J. Symbolic Computation (1999) **27**, 521–534
Article No. jsco.1999.0279
Available online at http://www.idealibrary.com on IDEAL®

The Pohlig–Hellman Method Generalized for Group Structure Computation

EDLYN TESKE

Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

We present a new algorithm that extends the techniques of the Pohlig–Hellman algorithm for discrete logarithm computation to the following situation: given a finite Abelian group and group elements $h, g_1, \dots, g_l$, compute the least positive integer y and numbers $x_1, \dots, x_l$ such that $h^y = \prod g_i{}^{x_i}$. This computational problem is important for computing the structure of a finite Abelian group.

1. Introduction

Let G be a finite Abelian group, written multiplicatively. Let $|G|$ denote the group order, and assume the prime factorization of $|G|$ is known as

$$|G| = \prod_{\substack{p \mid |G| \\ p \text{ prime}}} p^{\eta(p)},$$

with pairwise distinct primes p, and with $\eta(p) \in \mathbb{N}$. Assume that the following *group operations* are possible: for $a, b \in G$ we can compute $c = a * b$, for $a \in G$ we can compute a^{-1}, and for $a, b \in G$ we can test whether $a = b$. As an example, we have in mind the group of points of elliptic curves over a finite field. Other examples are the multiplicative groups of finite fields and class groups of imaginary quadratic orders.

For any subset R of G, denote by $\langle R \rangle$ the subgroup of G generated by R. If $R = \{g\}$, we write $\langle g \rangle$ instead of $\langle \{g\} \rangle$.

Consider the *discrete logarithm problem* (DLP): given two group elements g and h in G, decide whether $h \in \langle g \rangle$. If this is the case, find $\log_g h$, the *discrete logarithm* of h to the base g, i.e. the least non-negative integer x such that $g^x = h$.

If $|G|$ has only small prime factors, there is an efficient algorithm solve the DLP, which was published by Pohlig and Hellman (1978). But if $h \notin \langle g \rangle$, this algorithm only outputs that the equation $h = g^x$ is not solvable. However, there are applications such as group structure computation where one also wants to know the least positive integer y such that $h^y \in \langle g \rangle$ and the least non-negative integer x such that $h^y = g^x$. In this paper, we give an algorithm which uses the Pohlig–Hellman method to find such a solution (y, x). Our algorithm has the advantage that apart from an $O(\log |G|)$ term, its run time is the same as the run time of the Pohlig–Hellman algorithm, regardless of whether y is known in advance or not.

An interesting variant of the DLP is the situation that a set of several group elements serves as a base of the discrete logarithm rather than a single element. By this we mean the following: given group elements h and $g_1, \ldots, g_l$, decide whether h belongs to the group generated by $\{g_1, \ldots, g_l\}$. If this is the case, compute numbers $x_1, \ldots, x_l$ such that $h = {g_1}^{x_1} * \cdots * {g_l}^{x_l}$. We show how the Pohlig–Hellman algorithm can be generalized to this situation.

Both computational problems above are special cases of a problem which we call the *extended discrete logarithm problem* (EDLP). This problem, which we explicitly define further below, arises in the context of group structure computation. By this we mean the following: assume we are given a set $R = \{g_1, \ldots, g_l\}$ that generates the finite Abelian group G, then compute positive integers $m_1, \ldots, m_k$ with $m_1 > 1$, $m_i | m_{i+1}$, $1 \le i \le k$ *and* an isomorphism $\phi : G \to \mathbb{Z}/m_1\mathbb{Z} \times \cdots \times \mathbb{Z}/m_k\mathbb{Z}$, which is given in terms of the images of the generators. Note that this is a much harder computational problem than just to determine the invariants $m_1, \ldots, m_k$. A method to compute the group structure is to compute a set of vectors $\vec{b}_1, \ldots, \vec{b}_l$ in $\mathbb{Z}^l$ such that $\vec{b}_j = (b_{1j}, \ldots, b_{jj}, 0, \ldots, 0)$, $b_{jj} > 0$, and $\prod_{i=1}^{l} {g_i}^{b_{ij}} = 1$, and such that each $\vec{b}_j$ is minimal in the following sense: for each vector $\vec{c} = (c_1, \ldots, c_j, 0, \ldots, 0)$ with $c_j \neq 0$ and $\prod_{i=1}^{l} {g_i}^{c_i} = 1$ we have $|c_j| \ge b_{jj}$. As soon as such a set of vectors is computed, the group structure can be obtained from the Smith normal form of the matrix $(\vec{b}_1, \ldots, \vec{b}_l)$ and the corresponding left transformation matrix. See Buchmann *et al.* (1997) for details.

The Pohlig–Hellman method and our new algorithm require the knowledge of the factorization of the group order. For this we need to know the group order; once the group order is known, its factorization can be computed in expected subexponential time by algorithms such as the elliptic curve method or the quadratic sieve.

An example is the multiplicative group of the finite field $\mathbb{F}_q$, whose group order is $q-1$. Another example is the group of points of an elliptic curve over a finite field, for which we have an algorithm (Schoof, 1985) to compute the group order in polynomial time. Note that the knowledge that any elliptic curve group has only at most two components does not help to compute the isomorphism ϕ. A third example is the class group of an imaginary quadratic order, where we have subexponential algorithms (for example, Hafner and McCurley, 1989) to determine the class number. Of course, the Hafner–McCurley-type algorithms also give the invariants $m_1, \ldots, m_k$ of the class group, but not the isomorphism ϕ.

To define the EDLP, we need some notation. Let R be a subset of the finite Abelian group G. By $\mathbb{Z}^R$ we denote the set of all maps $x : R \to \mathbb{Z}$. For $x, y \in \mathbb{Z}^R$ we define the sum $x + y$ and the product xy componentwise; that is, we set

$$(x + y)(g) = x(g) + y(g) \qquad \text{and} \qquad xy(g) = x(g)y(g), \qquad g \in R.$$

Further, for $x \in \mathbb{Z}^R$ we define the power $R^x \in G$ by

$$R^x = \prod_{g \in R} g^{x(g)}.$$

Now assume that R is such that its elements generate a direct product:

$$\langle R \rangle = \bigotimes_{g \in R} \langle g \rangle.$$

Then for each h in $\langle R \rangle$ there is a uniquely determined map $x \in \mathbb{Z}^R$ such that

$$R^x = h \qquad \text{and} \qquad 0 \le x(g) < \operatorname{ord} g, \qquad g \in R. \tag{1.1}$$

If $h \in G \setminus \langle R \rangle$, there is a smallest positive integer y such that $h^y \in \langle R \rangle$. This leads to the following definition.

Definition 1.1. Let G be a finite Abelian group.

(1) Let $R \subset G$ such that $\langle R \rangle = \bigotimes_{g \in R} \langle g \rangle$, and let $h \in G$. If $h \in R$, then let x be the uniquely determined map in $\mathbb{Z}^R$ such that (1.1) holds. We call x the *extended discrete logarithm (EDL)* of h to the base R. We write

$$x = \log_R h.$$

(2) By the *extended discrete logarithm problem (EDLP)* we mean the following problem: given R and h as in (1), determine the least positive integer y such that $h^y \in \langle R \rangle$ and compute $x = \log_R h^y$. We say that (x, y) is a *solution of the EDLP* for the base (R, h).

The condition that the elements of R form a direct product is necessary for the uniqueness of the representation. In the context of group structure computation this does not mean a restriction. This is because even if the initially given generators do not form a direct product, they can be transformed accordingly in the course of the computation (see Teske, 1998a, for details).

In this paper we give an algorithm to solve the EDLP. This algorithm has the nice property that x and y are computed "simultaneously", which implies that the lack of any *a priori* knowledge about y does not increase the run time. We show that, if combined with the baby-step giant-step method, our algorithm has run time

$$O\left(\max\left\{\left\lceil p^{1/2}\right\rceil^{|R|} \cdot \log_p e_p\right\}\right),$$

where the maximum is taken over all prime divisors of $|G|$, and e_p denotes the *exponent* of the p-Sylow subgroup of G. The exponent of a group is the least integer e such that $g^e = 1$ for all group elements g.

In the next section, we show how to reduce the EDLP to the EDLP in groups of prime power order. In Section 3 we recall the Pohlig–Hellman algorithm and generalize it to the EDLP with a single element as basis. Next, in Section 4 we present the algorithm Solve EDLP; we prove its correctness in Section 5, and the analysis of its run time follows in Section 6.

2. Reduction to Groups of Prime Power Order

Let G be the finite Abelian group generated by $S := \{h\} \cup R$. As before, let

$$|G| = \prod_{\substack{p \mid |G| \\ p \text{ prime}}} p^{\eta(p)}.$$

For each prime divisor p of $|G|$ let

$$H(p) = \frac{|G|}{p^{\eta(p)}}$$

and

$$G(p) = \{g^{H(p)} : g \in G\}.$$

It is immediately clear that $G(p)$ is a subgroup of G, and that a generating set of $G(p)$ is given by

$$S(p) = \{h^{H(p)}\} \cup R(p), \qquad \text{where} \qquad R(p) = \{g^{H(p)} : g \in R\}.$$

Further, if $p_1, \ldots, p_k$ is an enumeration of the pairwise distinct primes p dividing $|G|$, the map

$$G \longrightarrow G(p_1) \times \cdots \times G(p_k), \qquad g \mapsto (g^{H(p_1)}, \ldots, g^{H(p_k)}) \tag{2.2}$$

is an isomorphism of groups. Moreover, for every prime divisor p of $|G|$, the group $G(p)$ has group order $p^{\eta(p)}$ (and therefore is a Sylow p-subgroup of G).

The problem of computing the EDL of h to the base R can be reduced to the problem of computing the EDL of each $h^{H(p)}$ to the base $R(p)$. To see this, let us assume that for each p dividing $|G|$ we already know the solution of the EDLP for the base $(h^{H(p)}, R(p))$. We denote this solution by (x_p, y_p). Put

$$y = \prod_{p||G|} y_p$$

and use the Chinese remainder algorithm to find $x \in \mathbb{Z}^R$ such that

$$x(g) \equiv \frac{x_p(g) \cdot y}{y_p} \bmod p^{\eta(p)}, \qquad p||G|.$$

We claim that (x, y) solves the EDLP for the base (h, R). To show this, we put $a = h^{-y} * R^x$ and note that for each prime divisor p of $|G|$ we have

$$a^{H(p)} = \left(h^{-H(p)y_p} * R(p)^{x_p}\right)^{y/y_p} = 1.$$

Since (2.2) defines an isomorphism of groups, it follows that $a = 1$ and $x = \log_R h^y$. Now let z be the least positive integer such that $h^z \in \langle R \rangle$. Then, since z is the order of $h\langle R \rangle$ in the factor group $G/\langle R \rangle$, we have that z divides y. On the other hand, since $(h^{H(p)})^z \in \langle R(p) \rangle$ for each p dividing $|G|$, we have that each y_p divides z. Hence $y = \prod y_p \le z \le y$, which implies that $y = z$.

Thus, from now on, we only work in groups of prime power order:

$$|G| = p^\eta.$$

For g in G we define the non-negative integer $\sigma(g)$ by

$$\sigma(g) = \min\{\tau \in \mathbb{N}_0 : g^{p^\tau} = 1\}.$$

In other words, $p^{\sigma(g)}$ is the order of $g \in G$. For $R \subset G$ this definition induces a map $\sigma : R \to \mathbb{N}_0$, $g \mapsto \sigma(g)$. Given $g \in G$ and the prime number p such that $|G| = p^\eta$, we compute $\sigma(g)$ simply by looking at the powers g^{p^i}, $i = 0, 1, 2 \ldots$, where we use that $g^{p^{i+1}} = \left(g^{p^i}\right)^p$. Then, if we use the binary method of exponentiation to compute the pth powers, the computation of $\sigma(g)$ takes at most

$$\sigma(g)(2 \log p + 1) \le 3 \log(\operatorname{ord} g) \tag{2.3}$$

group multiplications and $\sigma(g) + 1$ equality checks.

3. The Pohlig–Hellman Algorithm. Solving the EDLP for $|R| = 1$

Let g, $h \in G$ and assume that $\operatorname{ord} g = p^\sigma$. The Pohlig–Hellman algorithm uses the fact that if $h \in \langle g \rangle$, then $x = \log_g h$ can be represented as

$$x = x_0 + x_1 p + \cdots + x_{\sigma-1} p^{\sigma-1}, \tag{3.1}$$

where $0 \leq x_i < p$ for $i = 0, \dots, \sigma - 1$. The coefficients x_i of this base p expansion can be computed iteratively by solving discrete logarithm problems in the group generated by $g^{p^{\sigma}-1}$, which is of order p: the first coefficient, x_0, is determined from the equation $h = g^x$, by raising g and h to the $p^{\sigma-1}$th power,

$$h^{p^{\sigma-1}} = \left(g^{p^{\sigma-1}}\right)^{x_0}.$$

Having computed $x_0, \dots, x_{i-1}$, one can determine x_i $(1 \leq i < \sigma)$ by solving the equation

$$\left(h * g^{-\Sigma_{j=0}^{i-1} x_j p^j}\right)^{p^{\sigma-(i+1)}} = \left(g^{p^{\sigma-1}}\right)^{x_i}.$$

If any of these equations are not solvable, the algorithm terminates with output "$h \notin \langle g \rangle$".

Now we consider the EDLP, for the case $R = \{g\}$. Since the least positive number y such that $h^y \in \langle g \rangle$ is the order of $h\langle g \rangle$ in the factor group $G/\langle g \rangle$, and since G is a group of prime power order, we know that y is of the form $y = p^\tau$ with some $\tau \geq 0$. Thus, we have to find the least value of τ and $0 \leq x < \operatorname{ord} g$ such that $h^{p^\tau} = g^x$. *A priori*, we know that $0 \leq \tau \leq \sigma(h)$. Further, $\tau \geq \sigma(h) - \sigma(g)$, since $p^{\sigma(h)-\tau} = \operatorname{ord} h^{p^\tau} = \operatorname{ord} g^x \leq p^{\sigma(g)}$. Moreover, in the representation (3.1) the coefficients associated with powers p^i for $i < \sigma(g) - \sigma(h) + \tau$ must vanish; this follows from the fact that $g^{xp^{\sigma(h)-\tau}} = 1$. Therefore, we modify the Pohlig–Hellman algorithm as follows: we put

$$\nu = \sigma(h) - \tau$$

and write x in its base p expansion

$$x = x_0 p^{\sigma(g)-\nu} + x_1 p^{\sigma(g)-\nu+1} + \cdots + x_{\nu-1} p^{\sigma(g)-1} = \sum_{i=0}^{\nu-1} x_i p^{\sigma(g)-\nu+i}, \tag{3.2}$$

where $x_0 \neq 0$ if $\nu > 0$, and $0 \leq x_i < p$. By a similar technique as above, we determine recursively the coefficients $x_0, x_1, \dots$. *A priori*, we know that $0 \leq \nu \leq \min\{\sigma(g), \sigma(h)\}$. The more coefficients we succeed in computing, the larger the lower bound for ν, which is strictly increasing, becomes. As soon as the upper bound for ν matches the lower bound, or the computation of a coefficient fails, we can determine ν and τ exactly, as we shall now show.

The least significant coefficient, x_0, is determined by raising g and h to the $p^{\sigma(g)-1}$th and the $p^{\sigma(h)-1}$th power, respectively:

$$h^{p^{\sigma(h)-1}} = \left(g^{p^{\sigma(g)-1}}\right)^{x_0}.$$

If this equation has no solution, we know that $h^{p^{\sigma(h)-1}} \notin \langle g \rangle$, hence $\tau = \sigma(h)$ and $x = 0$. Having computed x_{i-1}, we determine x_i from the equation

$$h^{p^{\sigma(h)-(i+1)}} * \left(g^{-\Sigma_{j=0}^{i-1} x_j p^j}\right)^{p^{\sigma(g)-(i+1)}} = \left(g^{p^{\sigma(g)-1}}\right)^{x_i}, \tag{3.3}$$

for $i = 1, \ldots, \min\{\sigma(g), \sigma(h)\} - 1$. If this equation has no solution, we know that $h^{p^{\sigma(h)-(i+1)}} \notin \langle g \rangle$. Then, we put $\tau = \sigma(h) - i$ and $x = \sum_{j=0}^{i-1} x_j p^{j+\sigma(g)-(i+1)}$ and stop.

This method has the advantage that each coefficient x_i computed by the algorithm also appears as a coefficient in the final result, although at the time of its computation it is not known with which power of p it is associated. But this is clear as soon as the algorithm stops. Thus, our algorithm performs as many discrete logarithm computations in groups of order p as the original Pohlig–Hellman algorithm would have to perform to compute $x = \log_{h^{p^\tau}} g$ under the assumption that τ is already known. In other words, we obtain τ (almost) for free.

The proof of the correctness and the complexity analysis of our algorithm follow from the corresponding statements for the general case in Sections 5 and 6.

4. Solving the EDLP

Let R be a subset of G whose elements generate a direct product. Let H denote the subgroup of G generated by R. Similarly to the special case $|R| = 1$, we have that if (x, y) is a solution of the EDLP for the base (R, h), the number y is the order of hH in the factor group G/H and therefore a prime power: $y = p^\tau$ for some $\tau \geq 0$.

We need the following preliminary result, which, for each non-negative integer i such that p^i is less than or equal to the exponent of H, provides a generating set of the subgroup of H of exponent p^i and such that this set generates a direct product.

Lemma 4.1. *Let $i \in \mathbb{N}_0$, $0 \leq i \leq \max\{\sigma(g) : g \in R\}$. Let*

$$a_{g,i} = \begin{cases} g^{p^{\sigma(g)-i}} & \text{if } \sigma(g) > i, \\ g & \text{if } \sigma(g) \leq i, \end{cases}$$

and

$$R_i = \{a_{g,i} : g \in R\}.$$

Then $\sigma(a_{g,i}) = \min\{i, \sigma(g)\}$ for all $g \in R$. Further, the set R_i generates the subgroup A_i of all elements of H of order $\leq p^i$ in G, and we even have

$$A_i = \bigotimes_{a \in R_i} \langle a \rangle.$$

Proof. The first assertion follows immediately from the definitions of $a_{g,i}$ and the map σ. Given $h \in A_i$, let $x \in \mathbb{Z}^R$ be the uniquely determined map such that

$$R^x = h \qquad \text{and} \qquad 0 \leq x(g) < p^{\sigma(g)}, \qquad g \in R,$$

and put

$$y(a_{g,i}) = \begin{cases} x(g)/p^{\sigma(g)-i} & \text{if } \sigma(g) > i, \\ x(g) & \text{if } \sigma(g) \leq i, \end{cases} \qquad g \in R. \tag{4.1}$$

Using the facts that $H = \bigotimes_{g \in R} \langle g \rangle$ and that $g^{p^i x(g)} = 1$ for each $g \in R$, we obtain that (4.1) defines a map y in $\mathbb{Z}^{R_i}$, and we also have $0 \leq y(a) < p^{\sigma(a)}$ for all $a \in R_i$. Moreover, ${R_i}^y = R^x = h$. That y is uniquely determined by these properties can be easily derived from the uniqueness of x. □

Now, to simplify the notation, for $i \in \mathbb{Z}$ we define the map $P_i \in \mathbb{Z}^R$ by

$$P_i(g) = \begin{cases} p^{\sigma(g)+i} & \text{if } i \geq -\sigma(g), \\ 1 & \text{if } i < -\sigma(g), \end{cases} \qquad g \in R.$$

If $h \in H$, and (x, p^τ) is a solution of the EDLP for the base (R, h), then, analogously to (3.2), there exists $\nu \in \mathbb{N}_0$ and $x_0, \ldots, x_{\nu-1} \in \mathbb{Z}^R$ such that

$$x = x_0 P_{-\nu} + x_1 P_{-\nu+1} + \cdots + x_{\nu-1} P_{-1} = \sum_{i=0}^{\nu-1} x_i P_{-\nu+i}\ , \tag{4.2}$$

i.e.

$$x(g) = \sum_{i=0}^{\nu-1} x_i(g) p^{\sigma(g)-\nu+i}, \qquad g \in R,$$

where

$$\begin{array}{rll} x_0 \neq 0 & (\text{if } \nu \geq 1), & \\ x_i(g) = 0, & i < \nu - \sigma(g), & g \in R, \\ 0 \leq x_i(g) < p, & i = 0, \ldots, \nu - 1, & g \in R. \end{array} \tag{4.3}$$

It is immediately seen that the coefficient maps $x_0, x_1, \ldots, x_{\nu-1}$ are uniquely determined. They are computed recursively. We will give an algorithm that computes these maps and determines ν and τ, from which we obtain the solution (x, p^τ) of the EDLP with base (R, h).

Proposition 4.1. *Let τ be the smallest non-negative integer such that $h^{p^\tau} \in H$. Let $x = \log_R h^{p^\tau}$ be written as in (4.2) with $x_0, \ldots, x_{\nu-1}$ as in (4.3). Then we have $\nu = \sigma(h) - \tau$.*

Proof. If $\nu = 0$ in (4.2), we have $x = 0$. Therefore, $h^{p^\tau} = 1$. Because of the minimality of τ this implies that $\tau = \sigma(h)$. Now assume that $\nu \geq 1$ in (4.2). Since for each g in R we have that

$$\operatorname{ord} g^{x(g)} = \frac{\operatorname{ord} g}{\gcd(x(g), \operatorname{ord} g)} \leq \frac{\operatorname{ord} g}{p^{\sigma(g)-\nu}} = p^\nu,$$

the order of R^x is bounded by p^ν. But since $\nu \geq 1$, there exists $\hat{g}$ in R such that $x_0(\hat{g}) \neq 0$, which implies that $\hat{g}^{x(\hat{g})}$ has order p^ν. Therefore, $\operatorname{ord} R^x = p^\nu$. Since h^{p^τ} has order $p^{\sigma(h)-\tau}$, it follows that $\nu = \sigma(h) - \tau$. □

The method to compute ν and τ and the coefficient maps $x_0, \ldots, x_{\nu-1}$ works analogously to the technique used for $|R| = 1$. The complicating difference is that we have to deal with all elements of R simultaneously.

In the beginning we set

$$\begin{aligned} \sigma_{\max} &= \max\{\sigma(g) : g \in R\}, \\ \nu_0 &= \min\{\sigma_{\max}, \sigma(h)\}. \end{aligned} \tag{4.4}$$

Note that ν_0 is an upper bound for ν, while the index i below always serves as a lower bound for ν. If $\nu_0 = 0$, we put

$$\tau = \sigma(h) \qquad \text{and} \qquad x = 0$$

and stop. If $\nu_0 > 0$, we necessarily have $R \neq \{1\}$ so that $\sigma(g) \geq 1$ for all g in R. We set

$$\begin{aligned} T &= \{g^{p^{\sigma(g)-1}} : g \in R\} \qquad (= R_1 \text{ from Lemma 4.1}), \\ \alpha_0 &= h^{p^{\sigma(h)-\nu_0}}, \\ \gamma_0 &= h^{p^{\sigma(h)-1}}, \end{aligned} \tag{4.5}$$

and for $i = 0, 1, 2, \ldots$ we do the following. We check whether $\gamma_i \in \langle T \rangle$. If this is not the case, we put

$$\nu = i, \qquad \tau = \sigma(h) - \nu \qquad \text{and} \qquad x = \sum_{i=0}^{\nu-1} x_i P_{-\nu+i}$$

and stop. Otherwise, we compute

$$x_i = \log_T \gamma_i$$

and determine

$$\nu_{i+1} = \min\{\{\nu_i\} \cup \{\sigma(g) + i : x_i(g) \neq 0\}\}, \tag{4.6}$$

which serves as a new upper bound for ν. If $\nu_{i+1} = i + 1$, we put

$$\nu = \nu_{i+1}, \qquad \tau = \sigma(h) - \nu \qquad \text{and} \qquad x = \sum_{i=0}^{\nu-1} x_i P_{-\nu+i}$$

and stop. Otherwise, we compute

$$\begin{aligned} \alpha_{i+1} &= \alpha_i^{p^{\nu_i - \nu_{i+1}}} * R^{-x_i P_{-\nu_{i+1}+i}}, \\ \gamma_{i+1} &= \alpha_{i+1}^{p^{\nu_{i+1}-i-2}}. \end{aligned} \tag{4.7}$$

Note that since $\nu_{i+1} \leq \sigma(g)$ for all g with $x_i(g) \neq 0$, and $\nu_{i+1} \geq i + 2$, both expressions on the right-hand sides are well defined. We then increment i and do the next loop.

Here is the pseudocode for this algorithm.

Algorithm 4.1. Solve EDLP

Input: $h \in G$, $R \subseteq G$ *such that* $\langle R\rangle = \bigotimes_{g\in R}\langle g\rangle$, *prime number* p *such that* $|G| = p^\eta$ *for some* $\eta \in \mathbb{N}$.
Output: $x \in \mathbb{Z}^R$ *and* $\tau \in \mathbb{N}_0$ *such that* (x, p^τ) *solves the EDLP for the base* (R, h).

```
$\sigma_{\max} = \max\{\sigma(g) : g \in R\}$
$\nu = \min\{\sigma_{\max}, \sigma(h)\}$
if $(\nu \neq 0)$ then
  $T = \{g^{p^{\sigma(g)-1}} : g \in R\}$
  $\alpha = h^{p^{\sigma(h)-\nu}}$
  $\gamma = h^{p^{\sigma(h)-1}}$
  $i = 0$                                              /* $i$ = current lower
  while $(i < \nu)$ do                                 bound for $\nu$ */
    check whether $\gamma \in \langle T\rangle$
    if $(\gamma \notin \langle T\rangle)$ then
      $\nu = i$
    else
      compute $x_i = \log_T \gamma$                     /* $x_i \in \mathbb{Z}^R$ */
      $\mu = \nu$
      $\nu = \min\{\{\nu\} \cup \{\sigma(g) + i : x_i(g) \neq 0, g \in R\}\}$
      if $(\nu > i + 1)$ then
        $\alpha = \alpha^{p^{\mu-\nu}} * R^{-x_i P_{-\nu+i}}$       /* prepare for next
        $\gamma = \alpha^{p^{\nu-i-2}}$                          while loop */
      fi
      $i = i + 1$
    fi
  od
fi
$x = \sum_{i=0}^{\nu-1} x_i P_{-\nu+i}$                /* $x \in \mathbb{Z}^R$ */
$\tau = \sigma(h) - \nu$
return $x,\tau$
```

Remark 4.1. We intentionally do not specify how to check whether $\gamma \in \langle T\rangle$, and how to compute $\log_T \gamma$ if this is the case. For example, this can be done by Shanks' baby-step giant-step method (Buchmann *et al.*, 1997) or Pollard's rho method (Teske, 1998b).

Remark 4.2. We implemented Solve EDLP in C++ using the Computer Algebra System LiDIA (1997). We used it in the context of group structure computation for groups of points of elliptic curves over finite fields of characteristic $\neq 2, 3$ and for class groups of imaginary quadratic orders.

4.1. A TOY EXAMPLE

Let $G = \mathbb{F}_{97}^*$ be the multiplicative group of the finite field $\mathbb{F}_{97}$ with group order $|G| = 2^5 \cdot 3$. Let $g = 73$ and $h = 23$. We want to solve the EDLP for the base $(\{g\}, h)$. All computations are modulo 97. In the 2-Sylow group of G we work with $g_2 = 73^3 = 47$ and $h_2 = 23^3 = 42$. Then $\sigma(g_2) = 3$ and $\sigma(h_2) = 5$ and $T = 96$. Initially, we have $\nu = 3$ and $\alpha = 33$ and $\gamma = 96$. Since $\gamma = T$, we obtain that $x_0 = 1$ and compute the updated values for α and γ as $\alpha = 33 * 47^{-1} = 75$ and $\gamma = 75^2 = 96$. Since $\gamma = T$, we have $x_1 = 1$. The new values for α and γ are $\alpha = 75 * 47^{-2} = 1$ and $\gamma = 1$, which implies that $x_2 = 0$. We then obtain that $\tau = 2$ and $x_{p=2}(g) = 3$, i.e. ${h_2}^4 = {g_2}^3$. In the 3-Sylow group of G we work with $g_3 = 73^{32} = 35$ and $h_3 = 23^{32} = 61$. Hence, $\sigma(g_3) = \sigma(h_3) = 1$. Moreover, $T = 35$ and, initially, $\nu = 1$ and $\alpha = \gamma = 61$. We compute that $x_0 = 2$ and $\tau = 0$, which means that $x_{p=3} = 2$ and $h_3 = {g_3}^3$. Hence, $y = 4$ and $x(g) \equiv 3 \bmod 8$ and $x(g) \equiv 2 \bmod 3$. By Chinese remaindering we get that $(y, x) = (2, 11)$ solves the EDLP for the base $(\{g\}, h)$: $h^4 = g^{11}$ while $h^2 \notin \langle g \rangle$.

5. Correctness of the Algorithm Solve EDLP

Now we prove that the algorithm Solve EDLP always terminates and that its output $\{x, \tau\}$ is correct. This will follow from the next two theorems.

Theorem 5.1. *Let τ be the smallest non-negative integer such that $h^{p^\tau} \in H$. Let $\nu = \sigma(h) - \tau$, and let T and γ_i, ν_i, $i = 0, 1, 2, \ldots$ be given by the equations (4.4)–(4.7). Then the following two implications hold.*

$$\gamma_i \in \langle T \rangle \Longrightarrow \nu_{i+1} \geq \nu \geq i + 1, \qquad i = 0, 1, 2, \ldots, \tag{5.1}$$

and

$$\gamma_i \notin \langle T \rangle \Longrightarrow \nu \leq i, \qquad i = 0, 1, 2, \ldots. \tag{5.2}$$

Let $x_0, x_1, x_2, \ldots$ be the coefficient maps computed by EDL for p-Groups. *Then*

$$h^{p^{\sigma(h)-i-1}} = R^{\sum_{j=0}^{i} x_j P_{-i+j-1}}, \qquad i = 0, 1, 2, \ldots. \tag{5.3}$$

Proof. First note that when working with γ_i, $i = 1, 2, \ldots$, we always assume that $\gamma_j \in \langle T \rangle$ for $0 \leq j < i$. There is no loss of generality, since this is exactly the situation of Solve EDLP.

From (4.5) and (4.7) we obtain by induction over i that

$$\alpha_i = h^{p^{\sigma(h)-\nu_i}} * R^{-\sum_{j=0}^{i-1} x_j P_{-\nu_i+j}}, \qquad i = 0, 1, 2, \ldots,$$

hence

$$\gamma_i = h^{p^{\sigma(h)-i-1}} * R^{-\sum_{j=0}^{i-1} x_j P_{-i+j-1}}, \qquad i = 0, 1, 2, \ldots. \tag{5.4}$$

Now assume that for some non-negative integer i we have $\gamma_i \in \langle T \rangle$, and let $x_i = \log_T \gamma_i$. Then (5.4) can be rewritten as

$$h^{p^{\sigma(h)-i-1}} = T^{x_i} * R^{\sum_{j=0}^{i-1} x_j P_{-i+j-1}} = R^{\sum_{j=0}^{i} x_j P_{-i+j-1}}, \qquad i = 0, 1, 2, \ldots. \tag{5.5}$$

Note that if $i \geq 1$, the term γ_i is only computed if $i < \nu_i$. Further, if j is such that $0 \leq j \leq i - 1$ and $x_j(g) \neq 0$, we have by (4.6) that $\nu_i \leq \sigma(g) + j$. This implies that

$\sigma(g) - i + j - 1 \geq 0$ if $x_j(g) \neq 0$, so that the terms in (5.5) are well defined. This equation gives (5.3) and also implies that $\tau \leq \sigma(h) - i - 1$, hence $\nu \geq i + 1$.

To prove the statement in (5.1) on the upper bound for ν we set $\gamma_{-1} = 1$ and $x_{-1} = 0 \in \mathbb{Z}^R$. Then (5.4) and (5.5) hold for $i = -1, 0, 1, \ldots$. We then use induction over $i \geq -1$. Consider first $\nu_0 = \min\{\sigma(h), \sigma_{\max}\}$. If $\nu_0 = \sigma(h)$, then $\nu_0 \geq \nu$ by the definition of ν. If $\nu_0 = \max\{\sigma(g) : g \in R\}$, let $x \in \mathbb{Z}^R$ such that $h^{p^\tau} = R^x$. Since

$$p^\nu = p^{\sigma(h)-\tau} = \operatorname{ord} h^{p^\tau} = \operatorname{ord} R^x \leq \max\{\operatorname{ord} g : g \in R\} = p^{\nu_0},$$

we have $\nu_0 \geq \nu$. Now assume (5.1) holds for $i - 1 \geq -1$. Let $\gamma_i \in \langle T \rangle$, and let $x_i = \log_T \gamma_i$. We have to show that $\nu_{i+1} \geq \nu$. If in (4.6) we have $\nu_{i+1} = \nu_i$, we use the induction hypothesis $\nu_i \geq \nu$ to obtain that $\nu_{i+1} \geq \nu$. If $\nu_{i+1} \neq \nu_i$, let $\hat{g} \in R$ such that $x_i(\hat{g}) \neq 0$ and $\nu_{i+1} = \sigma(\hat{g}) + i$. We use (5.5) and write

$$h^{p^{\sigma(h)-i-1}} = \hat{g}^{y(\hat{g})} * \left(\prod_{g \in R \setminus \{\hat{g}\}} g^{\sum_{j=0}^{i} x_j(g) p^{\sigma(g)-i+j-1}} \right),$$

where the exponent $y(\hat{g})$ of $\hat{g}$ satisfies

$$0 < y(\hat{g}) = x_i(\hat{g}) p^{\sigma(\hat{g})-1} + \sum_{j=0}^{i-1} x_j(\hat{g}) p^{\sigma(\hat{g})-i+j-1} < p^{\sigma(\hat{g})}.$$

Now assume $\nu_{i+1} < \nu$. Then there exists some non-negative integer ρ such that $\nu_{i+1} + 1 + \rho = \nu$, hence $\sigma(\hat{g}) + i + 1 + \rho = \sigma(h) - \tau$. Equivalently, $\sigma(h) - i - 1 = \tau + \sigma(\hat{g}) + \rho$. Let $x = \log_R h^{p^\tau}$. Then

$$h^{p^{\sigma(h)-i-1}} = \left(h^{p^\tau}\right)^{p^{\sigma(\hat{g})+\rho}} = (R^x)^{p^{\sigma(\hat{g})+\rho}} = \left(\prod_{g \in R \setminus \{\hat{g}\}} g^{x(g)} \right)^{p^{\sigma(\hat{g})+\rho}},$$

since $\hat{g}^{x(\hat{g}) p^{\sigma(\hat{g})+\rho}} = 1$. This is a contradiction to the uniqueness of $\log_R h^{p^{\sigma(h)-i-1}}$. Hence, $\nu_{i+1} \geq \nu$, which completes the proof of (5.1).

Next, from (5.4) we deduce that

$$\gamma_{i-1} = \gamma_i^p * T^{x_{i-1}}, \qquad i = 1, 2, \ldots.$$

On the other hand, we have $\gamma_{i-1} = T^{x_{i-1}}$, $i = 1, 2, \ldots$. Therefore, $\gamma_i^p = 1$ for $i = 1, 2, \ldots$, so that γ_i has order $\leq p$, which also holds for $i = 0$. This, together with Lemma 4.1, implies that if $\gamma_i \in H$, then $\gamma_i \in \langle T \rangle$. In other words, $\gamma_i \notin \langle T \rangle$ implies that $\gamma_i \notin H$. Now assume there exists $i \geq 0$ such that $\gamma_i \notin \langle T \rangle$ and $\nu > i$. Then since $\tau \leq \sigma(h) - i - 1$ and since $h^{p^\tau} \in H$, there exists $y \in \mathbb{Z}^R$ such that $R^y = h^{p^{\sigma(h)-i-1}}$. Together with (5.4) this yields

$$\gamma_i = R^{y - \sum_{j=0}^{i-1} x_j P_{-i+j-1}} \in H,$$

which is a contradiction. Therefore, we must have $\nu \leq i$ if $\gamma_i \notin \langle T \rangle$, so that (5.2) holds. □

Theorem 5.2.

(1) Solve EDLP *always terminates.*
(2) The output of Solve EDLP *is correct.*

Proof. (1) Let ν_i, γ_i and T be as in (4.4)–(4.6). If $\nu_0 = 0$, or if for some $i \geq 0$ it happens that $\log_T \gamma_i$ does not exist, the finiteness of Solve EDLP is clear. Now assume that $\nu_0 \geq 1$ and that for $i = 0, 1, 2, \ldots$ we have $\gamma_i \in \langle T \rangle$. We claim that for some $i \leq \nu_0$ we have $\nu_{i+1} = i + 1$. To see this, note that the sequence $(\nu_i)_{i \in \mathbb{N}_0}$ is decreasing and the sequence $(c_i)_{i \in \mathbb{N}_0}$ given by $c_i = i$ is strictly increasing. Therefore, and since the sequence (c_i) is discrete, there exists a smallest number $i_* \in \mathbb{N}$ such that $\nu_{i_*+1} \leq i_* + 1$. On the other hand, it follows from (5.1) that $\nu_{i+1} \geq i+1$ for $i = 0, 1, 2, \ldots$. Hence, $\nu_{i_*+1} = i_*+1$, so that the algorithm terminates when $i = i_*$.

(2) Solve EDLP can terminate on three different occasions:

(i) it terminates because $\nu_0 = 0$. Then $\sigma(h) = 0$, or $\sigma(g) = 0$ for all $g \in R$, if $\sigma(h) = 0$. This means that $h = 1$, hence $h = R^x$ with $x = 0$. If $\sigma(g) = 0$ for all $g \in R$, we have $H = \{1\}$. Therefore, $h^{p^y} \in H$ only if $y \geq \sigma(h)$ and the smallest such number y is given by $\sigma(h)$; then $h^{p^{\sigma(h)}} = R^x$ with $x = 0$. In both cases, the output $x = 0$ and $\tau = \sigma(h)$ is correct.

(ii) There is some $i_* \geq 0$ such that $\log_T \gamma_{i_*}$ does not exist. Equations (5.1) and (5.2) of Theorem 5.1 imply that $\nu \geq i_*$ and $\nu \leq i_*$, hence $\nu = i_*$. With τ denoting the smallest non-negative integer such that $h^{p^\tau} \in H$ and $\nu = \sigma(h) - \tau$, putting $i = \nu - 1$ in (5.3) yields

$$h^{p^\tau} = R^{\sum_{j=0}^{\nu-1} x_j P_{-\nu+j}},$$

so that the output $x = \sum_{j=0}^{\nu-1} x_j P_{-\nu+j}$ and $\tau = \sigma(h) - i_*$ is correct.

(iii) There is some $i_* \geq 1$ such that $\log \gamma_{i_*}$ exists and $\nu_{i_*+1} = i_* + 1$. In this case, Equation (5.1) gives $i_* + 1 \leq \nu \leq \nu_{i_*+1}$. Therefore, $\nu = i_* + 1$. Just as in case (ii) we then obtain from (5.3) that the output $x = \sum_{j=0}^{i_*} x_j P_{-i_0+j}$ and $\tau = \sigma(h) - (i_* + 1)$ is correct. □

6. Run Time of Solve EDLP

To estimate the run time of the algorithm Solve EDLP, we consider the algorithm to check whether $\gamma \in \langle T \rangle$ and to compute $x_i = \log_T \gamma$ as a black-box algorithm with (expected) run time bounded by $Z(p, t)$, where p is the exponent and t is the rank of the subgroup $\langle T \rangle$. Note that since $\langle T \rangle$ is the direct product of its generators (Lemma 4.1), we have $t = |T| = |R|$. For example, if we use the baby-step giant-step method in this black-box algorithm, by arguments similar to those in Buchmann *et al.* (1997), we obtain that one has to perform

$$Z_{\mathrm{S}}(p, t) = \Theta\left(\left\lceil p^{1/2} \right\rceil^t\right)$$

group multiplications, and the space requirements are of the same size.

The run time of the algorithm Solve EDLP in terms of the black-box run time $Z(\cdot, \cdot)$ is given by the following theorem.

Theorem 6.1. *Let h, R and p be the input of* Solve EDLP*, and let s be such that p^s is the exponent of $\langle R \rangle$. Let G be the group generated by $R \cup \{h\}$. To compute the solution (x, p^τ) of the EDLP for the base (R, h), the algorithm* Solve EDLP *needs a run time bounded by*

$$\min\{s, \sigma(h)\} \cdot Z(p, |R|)$$

and, in addition to this, performs

$$\min\{s,\sigma(h)\}\cdot O(\log|G|)$$

group multiplications, $|R|$ *inversions and* $O(\log|G|)$ *equality checks.*

Proof. Since the elements of R generate a direct product, we have

$$\prod_{g\in R} p^{\sigma(g)} = \prod_{g\in R} \operatorname{ord} g = |\langle R\rangle|. \tag{6.1}$$

Together with the bound (2.3), this implies that to compute $\sigma(g)$ for all $g\in R$, Solve EDLP performs at most $3\log|\langle R\rangle|$ group multiplications and at most $|R|+\frac{\log|\langle R\rangle|}{\log p}$ equality checks. Note that $|R|\le\frac{\log|\langle R\rangle|}{\log p}$. To compute $\sigma(h)$ requires at most $3\log(\operatorname{ord} h)$ group multiplications and $1+\log(\operatorname{ord} h)/\log p$ equality checks. Equation (6.1) also implies that to compute the set $T=\{g^{p^{\sigma(g)-1}} : g\in R\}$ requires at most $2\log|\langle R\rangle|$ group multiplications, if we use the binary method of exponentiation. Correspondingly, to initialize α and γ requires at most $4\log(\operatorname{ord} h)$ multiplications. Thus, the whole initialization step requires $O(\log|G|)$ multiplications and equality checks in G. It remains to estimate the run time of the while loop, which is executed at most $\min\{s,\sigma(h)\}$ times. Each while loop requires time $Z(p,|R|)$, plus the time to update α and γ by the binary method of exponentiation. To estimate the latter time, note that $\mu-\nu\le\sigma(h)$, and that $\nu>i+1$ implies that $x_i(g)p^{\sigma(g)-\nu+i}\le p^{\sigma(g)-2}$ for each $g\in R$. We use these estimates together with (6.1) and obtain that to update each α requires at most $2\log(\operatorname{ord} h)+2\log|\langle R\rangle|$ multiplications in G. Moreover, in the first while loop, $|R|$ inversions are performed to compute $\{g^{-1} : g\in R\}$. To update γ requires at most $2\log(\operatorname{ord} h)$ multiplications. Hence, to execute all while loops requires run time at most $\min\{s,\sigma(h)\}\cdot Z(p,|R|)$, at most

$$\min\{s,\sigma(h)\}\cdot O(\log|G|)$$

multiplications and $|R|$ inversions. □

Remark 6.1. It can immediately be seen from the proof of Theorem 6.1 that as O-constants for the numbers of multiplications and equality checks we can choose 18 and 4, respectively.

Remark 6.2. Using Pohlig–Hellman-like techniques to check for $j=0,1,2,\ldots$ whether $h^{p^j}\in\langle R\rangle$ and to compute an appropriate $x\in\mathbb{Z}^R$ if this is the case has run time $O\left(\min\{s,\sigma(h)\}^2\cdot Z(p,|R|)\right)$ (cf. Paulus, 1992).

7. Conclusion

We have generalized the discrete logarithm problem to the cases that a set of several elements serves as a base rather than a single element, and that $h\notin\langle g\rangle$ but the least number y is wanted such that $h^y\in\langle g\rangle$. We have described and analyzed a Pohlig–Hellman-like algorithm to solve this extended DLP. We implemented this algorithm, Solve EDLP, in `C++` using the Computer Algebra System LiDIA (1997).

References

Buchmann, J., Jacobson, Jr, M. J., Teske, E. (1997). On some computational problems in finite Abelian groups. *Math. Comput.*, **66**, 1663–1687.

Hafner, J. L., McCurley, K. S. (1989). A rigorous subexponential algorithm for computation of class groups. *J. Am. Math. Soc.*, **2**, 839–850.

LiDIA (1997). LiDIA–A library for computational number theory, Version 1.3. LiDIA Group, Technische Universität Darmstadt. Available from `http://www.informatik.tu-darmstadt.de/TI/LiDIA`.

Paulus, S. (1992). Algorithmen für endliche abelsche Gruppen. Master's Thesis, Universität des Saarlandes, Saarbrücken, Germany.

Pohlig, S. C., Hellman, M. E. (1978). An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Trans. Inf. Theory*, **24**, 106–110.

Schoof, R. J. (1985). Elliptic curves over finite fields and the computation of square roots mod p. *Math. Comput.*, **44**, 483–494.

Teske, E. (1998a). New algorithms for finite Abelian groups. Ph.D. Thesis, Technische Universität Darmstadt, Germany.

Teske, E. (1998b). A space efficient algorithm for group structure computation. *Math. Comput.*, **67**, 1637–1663.

Originally Received 31 July 1998
Accepted 05 April 1999