for computing discrete logarithms in groups of known order in which a smoothness concept is available. The running time of the algorithm can be proved without using any heuristics and leads to a subexponential complexity in particular for finite fields and class groups of number and function fields which were proposed for use in cryptography. In class groups, our algorithm is substantially faster than previously suggested ones. The subexponential complexity is obtained for cyclic groups in which a certain smoothness assumption is satisfied. We also show how to modify the algorithm for cyclic subgroups of arbitrary groups when the smoothness assumption can only be verified for the full group.

Keywords: discrete logarithm, index calculus, class groups, subexponentiality.

Résumé

Nous décrivons un algorithme générique pour calculer des logarithmes discrets dans les groupes d'ordre connu pour lesquels une certaine notion de friabilité est disponible. Le temps de calcul de l'algorithme peut être prouvé sans utiliser d'heuristiques et donne une complexité sous-exponentielle pour les corps finis et les groupes de classes de corps de nombres et de corps de fonctions, qui ont été proposés en cryptographie. Pour les groupes de classes, notre algorithme est substantiellement plus rapide que ceux précédemment suggérés. La complexité sous-exponentielle est obtenue pour les groupes cycliques dans laquelle une certaine hypothèse sur la friabilité est satisfaite. Nous montrons aussi comment modifier l'algorithme pour des sous-groupes cycliques quand l'hypothèse de friabilité ne peut être montrée que pour le groupe entier.

Mots clés: logarithme discret, calcul d'index, groupes de classes, sous-exponentialité.

1 Introduction

Every finite abelian group is isomorphic to a direct sum of cyclic groups $\mathbb{Z}/e_i\mathbb{Z}$, where the e_i are unique provided that e_{i+1} divides e_i. Given such a group a natural task is to make this isomorphism explicit, hence two fundamental computational problems arise. The first one is to determine the order of the group and its structure (i.e. the e_i), and the second one is to compute discrete logarithms in it. When those two tasks are feasible, most of the questions concerning the group can be transferred to the simpler representation.

In this paper we assume that the group order is known and we concentrate on the discrete logarithm computation: Given an additively written abelian group G, an element $g_1 \in G$ and another element $g_2 \in \langle g_1 \rangle$, the problem consists of finding an integer l such that $g_2 = lg_1$. Obviously, the difficulty of the problem depends on the representation of the group. In $(\mathbb{Z}/N\mathbb{Z}, +)$, for instance, the problem reduces to taking an inverse modulo N and can be solved in polynomial time by the Euclidean algorithm. In general the problem is hard and, following Diffie and Hellman's and ElGamal's constructions [DH76, ElG85], it is possible to base a cryptosystem on it.

The first examples of such groups were the multiplicative groups of finite fields; it turned out that in this case the discrete logarithm problem can be solved in expected subexponential time by so-called "index calculus algorithms" (see [Odl99] and the references therein), so that the key size in a secure system based on such a group must be relatively large.

Other important examples are class groups of number fields [McC89, BW88] and Jacobians, i.e. divisor class groups, of elliptic [Mil86, Kob87] and hyperelliptic curves [Kob89] over finite fields.

Extending the "index calculus" method for finite fields, algorithms with conjectured or rigorously proven subexponential running time for the discrete logarithm problem in class groups of number fields [Buc90, Dül91] and high-genus hyperelliptic Jacobians [ADH94, Eng99] and related structures [MST99] were devised. However, the constants of the running time bounds for these algorithms are substantially weaker than for the finite field case. This is due to the fact that the index calculus algorithms for class groups proceed by solving both computational tasks mentioned in the first paragraph: In a first step, the structure of the group under consideration is determined as a product of cyclic groups, in a second step, the actual discrete logarithm is computed. For the multiplicative groups of finite fields, which are cyclic and of known order, the first phase can be omitted.

In the present article we close the gap between the running times of discrete logarithm algorithms for finite fields and class groups. We provide a general framework for solving the discrete logarithm problem in finite abelian groups which possess an analogue to the unique prime decomposition of integers. Our basic additional assumption is that the group be cyclic and of known order, which closely follows the special case of finite fields. If a certain smoothness assumption is satisfied, our algorithm has a provable subexponential expected running time. Examining how the well-studied discrete logarithm problem in finite fields fits into our framework, we recover the same running time as for the fastest rigorously analysed algorithms known so far. We obtain corresponding running time results for class groups, which constitute a major improvement compared to the above mentioned algorithms.

The main result of this paper is the following:

Theorem 1 *There exists a probabilistic algorithm that solves a discrete logarithm problem in the Jacobian of a hyperelliptic curve of genus g over the finite field $\mathbb{F}_q$ when the group is cyclic of known* *order and $g/\log q$ tends to infinity. It takes expected time*

$$L_{q^g}(\sqrt{2}+o(1)).$$

Assuming the extended Riemann hypothesis, there exists a probabilistic algorithm that solves a discrete logarithm problem in the class group of an imaginary quadratic field of discriminant $-D$ when the group is cyclic of known order N. It takes expected time

$$L_N(\sqrt{2}+o(1)) = L_D(1+o(1)).$$

(Here $L_x(c)$ stands for $\exp(c\sqrt{\log x}\sqrt{\log\log x})$.)

Note that we are interested in algorithms with a provable running time only (possibly under the Riemann hypothesis in the number field case), whence we do not take into account algorithms of the number field sieve type. Such algorithms are conjectured to be asymptotically faster than ours (and there are both plausible theoretical considerations and numerical evidence to support this conjecture). In any case, they are only available for finite fields and not for class groups.

The assumption that the group is cyclic and of known order is no restriction from a cryptographic point of view, as knowledge of the group order is required for digital signatures and to prove resistance against the Pohlig–Hellman attack. However, the discrete logarithm problem often has to be solved in a cyclic subgroup, whereas the smoothness assumption can only be verified for the full group, which itself may not be cyclic. We show that a variant of the algorithm applies to cryptographically suitable subgroups. Moreover, the modifications allow to detect the case that no discrete logarithm exists, in which the original algorithm would run forever.

We proceed as follows. In Section 2 we introduce the general setting in which the algorithm described in Section 3 can be used to compute discrete logarithms. The subsequent sections are devoted to the analysis of the algorithm. We extend some results on sparse linear algebra to evaluate the probability that the algorithm succeeds and verify its subexponential running time. Along the way, the general considerations are specialised to classical examples. Finally, in Section 7 we explain how to adapt the algorithm for the cases where the group is not cyclic.

2 Notations and prerequisites

Throughout the paper, we denote by $\mathbb{Z}$ the set of integers, by $\mathbb{N}$ the set of positive integers, by log the natural and by $\log_2$ the dual logarithm. Let G be an additively written cyclic group of known order N, and let $\mathbb{Z}_N$ denote $\mathbb{Z}/N\mathbb{Z}$. In general, one would expect that the elements of G are represented by $O(\log N)$ bits and measure algorithmic complexities as functions of N. It is, however, possible to construct groups whose elements are naturally represented by bit strings of length in $O(\log N')$ for some value N' considerably larger than N. For instance, Jacobian groups of curves of genus g over $\mathbb{F}_2$ are given by $\Theta(\log N')$ bits for $N' = 2^g$, whereas the only known lower bound on N in this case is the Hasse–Weil bound $(\sqrt{2}-1)^g \leq 1$. While this situation results in a waste of bandwidth for cryptographic applications and is thus unlikely to occur, for preserving as much generality as possible we henceforth denote by $\log N'$ the input size of the problem and measure all complexities by functions in $\log N'$. It turns out that factors polynomial in $\log N'$ do not affect the subexponential running time. Hence to simplify the analysis we follow [GG99] and for some positive function f of N' denote by $O^{\sim}(f)$ the class of functions which are in $O(f)$ up to a factor bounded by some power of $\log N'$.

To apply the index calculus idea, G must behave similarly to the natural integers or the polynomials over a finite field in that each element of G admits a unique decomposition into a "sum of primes". To model this behaviour, assume that there is a free abelian monoid $\mathbb{M}$ over a countable set $\mathcal{P}$, whose elements are called *primes*, together with an equivalence relation $\sim$ on $\mathbb{M}$ which is compatible with its composition law, such that $G \simeq \mathbb{M}/\sim$. Thus, each element of $\mathbb{M}$ has a unique decomposition into a sum of primes. Let each element of G be represented by a unique element of $\mathbb{M}$ of bit size in $O^{\sim}(1)$, then G inherits the unique decomposition property. We assume that the arithmetic in G (addition, negation and test for equality) is performed by manipulating these unique representatives in time polynomial in $\log N'$. Furthermore, we assume that there is a homomorphism of monoids $\deg : \mathbb{M} \to \mathbb{R}^+$, which to each element of $\mathbb{M}$ (and thus of G) associates its "*size*". As $\mathbb{M}$ is free over $\mathcal{P}$, any such homomorphism is given by assigning a non-negative size to each prime and extending additively. Usually the "size" $\deg(m)$ and the bit size of m are closely related in the sense that the latter is in $\Theta(\deg(m))$.

The setting above was introduced by Knopfmacher in [Kno75], who calls $\mathbb{M}$ an *additive arithmetical semigroup* and G an *arithmetical formation*. In addition we require that $\deg(p) \geq 1$ for $p \in \mathcal{P}$ and $\deg(g) \in O^{\sim}(1)$ for any $g \in G$. This ensures that the number of primes in the decomposition of a group element, counting multiplicities, is bounded above by $O^{\sim}(1)$.

For a *smoothness bound* $S \in \mathbb{N}$ denote by $\mathcal{P}_S$ the set of primes of size at most S, by n_S the cardinality of $\mathcal{P}_S$ and by n'_S the number of elements of $\mathbb{M}$ of size at most S. From an algorithmic point of view, we demand that n'_S be finite, that the elements of size at most S can be enumerated in time polynomial in S and linear in n'_S and that an element $m \in \mathbb{M}$ can be tested for being prime in time polynomial in $\deg(m)$ and linear in $n'_{\deg(m)}$ (by trial division by all elements of size smaller than $\deg m$, for instance). Thus, $\mathcal{P}_S$ can be constructed in time polynomial in S and quadratic in n'_S. In practice, Eratosthenes's sieve could be used to lower the complexity in n'_S; however, the algorithms studied below are at least quadratic in n'_S.

An element of G is called *S–smooth* if its decomposition involves only primes of $\mathcal{P}_S$. As the size of the elements of G is in $O^{\sim}(1)$, a distinction into smooth and non-smooth elements arises only for $S \in O^{\sim}(1)$. For technical reasons we assume furthermore that $\log n_S \in O^{\sim}(1)$. We require that elements of G can be tested for S–smoothness and, if possible, be decomposed into a sum of primes from $\mathcal{P}_S$ in $O^{\sim}(n_S)$, which usually amounts to trial division by the elements of $\mathcal{P}_S$. In all cases considered below, the smoothness test and the decomposition are even in $O^{\sim}(\sqrt{n'_S})$ or $O^{\sim}(1)$, which results in better running times.

The most efficient smoothness test available for integers to date, which is subexponential in $\log n'_S$, is non-deterministic and not completely reliable in that it may not recognise a smooth element. Thus, we extend our model as follows: The smoothness test rejects all non-smooth elements; it recognises a smooth element up to a certain error probability, which may depend on the element tested, but does not exceed $1/2$.

It should be noted that we could work with a more general definition of smoothness, which does not involve the notion of the size of an element. Also, unique decomposability could be defined in an abstract way, not involving the notion of a free abelian monoid. However, the more intuitive definitions apply to all groups considered in the literature so far.

Examples.

1. Finite prime fields $G = \mathbb{F}_p^\times$
   Then G can be represented as $(\mathbb{N}, \cdot)/\sim$, where $m_1 \sim m_2$ if and only if $p | m_1 - m_2$, and $\mathcal{P}$ is

the set of natural prime numbers. The size of an element is given by its logarithm to the base 2, $\deg(m) = \log_2 m$, and $N' = N = p - 1$.

2. Finite fields of characteristic 2, $G = \mathbb{F}_{2^k}^\times$
Then G can be represented as $(\mathbb{F}_2[X]\backslash\{0\}, \cdot)/\sim$, where $f_1 \sim f_2$ if and only if $f | f_1 - f_2$ for some fixed irreducible polynomial f of degree k in $\mathbb{F}_2[X]$, and $\mathcal{P}$ is the set of irreducible polynomials over $\mathbb{F}_2$. The size of an element is given by its usual degree, and $N' = N = 2^k - 1$.

3. Finite fields of the form $G = \mathbb{F}_{p^k}^\times$, p prime
Then G can be represented by the polynomials of degree less than k over $\mathbb{F}_p$. Denote by $\mathbb{F}_p[X]'$ the set of monic polynomials over $\mathbb{F}_p$. Noticing that any polynomial is the unique product of its leading coefficient and a monic polynomial, G can be represented as $(\mathbb{N}, \cdot) \times (\mathbb{F}_p[X]', \cdot)/\sim$, where $(m_1, f_1) \sim (m_2, f_2)$ if and only if $p | m_1 - m_2$ and $f | f_1 - f_2$ for some fixed irreducible polynomial f of degree k over $\mathbb{F}_p$. The set of primes $\mathcal{P}$ is given by the union of the set of natural primes and the set of monic irreducible polynomials over $\mathbb{F}_p$, each embedded into the cartesian product. The size of an element is $\deg(m_1, f_1) = \log_2 m_1 + \deg f_1$, and $N' = N = p^k - 1$. Notice that these definitions are compatible with Examples 1. and 2.

4. Class groups of number fields
Let K be a number field and $\mathcal{O}$ its ring of integers. Then the class group G of K is defined as $\mathbb{M}/\sim$, where $\mathbb{M}$ is the set of ideals of $\mathcal{O}$ (a free abelian monoid over the set $\mathcal{P}$ of prime ideals), and $\sim$ is induced by the submonoid of principal ideals. The size of an ideal is given by the logarithm of its norm. If K has unit rank 0, then each ideal class contains a unique so-called reduced ideal, which can be computed in polynomial time from any representative of the class as long as $(K : \mathbb{Q})$ is fixed. This reduced ideal then constitutes the canonic representative for the class.

   In particular, the case of imaginary quadratic fields $\mathbb{Q}(\sqrt{D})$ of discriminant $D < 0$ is covered. Let $\omega = \frac{D+\sqrt{D}}{2}$ with minimal polynomial $X^2 - DX + \frac{D^2-D}{4}$ be an element generating an integral power basis. Then $\mathcal{P}_S$ can be constructed by enumerating all rational primes $p \leq 2^S$ and solving the equation $y_p^2 - Dy_p + \frac{D^2-D}{4} \equiv 0 \pmod{p}$, which can be done in expected polynomial time by a probabilistic algorithm. If this equation does not have a solution, then p is inert and $p\mathcal{O}$ is a principal prime ideal, whence it may be omitted from the factor base. If the equation has the double solution $y_p = 0$, then p is ramified and (p, ω) is the only prime ideal above p in $\mathcal{O}$. Finally, if the equation has two solutions y_p and $\overline{y}_p$, then p is splitting and $\mathfrak{p} = (p, \omega - y_p)$ and $\overline{\mathfrak{p}} = (p, \omega - \overline{y}_p)$ are the two prime ideals above p in $\mathcal{O}$.

   A reduced ideal $\mathfrak{a} = (a, \omega - b)$ with $a | b^2 - Db + \frac{D^2-D}{4}$ is S–smooth if and only if all prime divisors of a are bounded above by 2^S. Let p be a prime divisor of a and ν the exponent of p in a. Then, as the ideal is reduced, it can be shown that p is not inert, so there is an ideal of the form $\mathfrak{p} = (p, \omega - y_p)$ above p in $\mathcal{O}$. If $y_p \equiv b \pmod{p}$, then the ideal occurs with multiplicity ν in the decomposition of $\mathfrak{a}$, otherwise, $\overline{\mathfrak{p}}$ occurs with multiplicity ν. Thus, the smoothness test and the decomposition of class group elements into primes is completely reduced to the same problems over the rational integers. Again, we let $N' = N$.

5. Jacobians of curves over finite fields
Let $C \in \mathbb{F}_q[X, Y]$ be a plane irreducible projective curve, and G its Jacobian. Fix a divisor $\mathcal{O}$ of degree 1, and let $\mathcal{P}$ denote the set of all $P - (\deg P)\,\mathcal{O}$ with a prime divisor P. The size of such an element of $\mathcal{P}$ is given by $\deg P$. Then $G \simeq \mathbb{M}/\sim$, where $\mathbb{M}$ is the free monoid over $\mathcal{P}$ and $\sim$ is induced by the submonoid of principal divisors. If the prime at infinity of $\mathbb{F}_q[X]$

is totally ramified in the function field of the curve, we may choose $\mathcal{O}$ as the prime divisor at infinity, and the Jacobian is isomorphic to the ideal class group of the integral closure of $\mathbb{F}_q[X,Y]$ in the function field. Of particular interest is the case of hyperelliptic curves, which constitute the function field analogue of quadratic number fields. A hyperelliptic curve of genus g over $\mathbb{F}_q$ is the smooth projective model of a plane curve of the form

$$H = Y^2 + hY - f$$

with $h \in \mathbb{F}_q[X]$ of degree at most g and $f \in \mathbb{F}_q[X]$ monic of degree $2g+1$ or $2g+2$.

If $\deg f = 2g+1$, the prime at infinity is ramified and the representation of the hyperelliptic curve by H is called *imaginary*. The use of such curves in cryptography has been suggested by Koblitz in [Kob89]. Each divisor class of a hyperelliptic Jacobian in imaginary representation contains a unique reduced divisor $\mathrm{div}(a,b)$ which corresponds to the ideal $(a, Y-b)$ of $\mathbb{F}_q[X,Y]/(H)$ and satisfies a, $b \in \mathbb{F}_q[X]$, $\deg b < \deg a \leq g$ and $a | b^2 + bh - f$. Its size is $\deg a$. So the input size of the problem is $O(\log N')$ for $N' = q^g$. The assumption that $N \in O^\sim(N')$ holds since $N \leq (2g+1)q^g$. For q a prime, this bound is due to Artin ([Art24], §24, Formula (8)), whose arguments are easily extended to the general case replacing the Artin character by the general quadratic character. Given a divisor $\mathrm{div}(a', b')$, the canonical reduced representative in its class can be computed in time polynomial in N'. The set $\mathcal{P}_S$ can be constructed by enumerating all irreducible polynomials $p \in \mathbb{F}_q[X]$ of degree at most S and solving the equation $y_p^2 + hy_p - f \equiv 0 \pmod{p}$ in expected polynomial time by a probabilistic algorithm. All further steps are completely analogous to the case of imaginary quadratic number fields. The only difference is that testing for smoothness and decomposing a group element into primes is reduced to the corresponding problems over the univariate polynomials instead of the rational integers.

For $\deg f = 2g+2$, the prime at infinity is splitting and H is called a *real* representation of the curve. Its Jacobian is no longer isomorphic to the ideal class group of $\mathbb{F}_q[X,Y]/(H)$. Instead of working in the Jacobian, it has been suggested to base cryptosystems on discrete logarithms in the so called *infrastructure* of the curve [SSW96, MVZ98]. It has been observed in [PR99] that at least in odd characteristic a constant field extension of degree at most $2g+2$ allows to transform a real into an equivalent imaginary representation. Thus we restrict our presentation to imaginary curves.

3 Algorithm

We describe the algorithm for a cyclic group whose order is known and not necessarily prime. Unlike in the Pohlig–Hellman method we do not split the discrete logarithm problem into a series of problems in subgroups of prime order. In fact, we need to work in the full group to guarantee a provable proportion of smooth elements, and the generalisation of the algorithm to subgroups poses challenges which are discussed in Section 7. However, we factor the group order to facilitate the linear algebra step.

The algorithm takes as input a generator g_1 of a cyclic group G of order N and another element $g_2 \in G$. It outputs $\log_{g_1} g_2$, i.e. an integer $l \in \{0, \ldots, N-1\}$ such that $g_2 = lg_1$.

1) Choose a smoothness bound S and construct the *factor base* $\mathcal{P}_S = \{p_1, \ldots, p_n\}$ with $n = n_S$. Set $k = \lceil \log_2 n + \log_2 \log_2 N \rceil + 1$.

2) Construct a matrix $A = (a_{ij}) \in \mathbb{Z}_N^{n\times(2kn)}$ as follows: For $j = 1, \dots, kn$, select randomly and uniformly α_j, $\beta_j \in \mathbb{Z}_N$ until $\alpha_j g_1 + \beta_j g_2$ is S–smooth, and write

$$\alpha_j g_1 + \beta_j g_2 = \sum_{i=1}^{n} a_{ij} p_i. \tag{1}$$

For $j = kn+1, \dots, 2kn$, write $j = (k+l)n + m$ with $0 \le l \le k-1$, $1 \le m \le n$, and select randomly and uniformly α_j, $\beta_j \in \mathbb{Z}_N$ until $\alpha_j g_1 + \beta_j g_2 - p_m$ is S–smooth; then write

$$\alpha_j g_1 + \beta_j g_2 = p_m + \sum_{i=1}^{n} b_{ij} p_i = \sum_{i=1}^{n} a_{ij} p_i. \tag{2}$$

(In practice, one would only create a few more than n columns of the first type, see [Gau99]. The second type of columns and the constant k are merely needed to make a rigorous proof of the running time possible.)

3) By the randomised procedure described in Section 4, try to find a non-zero vector $\gamma = (\gamma_1, \dots, \gamma_{2kn}) \in \mathrm{Ker}(A)$. (During this step N is factored.) If the procedure fails, return to 2).

4) If $\sum_{j=1}^{2kn} \beta_j \gamma_j$ is invertible in $\mathbb{Z}_N$, then output $-\left(\sum_{j=1}^{2kn} \beta_j \gamma_j\right)^{-1} \left(\sum_{j=1}^{2kn} \alpha_j \gamma_j\right)$; otherwise return to 2).

If the algorithm halts in Step 4), then it outputs the correct discrete logarithm of g_2 to the base g_1. The fact that $\gamma \in \mathrm{Ker}(A)$ means that

$$0 = \sum_{j=1}^{2kn} a_{ij} \gamma_j \quad \forall i = 1, \dots, n;$$

multiplying these equations by p_i and summing them up yields

$$0 = \sum_{j=1}^{2kn} \left(\sum_{i=1}^{n} a_{ij} p_i \right) \gamma_j = \left(\sum_{j=1}^{2kn} \alpha_j \gamma_j \right) g_1 + \left(\sum_{j=1}^{2kn} \beta_j \gamma_j \right) g_2.$$

As g_1 and g_2 are both of N–torsion, multiplying by the inverse of $\sum_{j=1}^{2kn} \beta_j \gamma_j$ in $\mathbb{Z}_N$, if it exists, shows the correctness of the result.

4 Linear algebra

As $\operatorname{rank} A \le n$, it is possible to find a non-zero vector $\gamma \in \mathrm{Ker}(A)$. How this is done, however, needs further explication. On one hand, it is desirable to exploit the sparse structure of the matrix, which has only $O^{\sim}(1)$ entries per column, and the corresponding algorithms are prone to failure with a certain probability. On the other hand, a complication is introduced by the fact that N need not be prime, so that $\mathbb{Z}_N$ may not be a field.

To exploit the matrix sparseness, one may use a randomised Lanczos algorithm; we rely on the following trivial corollary of Theorem 6.2 in [EK97].

Theorem 2 *Let $\mathbb{F}_q$ be the finite field with q elements and let $A \in \mathbb{F}_q^{n\times d}$ be a matrix of rank r with ω non-zero entries and $b \in \mathbb{F}_q^n$. There is a probabilistic algorithm which either returns a vector $x \in \mathbb{F}_q^d$ such that $Ax = b$ or reports failure. The algorithm requires $O(r(\omega + d))$ operations in $\mathbb{F}_q$ and has a failure probability of at most $\frac{11d^2-d}{2(q-1)}$.*

Moreover, the solution vector returned by the algorithm can be made to vary uniformly over all possible solutions by randomising the right hand side in the following way (in fact, this randomisation is already part of the algorithm in [EK97]): Choose $y \in \mathbb{F}_q^d$ according to a uniform distribution, solve $A\overline{x} = b + Ay$ and let $x = \overline{x} - y$. If y varied over a fixed class of $\mathbb{F}_q^d / \operatorname{Ker} A$, then $\overline{x}$ would not depend on y, and x would be distributed uniformly over the solution space $\overline{x} + \operatorname{Ker} A$ of the equation. Hence, the same assertion holds when y does not belong to a fixed class.

When q is small compared to d, it is not possible to apply the theorem directly. Instead, one may switch to a field extension. While this idea does not seem to be new — it was used, for instance, in the implementation of [LO90], see also [KS91] — we did not find it detailed in the literature and thus expand on the topic. In particular, it is possible to maintain the uniform distribution over the solution vectors. In the situation of Theorem 2, let p be the characteristic of $\mathbb{F}_q$, $\nu = \min\{l : q^l > 11d^2, p \nmid l\}$ and $q' = q^\nu$. Then $q^{\nu-2} \leq 11d^2$, so that $q' \in O(d^2q^2)$. We would like to solve a matrix equation over $\mathbb{F}_{q'}$ and project the solution onto a solution $x \in \mathbb{F}_q^d$ of $Ax = b$. For projection, one may use the trace function $\operatorname{Tr} : \mathbb{F}_{q'} \to \mathbb{F}_q$, which is a homomorphism of $\mathbb{F}_q$-vector spaces and acts on $\mathbb{F}_q$ as multiplication by ν. Let $b' = \nu' b \in \mathbb{F}_q^d$ with $\nu\nu' \equiv 1 \pmod p$, so that $\nu b' = b$. The value ν' exists because $\gcd(\nu, p) = 1$ and can be computed by the extended Euclidean algorithm in time $O(\log \nu \log p)$, which as well as the multiplication of b by ν' is negligible compared to the following linear algebra step. Solve $Ax' = b'$ by the algorithm in [EK97]. The success probability for this step is at least $1 - \frac{11d^2-d}{2(q'-1)} \geq \frac{1}{2}$. Let $x = \operatorname{Tr}(x')$. Then from the linearity of the trace we deduce that $Ax = \operatorname{Tr}(Ax') = \operatorname{Tr}(b') = \nu b' = b$. Moreover, any solution $x \in \mathbb{F}_q^d$ of $Ax = b$ can be obtained in this way, and all of them have the same probability of occurring. Namely, for a given solution x, the set of solutions to $Ax' = b'$ over $\mathbb{F}_{q'}^d$ which map to x under the trace function is given by $\nu' x + (\operatorname{Ker} A \cap \operatorname{Ker} \operatorname{Tr})$, whose cardinality $(q')^{\dim(\operatorname{Ker} A \cap \operatorname{Ker} \operatorname{Tr})}$ is independent of x. Thus, we have shown the following result:

Theorem 3 *Let $A \in \mathbb{F}_q^{n\times d}$ be a matrix of rank r with ω non-zero entries and $b \in \mathbb{F}_q^n$. There is a probabilistic algorithm which either returns a vector $x \in \mathbb{F}_q^d$ such that $Ax = b$ or reports failure. The running time of the algorithm is in $O(r(\omega + d) \log^2(dq))$, and its failure probability is at most $\frac{1}{2}$. Moreover, the resulting vector is uniformly distributed over all possible solutions.*

This solves the linear algebra step if N is prime. Otherwise, one factors N, computes γ modulo p^ν for all $p^\nu \| N$ and combines the results by the Chinese Remainder Theorem. The computations modulo p^ν may be broken up into ν iterations modulo p via a lifting procedure: Suppose that a non-zero solution $\gamma_1 \in \{0, \ldots, p^e - 1\}^{2kn}$ is known to the equation $Ax \equiv 0 \pmod{p^e}$, for instance $A\gamma_1 = p^e \delta$ with $\delta \in \mathbb{Z}^{2kn}$. Assume that there is a solution γ_2 of $Ax \equiv \delta \pmod p$. Then $p^e \gamma_2 - \gamma_1$ is a non-zero solution of $Ax \equiv 0 \pmod{p^{e+1}}$. If all computations modulo a prime return a random vector according to a uniform distribution over all possible solutions, then the combined result varies uniformly over the kernel of A.

Considering the elementary divisor form of the matrix A, however, it is easily seen that the lifting procedure may fail if (and only if) $\operatorname{rank}_{\mathbb{Q}} A \neq \operatorname{rank}_{\mathbb{Z}_p} A$ because then the matrix equation $Ax \equiv \delta$

(mod p) need not have a solution. This is the reason why, following [Pom87], we create the matrix A in a special way, generating many more than the $n+1$ columns one would expect to need in practice and introducing the canonical basis elements p_m into the matrix. Indeed, it is proved in Lemma 4.1 and the subsequent remark of [Pom87] that with high probability the matrix has full rank over $\mathbb{Z}_p$. We recall this lemma with our notations.

Lemma 4 *Let V be a vector space over a field $\mathbb{F}$ with $\dim V = n < \infty$. Let $\mathcal{S}$ be a finite set of vectors in V and $b_1, \dots, b_n$ a basis for V. Let $k \in \mathbb{N}$. We make $2kn$ independent choices of elements from $\mathcal{S}$ with an arbitrary probability distribution over $\mathcal{S}$, labelling the chosen vectors $v_1, \dots, v_{kn}, w_1, \dots, w_{kn}$, and we denote by V' the subspace of V spanned by $v_1, \dots, v_{kn}$, and the vectors $b_j + w_{(j-1)k+i}$ for $j = 1, \dots, n$ and $i = 1, \dots, k$. Then with probability at least $1 - \frac{n}{2^{k-1}}$ we have $V = V'$.*

In our case, the vector space V is the space of column vectors of size n with coefficients in $\mathbb{Z}_p$, the basis is the canonical basis, and the set $\mathcal{S}$ is the set of all column vectors representing a smooth element of G. We see that the vectors generating V' correspond precisely to the vectors forming the matrix A. Hence the probability that the lifting is possible on $\mathbb{Z}_p$ is at least $1 - \frac{n}{2^{k-1}}$. There are at most $\frac{\log_2 N}{2}$ distinct primes p whose squares divide N, thus the probability that the lifting is possible for all of them is at least $1 - \frac{n \log_2 N}{2^k} \geq \frac{1}{2}$ for our choice of k. In this case, repeating $\log_2(2\log_2 N)$ times the algorithm of Theorem 3, we obtain a solution of one problem modulo a prime with probability at least $1 - \frac{1}{2^{\log_2(2\log_2 N)}} = 1 - \frac{1}{2\log_2 N}$. As at most $\log_2 N$ single problems have to be solved, we get a solution modulo N with probability at least $\frac{1}{2}$. Altogether, Step 3) is thus successful with a probability of at least $\frac{1}{4}$, in which case the output vector is uniformly distributed over the kernel.

5 Success probability and running time

To estimate the success probability of the algorithm during one run of Steps 2) to 4), we assume that Step 2) has been accomplished successfully, the study of this step being postponed to the running time analysis below.

As shown above, Step 3) succeeds with probability at least $\frac{1}{4}$. The algorithm may also fail if $\sum_{j=1}^{2kn} \beta_j \gamma_j$ is not invertible in $\mathbb{Z}_N$ in Step 4). However, this happens with a sufficiently low probability. For given $j \leq kn$ and any β_j, as g_1 is a generator of G and α_j is uniformly distributed, the element $\alpha_j g_1 + \beta_j g_2$ is uniformly distributed over all group elements. The same holds for $j > kn$ and $\alpha_j g_1 + \beta_j g_2 - p_m$. Consequently, the matrix A and the vector β are independent random variables, so that γ and β are also independent. Let p be a prime divisor of N. As γ is uniformly distributed over all vectors of the kernel, the probability that $\gamma \not\equiv 0 \pmod p$ is at least $1 - \frac{1}{p}$. Then the orthogonal space of γ mod p in $\mathbb{Z}^{2kn}$ has dimension $2kn - 1$, and the conditional probability that β mod p is not orthogonal to γ mod p is at least $1 - \frac{1}{p}$. Hence $\sum_{j=1}^{2kn} \beta_j \gamma_j$ is invertible in $\mathbb{Z}_N$ with probability at least $\prod_{p|N} \left(1 - \frac{1}{p}\right)^2 = \left(\frac{\varphi(N)}{N}\right)^2$. From (3.41) in [RS62] we have $\frac{\varphi(N)}{N} \in \Omega(1/\log\log N)$.

Thus, the total success probability for one run of Steps 2) to 4) is in

$$\Omega\left(\frac{1}{(\log\log N)^2}\right).$$

In accordance with Section 2, denote by $n' = n'_S$ the number of elements of the monoid $\mathbb{M}$ whose sizes are bounded above by S.

With the assumptions set forth in Section 2, $\mathcal{P}_S$ can be constructed in time $O^\sim(n'^2)$.

Denote by N_S the number of S–smooth elements of G, and let t_s and t_d be upper bounds on the expected time needed for a smoothness test and the decomposition of a smooth group element into a sum of primes, respectively. The time needed for computing one linear combination of g_1 and g_2 and testing for smoothness is in $O^\sim(t_s)$; this has to be repeated an expected $\frac{N}{N_S}$ times until a smooth element is obtained. This smooth element is recognised with a probability of at least $1/2$, so that no more than two repetitions of the previous procedure are needed on average until a column of the matrix can be filled. So the total time used in Step 2) is in

$$O^\sim\left(n\left(\frac{N}{N_S}t_s + t_d\right)\right) \subseteq O^\sim\left(n\frac{N}{N_S}t_s + n^2\right)$$

as $2kn$, $t_d \in O^\sim(n)$.

Let t_f be a time bound for factoring N. As explained in Section 4, $\log_2(2\log_2 N)\log_2 N \in O^\sim(1)$ executions of the algorithm behind Theorem 3 are required for Step 3). The number of entries in each column of A is in $O^\sim(1)$, so that Step 3) needs time in $O^\sim(t_f + n^2)$.

Finally, Step 4) can be performed in $O^\sim(n)$.

As only $O((\log\log N)^2) \subseteq O^\sim(1)$ repetitions of Steps 2) to 4) are needed on average and $n \le n'$, the total running time of the algorithm is in

$$O^\sim\left(t_f + n'^2 + n'\frac{N}{N_S}t_s\right). \tag{3}$$

(In all cases under consideration, n and n' differ only by polynomial factors in $\log N'$, i.e. $O^\sim(n) = O^\sim(n')$, so that we do not lose anything when replacing n by n'.)

Examples.

1. $G = \mathbb{F}_p^\times$, p prime
   With deterministic algorithms due to Pollard and Strassen [Pol74, Str76] we have $t_s \in O^\sim(\sqrt{n'})$. A more efficient probabilistic method has been proved using hyperelliptic curves. The test of [LPP93] recognises (and decomposes) a smooth number with probability at least $1/2$ in time $t_s \in O^\sim(L_{n'}(2/3, c))$, where L is the subexponential function as defined in Section 6 and c some positive constant. Thus, the total running time is in
   $$O^\sim\left(t_f + n'^2 + n' L_{n'}(2/3, c)\frac{N}{N_S}\right).$$

2. $G = \mathbb{F}_{2^k}^\times$
   Now $t_s \in O^\sim(1)$, as a smoothness test can be performed in deterministic polynomial time by computing the distinct degree factorisation of the polynomial representing the group element. Precisely, let $f \in \mathbb{F}_2[X]'$ be the element to be tested, and $g = \frac{f}{\gcd(f,f')}$ its square-free part.

Then f is S–smooth if and only if g is. As $X^{2^i} - X$ is the product of all irreducible polynomials of degree dividing i in $\mathbb{F}_2[X]'$, the latter is the case if and only if

$$g = \operatorname{lcm}\left(\left\{\gcd(g, X^{2^i} - X) \, : \, i = 1, \dots, S\right\}\right).$$

Computing $X^{2^i} - X \bmod g$ by successive squaring and reduction modulo g, this can be tested in time polynomial in S and $\deg f \in O(\log N)$. Thus, the total running time of the algorithm is in

$$O^{\sim}\left(t_f + n'^2 + n' \frac{N}{N_S}\right).$$

3. $G = \mathbb{F}_{p^k}$, p prime
An element $(m, f) \in \mathbb{N} \times \mathbb{F}_p[X]'$ is S–smooth if and only if m and f are S–smooth. The smoothness of m can be tested in time in $O^{\sim}(L_p(2/3, c))$ as mentioned in Example 1. The smoothness of f can again be checked by distinct degree factorisation in time $O^{\sim}(1)$. Thus, the total running time of the algorithm is in

$$O^{\sim}\left(t_f + n'^2 + n' L_p(2/3, c) \frac{N}{N_S}\right).$$

4. Class groups of imaginary quadratic number fields
As the smoothness test and the decomposition into primes are reduced to the case of natural integers, the analysis of Example 1. shows that the running time is in

$$O^{\sim}\left(t_f + n'^2 + n' L_{n'}(2/3, c) \frac{N}{N_S}\right).$$

5. Jacobians of hyperelliptic curves
Now the smoothness test and the decomposition are reduced to the case of monic polynomials, and the analysis of Example 2. carries over and shows that the running time is in

$$O^{\sim}\left(t_f + n'^2 + n' \frac{N}{N_S}\right).$$

6 Subexponentiality

Recall the definition of the subexponential function with respect to the input size $\log N'$ and parameters $\alpha \in (0, 1)$ and $c > 0$:

$$L_{N'}(\alpha, c) = e^{c(\log N')^{\alpha} (\log \log N')^{1-\alpha}}.$$

The smaller α, the closer this function is to the polynomial $L_{N'}(0, \lceil c \rceil) = (\log N')^{\lceil c \rceil}$ in $\log N'$. All rigorously proven subexponential algorithms for discrete logarithms, and also the algorithm of this article, have $\alpha = 1/2$. Thus we simplify the notation by omitting the first parameter when it is $1/2$. We state the simple relations

$$L_{N'}(c_1) \cdot L_{N'}(c_2) = L_{N'}(c_1 + c_2)$$

and

$$L_{N'}(c_1) + L_{N'}(c_2) \in \Theta(L_{N'}(\max(c_1, c_2))).$$

Furthermore, functions in $O^{\sim}(1)$ or $O(L_{N'}(\alpha, c))$ for $\alpha < 1/2$ are in $L_{N'}(o(1))$, where $o(1)$ stands for the set of real valued functions tending to zero as $N' \to \infty$.

Assume that we have a smoothness result of the following form:

The bound S can be chosen such that

$$n' \in O(L_{N'}(\rho + o(1)))$$

and

$$\frac{N}{N_S} \in O(L_{N'}(\sigma + o(1)))$$

for some constants ρ, $\sigma > 0$.

The assumption $N \in O^{\sim}(N')$ implies that $L_N(c) \in O(L_{N'}(c) + o(1))$. Taking into account that N can be factored in expected time in $O(L_N(1+o(1))) \subseteq O(L_{N'}(1+o(1)))$ by the algorithm presented in [LP92] and introducing an exponent τ such that $t_s \in O^{\sim}(n'^{\tau})$, we can specialise (3) to obtain

$$O\left(L_{N'}\left(\max(1, 2\rho, (1+\tau)\rho + \sigma) + o(1)\right)\right).$$

In fact, the constants for all examples presented below are worse than 1 anyway, so that the need for factoring N has no influence on our running time bounds.

Examples.

1. $G = \mathbb{F}_p^\times$, p prime; $N' = N = p - 1$
   With the usual notation $\psi(x, y)$ for the number of integers between 1 and x all prime factors of which are not larger than y, we have $\frac{N}{N_S} = \frac{N}{\psi(N, 2^S)}$. Let $S = \lceil \log(L_N(\rho)) \rceil$, so that $n' = 2^S \in [L_N(\rho), 2L_N(\rho)]$. Then Lemma 3.1 in [Pom87] shows that $\frac{N}{N_S} \in O(L_N(\sigma + o(1)))$ with $\sigma = \frac{1}{2\rho}$. Moreover from $n' \in O(L_N(\rho))$ we deduce $L_{n'}(2/3, c) \in L_N(o(1))$. The running time of the algorithm is thus in

   $$O\left(L_N\left(\max\left(2\rho, \rho + \frac{1}{2\rho}, 1\right) + o(1)\right)\right)$$

   for any $\rho > 0$; the optimal choice $\rho = 1/\sqrt{2}$ yields a running time in

   $$O(L_N(\sqrt{2} + o(1))).$$

   This is precisely the complexity of the fastest known algorithm described in [Pom87].

2. $G = \mathbb{F}_{2^k}^\times$; $N' = N = 2^k - 1$
   Denote by $N_q(d, m)$ the number of monic polynomials of degree d over $\mathbb{F}_q$ all prime factors of which have degree at most m. Then $\frac{N}{N_S} \leq \frac{2^k}{N_2(k-1, S)}$. Let $S = \lceil \log(L_N(\rho)) \rceil$, so that $n' \in \Theta(L_N(\rho))$. Theorem 2.1 in [BP98] shows that $N_2(k-1, S) \in \frac{2^{k-1}}{u^{(1+o(1))u}}$ for $u = \frac{k-1}{S} \leq$

$\frac{1}{\rho}\sqrt{\frac{\log N}{\log\log N}} \le \frac{1}{\rho}\sqrt{\log N}$. Thus, a few computations reveal that $\frac{N}{N_S} \in O(L_N(\sigma + o(1)))$ for $\sigma = \frac{1}{2\rho}$, and the running time of the algorithm is in $O\left(L_N\left(\max\left(2\rho, \rho+\frac{1}{2\rho}, 1\right) + o(1)\right)\right)$ for any $\rho > 0$. The optimal choice $\rho = 1/\sqrt{2}$ again yields a running time in

$$O(L_N(\sqrt{2} + o(1))),$$

which corresponds to the fastest known algorithms described in [Pom87] and [BP98].

3. $G = \mathbb{F}_{p^k}$, p prime; $N' = N = p^k - 1$

Notice first that in the polynomial representation we have chosen, it is impossible to obtain a subexponential running time for fixed $k \ge 2$ and $p \to \infty$. If we let $S = 0$, then only the constants have a chance of being smooth, and $\frac{N}{N_S} \ge \frac{p^k-1}{p-1} \ge p^{k-1}$ is exponential in N. If $S \ge 1$, then all p monic linear polynomials are contained in the factor base, which is thus of exponential size. Hence, we must restrict our attention to instances in which p is sufficiently small compared to k.

Letting $S = \lceil \log_p(L_N(\rho)) \rceil$, we obtain the estimate

$$\frac{N}{N_S} \le \frac{p}{\psi(p-1, 2^S)} \cdot \frac{p^{k-1}}{N_p(k-1, S)} \in O\left(pL_N\left(\frac{1}{2\rho} + o(1)\right)\right)$$

as in Example 2, which introduces an unwanted factor of p. Moreover, since we have to round up S, it need not be true any more that $n' \in O(L_N(\rho))$. In fact,

$$\begin{aligned}
n' &= \sum_{i=0}^{S} \left|\{f \in \mathbb{F}_p'[X] : \deg f = i\}\right| \cdot |\{m \in \{1, \ldots, p-1\} : \log_2 m \le S - i\}| \\
&= \sum_{i=0}^{S} p^i \min\{p-1, 2^{S-i}\} \\
&\le \sum_{i=0}^{S-1} p^{i+1} + p^S \\
&\in O\left(p^S\right) \\
&\subseteq O(pL_N(\rho)).
\end{aligned}$$

For a first special result we consider the case $p \in O(\log N)$ or more generally $p \in O^{\sim}(1)$, which implies $n' \in O^{\sim}(L_N(\rho))$ and $L_p(2/3, c) \in L_N(o(1))$. So the running time analysis of Example 2. carries over without modification.

More generally, we must ensure that p is subexponential in $\log N = k \log p$. Following the ideas in [Eng99], we consider the case $k \ge \vartheta \log p$ for some positive constant ϑ, in which $p \le L_N\left(\frac{1}{\sqrt{\vartheta}}\right)$. Then $n' \in O\left(L_N\left(\rho + \frac{1}{\sqrt{\vartheta}}\right)\right)$ and $\frac{N}{N_S} \in O\left(L_N\left(\frac{1}{2\rho} + \frac{1}{\sqrt{\vartheta}} + o(1)\right)\right)$ for the same value of S as above, $L_p(2/3, c) \in L_N(o(1))$ and the total running time is in

$$O\left(L_N\left(\max\left\{2\rho + \frac{2}{\sqrt{\vartheta}}, \rho + \frac{1}{2\rho} + \frac{2}{\sqrt{\vartheta}}, 1\right\} + o(1)\right)\right).$$

The optimal choice for ρ is $\frac{\sqrt{2}}{2}$, which yields a running time of

$$O\left(L_N\left(\sqrt{2} + \frac{2}{\sqrt{\vartheta}} + o(1)\right)\right).$$

Asymptotically for $\vartheta \to \infty$ (e.g., for p fixed), we recover the running time of Example 2.

In [AD93], Adleman and DeMarrais describe an algorithm with conjectured subexponential running time for $p > k$. They represent the field as the ring of integers of a number field modulo a prime ideal. See also [Sem95]. It is an interesting open question whether there is a provably subexponential algorithm for all finite fields using possibly such a representation.

4. Class groups of imaginary quadratic number fields
Let D denote the discriminant of the number field. It is shown in [Sey87], Proposition 4.4, that assuming the generalised Riemann hypothesis, $\frac{N}{N_S} \in O\left(L_{|D|}\left(\frac{1}{4c} + o(1)\right)\right)$ for $S = \lceil \log L_{|D|}(c) \rceil$. Due to a theorem of Siegel's [Sie36], $\log |D| \in (2 + o(1)) \log N$ so that $L_{|D|}(c + o(1)) = L_N(\sqrt{2}c + o(1))$. Letting $S = \lceil \log L_N(\rho) \rceil$, we deduce $n \in O(L_N(\rho))$ and $\frac{N}{N_S} \in O\left(L_N\left(\frac{1}{2\rho} + o(1)\right)\right)$. Repeating the analysis of Example 1 shows that our algorithm has a running time in
$$O\left(L_N\left(\sqrt{2} + o(1)\right)\right) = O\left(L_{|D|}\left(1 + o(1)\right)\right)$$
under the generalised Riemann hypothesis. This improves the running time of $O(L_{|D|}(\sqrt{2} + o(1)))$ of the algorithm described in [HM89] for determining the class group structure without any previous information.

5. Jacobians of hyperelliptic curves
Recall that $N' = q^g$. Letting $S = \lceil \log_q L_{q^g}(\rho) \rceil$, we have $n \leq 2qL_{q^g}(\rho)$. Again, we follow [Eng99] and consider only instances with $g \geq \vartheta \log q$ for some positive constant ϑ, so that $q \leq L_{q^g}\left(\frac{1}{\sqrt{\vartheta}}\right)$. It is shown in [ES00] that then $N_S \geq q^g L_{q^g}\left(-\frac{1}{2\rho}\right)$, so that $\frac{N}{N_S} \in O^{\sim}\left(\frac{N'}{N_S}\right) \subseteq O\left(L_{q^g}\left(\frac{1}{2\rho} + o(1)\right)\right)$, and the running time of the algorithm is in
$$O\left(L_{q^g}\left(\max\left\{2\rho + \frac{2}{\sqrt{\vartheta}}, \rho + \frac{1}{2\rho} + \frac{1}{\sqrt{\vartheta}}, 1\right\} + o(1)\right)\right).$$
A similar analysis as that of Example 3. shows that the optimal choice of ρ is
$$\min\{\overline{\rho}, \rho^*(\vartheta)\} = \min\left\{\frac{\sqrt{2}}{2}, \sqrt{\frac{1}{2} + \frac{1}{4\vartheta}} - \frac{1}{2\sqrt{\vartheta}}\right\} = \sqrt{\frac{1}{2} + \frac{1}{4\vartheta}} - \sqrt{\frac{1}{4\vartheta}},$$
which yields an overall running time of
$$O\left(L_{q^g}\left(\sqrt{2}\left(\sqrt{1 + \frac{1}{2\vartheta}} + \sqrt{\frac{1}{2\vartheta}}\right) + o(1)\right)\right).$$
This improves considerably on the running time of
$$O\left(L_{q^g}\left(\frac{5}{\sqrt{6}}\left(\sqrt{1 + \frac{3}{2\vartheta}} + \sqrt{\frac{3}{2\vartheta}}\right) + o(1)\right)\right)$$
in [Eng99], and asymptotically for $\vartheta \to \infty$ (e.g., for q constant), the constant of the subexponential function is again the same as in Example 2.

Hence, our algorithm shows that cryptosystems based on high-genus hyperelliptic Jacobians are significantly weaker than expected so far, in particular they offer no security gain when compared to cryptosystems in finite fields of comparable size.

7 Cyclic subgroups

Unlike finite fields, many groups of cryptographic interest do not have a cyclic structure. For instance the number of cyclic factors of hyperelliptic Jacobians can be up to twice as large as the genus. Hence it is an important task to compute discrete logarithms in a cyclic subgroup $H = \langle g_1 \rangle$ of a given abelian group G. From a heuristic point of view, this does not pose any problems: The algorithm in its formulation of Section 3 remains applicable. With the usual assumption that smoothness and membership in a subgroup are independent concepts, i.e. the proportion of smooth elements is the same in H as in G, the running time analysis carries over from the group to its subgroup. However, we are concerned with provable running times in this article, and the smoothness results presented so far apply exclusively to the full groups under consideration. In this section, we discuss a few approaches to deal with this situation.

Perturbing with elements of the complement

The simplest situation arises when $\gcd\left(|H|, \frac{|G|}{|H|}\right) = 1$; then H admits a complement H' in G, i.e., $G = H \times H'$. Assume that it is possible to select independently elements h_j of H' according to a uniform distribution in time polynomial in $\log N'$. This is for instance the case if we can select random elements in G, because multiplying a uniformly distributed element of G by $\frac{|G|}{|H|}$ yields a uniformly distributed element of H'. Another favourable situation is the case where we know a basis of H'. (In this context, we understand by a basis of H' a set $\{b_1, \ldots, b_r\}$ such that H' is equal to the direct sum $\langle b_1 \rangle \times \cdots \times \langle b_r \rangle$. The cardinality r of a basis is not an invariant of H', but it is bounded above by $\log_2 |H'|$.)

Then $\alpha_j g_1 + \beta_j g_2 + h_j$ is uniformly and independently of β distributed over G, so that the algorithm may be carried out with these group elements instead of $\alpha_j g_1 + \beta_j g_2$. If it is successful, then

$$\left(\sum_{j=1}^{2kn} \alpha_j \gamma_j\right) g_1 + \left(\sum_{j=1}^{2kn} \beta_j \gamma_j\right) g_2 = -\sum_{j=1}^{2kn} \gamma_j h_j \in H \cap H' = \{0\},$$

so that

$$\log_{g_1} g_2 = \left(\sum_{j=1}^{2kn} \beta_j \gamma_j\right)^{-1} \left(\sum_{j=1}^{2kn} \alpha_j \gamma_j\right)$$

as before. Also, the running time analysis remains unchanged.

While this situation seems to be very special, it is typical for cryptographic applications in which H is supposed to have large prime order and the cofactor $\frac{|G|}{|H|}$ is small, so that $|H|$ and $\frac{|G|}{|H|}$ are automatically coprime. Moreover, for $\frac{|G|}{|H|}$ polynomial in $\log N'$, the structure and, in particular, a basis of $H' \simeq G/H$ can be determined in polynomial time (see, for instance, [Coh93]), and the assumptions of this subsection are satisfied.

Using a basis for G

Assume that a basis $\{b_1, \ldots, b_r\}$ of G along with the orders $e_1, \ldots, e_r$ of its elements are known. Then the discrete logarithm problem can be solved in two steps. Instead of directly writing g_2 as a multiple of g_1 we first express g_1 as a linear combination of the basis elements and then proceed in the same way for g_2. The discrete logarithm can be computed by a few operations modulo the e_i.

In order to write g_1 in terms of the b_i, a slight variation of the algorithm allows to use the smoothness properties. For given $j \leq kn$, pick random elements α_{ij} and β_j until $\sum \alpha_{ij} b_i + \beta_j g_1$ is S–smooth and write this element as $\sum a_{ij} p_i$. Similarly, for $j > kn$ pick random elements until $\sum \alpha_{ij} b_i + \beta_j g_1 - p_m$ is S–smooth. Here again, the elements of G which are tested for smoothness are distributed uniformly and independently of β, so that the same analysis as in Section 5 can be carried out. Hence with high probability a non-zero vector of the kernel is obtained and g_1 is expressed as a linear combination $g_1 = \sum \gamma_i b_i$. The same process yields $g_2 = \sum \delta_i b_i$. Now try to solve the system of modular equations $\delta_i \equiv l\gamma_i \pmod{e_i}$. If this is possible, then l is the correct discrete logarithm of g_2 with respect to g_1. Otherwise, g_2 does not lie in the cyclic subgroup generated by g_1. In this case, which does not occur in the cryptographic setting, the original algorithm of Section 3 would run forever without giving proof of the non-existence of the discrete logarithm. Thus, the ability to detect this case is an additional advantage of the modified algorithm.

8 Conclusions

We have presented a generic probabilistic algorithm for computing discrete logarithm in cyclic groups of known order in which a notion of smoothness is available. Many groups suggested for cryptosystems fit into this context. The running time of the algorithm can be analysed rigorously without any heuristic assumptions. The analysis leads to a subexponential complexity as soon as a certain smoothness assumption is verified. In particular, we recover the running time bounds of the fastest algorithms with proven complexity for finite fields and obtain substantial improvements over the previously known algorithms for class groups. This theoretical result is backed by a recent implementation of the algorithm for hyperelliptic Jacobians, which shows that even curves of a rather small genus are insecure in a cryptographic context [Gau99].

When examining subexponential algorithms, it is common to distinguish between inefficient methods with a provable running time and practical methods with a conjectured running time. Our algorithm breaks with this tradition. It is the fastest known algorithm for the discrete logarithm problem in hyperelliptic Jacobians both in theory and in practice. In fact, the implementation of [Gau99] was the starting point of our study, and while the modifications described in this article are necessary to prove the running time, they do not alter the nature of the algorithm fundamentally.

In general, the groups we are concerned with are not cyclic, but we have shown that suitable modifications allow to apply the algorithm to a wide class of non-cyclic groups, preserving the subexponential running time. Among these are all groups of cryptographic interest. In particular, the analysis of the algorithm remains valid when a basis of G is known. It is thus an interesting question whether a basis can be built if only the group order and its factorisation are known. In the case that the order is square-free, an obvious probabilistic method constructs a basis. If the primes which occur with multiplicity are of polynomial size, they can be dealt with by an exhaustive construction. Considering the remaining case of a power of a large prime, it seems plausible that the notion of smoothness available in the group should allow to construct a probabilistic algorithm for determining the basis.

Acknowledgements

The results of this paper form part of the authors' doctoral theses; we thank our respective supervisors, Dieter Jungnickel and François Morain, for their continuous support and useful suggestions.

References

[AD93] L. M. Adleman and J. DeMarrais. A subexponential algorithm for discrete logarithms over all finite fields. *Math. Comp.*, 61(203):1–15, 1993.

[ADH94] L. M. Adleman, J. DeMarrais, and M.-D. Huang. A subexponential algorithm for discrete logarithms over the rational subgroup of the Jacobians of large genus hyperelliptic curves over finite fields. In L. Adleman and M.-D. Huang, editors, *ANTS-I*, volume 877 of *Lecture Notes in Comput. Sci.*, pages 28–40, 1994. 1st Algorithmic Number Theory Symposium - Cornell University, May 6–9, 1994.

[Art24] E. Artin. Quadratische Körper im Gebiete der höheren Kongruenzen II. *Math. Z.*, 19:207–246, 1924.

[BP98] R. Lovorn Bender and C. Pomerance. Rigorous discrete logarithm computations in finite fields via smooth polynomials. In D. A. Buell and J. T. Teitelbaum, editors, *Computational Perspectives on Number Theory: Proceedings of a Conference in Honor of A.O.L. Atkin*, volume 7 of *Studies in Advanced Mathematics*, pages 221–232. American Mathematical Society, International Press, 1998.

[Buc90] J. Buchmann. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. In C. Goldstein, editor, *Séminaire de Théorie des Nombres, Paris 1988–1989*, Progr. Math., pages 27–41, Boston, 1990. Birkhäuser.

[BW88] J. Buchmann and H. C. Williams. A key-exchange system based on imaginary quadratic fields. *J. Cryptology*, 1:107–118, 1988.

[Coh93] H. Cohen. *A course in algorithmic algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer–Verlag, 1993.

[DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, 22(6):644–655, November 1976.

[Dül91] S. Düllmann. *Ein Algorithmus zur Bestimmung der Klassengruppe positiv definiter binärer quadratischer Formen.* PhD thesis, Universität des Saarlandes, Saarbrücken, 1991. Available at ftp://ftp.informatik.tu-darmstadt.de/pub/TI/reports/duell.diss.ps.gz.

[EK97] W. Eberly and E. Kaltofen. On randomized Lanczos algorithms. In W. W. Küchlin, editor, *ISSAC 97 — Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, pages 176–183. ACM Press, 1997.

[ElG85] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inform. Theory*, 31(4):469–472, July 1985.

[Eng99] A. Enge. Computing discrete logarithms in high-genus hyperelliptic Jacobians in provably subexponential time. Combinatorics and Optimization Research Report CORR 99-04, University of Waterloo, February 1999. To appear in *Math. Comp.*; available at http://-cacr.math.uwaterloo.ca/techreports/1999/corr99-04.ps.

[ES00] A. Enge and A. Stein. Smooth ideals in hyperelliptic function fields. Combinatorics and Optimization Research Report CORR 2000-08, University of Waterloo, January 2000. Available at http://cacr.math.uwaterloo.ca/techreports/2000/corr2000-08.ps.

[Gau99] P. Gaudry. A variant of the Adleman–DeMarrais–Huang algorithm and its application to small genera. Research Report LIX/RR/99/04, LIX, 1999. Available at http://-www.lix.polytechnique.fr/Labo/Pierrick.Gaudry.

[GG99] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, 1999.

[HM89] J. L. Haffner and K. S. McCurley. A rigorous subexponential algorithm for computation of class groups. *J. Amer. Math. Soc.*, 2(4):837–850, 1989.

[Kno75] J. Knopfmacher. *Abstract Analytic Number Theory*, volume 12 of *North-Holland Mathematical Library.* North-Holland Publishing Company, Amsterdam, 1975.

[Kob87] N. Koblitz. Elliptic curve cryptosystems. *Math. Comp.*, 48(177):203–209, January 1987.

[Kob89] N. Koblitz. Hyperelliptic cryptosystems. *J. of Cryptology*, 1:139–150, 1989.

[KS91] E. Kaltofen and B. D. Saunders. On Wiedemann's method of solving sparse linear systems. In H. F. Mattson, T. Mora, and T. R. N. Rao, editors, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 539 of *Lecture Notes in Comput. Sci.*, pages 29–38, Berlin, 1991. Springer-Verlag.

[LO90] B. A. LaMacchia and A. M. Odlyzko. Solving large sparse linear systems over finite fields. In A. J. Menezes and S. A. Vanstone, editors, *Advances in Cryptology*, volume 537 of *Lecture Notes in Comput. Sci.*, pages 109–133. Springer–Verlag, 1990. Proc. Crypto '90, Santa Barbara, August 11–15, 1988.

[LP92] H. W. Lenstra Jr. and C. Pomerance. A rigorous time bound for factoring integers. *J. Amer. Math. Soc.*, 5(3):483–516, 1992.

[LPP93] H. W. Lenstra, Jr., J. Pila, and C. Pomerance. A hyperelliptic smoothness test, I. *Philos. Trans. Roy. Soc. London Ser. A*, 345:397–408, 1993.

[McC89] K. S. McCurley. Cryptographic key distribution and computation in class groups. In R. A. Mollin, editor, *Number Theory and Applications*, volume 265 of *NATO ASI Series C: Mathematical and Physical Sciences*, pages 459–479, Dordrecht, 1989. Kluwer Academic Publishers.

[Mil86] V. S. Miller. Use of elliptic curves in cryptography. In H. C. Williams, editor, *Advances in Cryptology — CRYPTO '85*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426, Berlin, 1986. Springer-Verlag.

[MST99] V. Müller, A. Stein, and C. Thiel. Computing discrete logarithms in real quadratic congruence function fields of large genus. *Math. Comp.*, 68(226):807–822, 1999.

[MVZ98] V. Müller, S. Vanstone, and R. Zuccherato. Discrete logarithm based cryptosystems in quadratic function fields of characteristic 2. *Des. Codes Cryptogr.*, 14(2):159–178, May 1998.

[Odl99] A. Odlyzko. Discrete logarithms: The past and the future. To appear in *Des. Codes Cryptogr.*, 1999.

[Pol74] J. M. Pollard. Theorems on factorization and primality testing. *Proc. Camb. Phil. Soc.*, 76:521–528, 1974.

[Pom87] C. Pomerance. Fast, rigorous factorization and discrete logarithm algorithms. In D. S. Johnson, T. Nishizeki, A. Nozaki, and H. S. Wolf, editors, *Discrete Algorithms and Complexity, Proceedings of the Japan–US Joint Seminar, June 4–6, 1986, Kyoto, Japan*, Perspectives in Computing, pages 119–143, Orlando, 1987. Academic Press.

[PR99] S. Paulus and H.-G. Rück. Real and imaginary quadratic representations of hyperelliptic function fields. *Math. Comp.*, 68(227):1233–1241, 1999.

[RS62] J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois J. Math.*, 6:64–94, 1962.

[Sem95] I. A. Semaev. Computation of discrete logarithms in an arbitrary finite field. *Discrete Math. Appl.*, 5(2):107–116, 1995.

[Sey87] M. Seysen. A probabilistic factorization algorithm with quadratic forms of negative discriminant. *Math. Comp.*, 48(178):757–780, 1987.

[Sie36] C. L. Siegel. Über die Classenzahl quadratischer Zahlkörper. *Acta Arith.*, 1:83–86, 1936.

[SSW96] R. Scheidler, A. Stein, and H. C. Williams. Key-exchange in real quadratic congruence function fields. *Des. Codes Cryptogr.*, 7(1/2):153–174, 1996.

[Str76] V. Strassen. Einige Resultate über Berechnungskomplexität. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 78:1–8, 1976.