Designs, Codes and Cryptography, 7, 153–174 (1996)
© 1996 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Key-Exchange in Real Quadratic Congruence Function Fields

R. SCHEIDLER scheidle@math.udel.edu
Department of Mathematical Sciences, University of Delaware, Newark, DE 19716 U. S. A.

A. STEIN andreas@math.uni-sb.de
FB-9 Mathematik, Universität des Saarlandes, 66041 Saarbrücken, Germany

HUGH C. WILLIAMS* hugh_williams@csmail.cs.umanitoba.ca
Department of Computer Science, University of Manitoba, Winnipeg, MB, Canada R3T 2N2

Dedicated to Gus Simmons

Abstract. We show how the theory of real quadratic congruence function fields can be used to produce a secure key distribution protocol. The technique is similar to that advocated by Diffie and Hellman in 1976, but instead of making use of a group for its underlying structure, makes use of a structure which is "almost" a group. The method is an extension of the recent ideas of Scheidler, Buchmann and Williams, but, because it is implemented in these function fields, several of the difficulties with their protocol can be eliminated. A detailed description of the protocol is provided, together with a discussion of the algorithms needed to effect it.

1. Introduction

Conventional or one-key cryptosystems are still the secure communication schemes of choice for many installations. This is because they are both fast and sufficiently secure for most applications. The real difficulty in employing such cryptosystems is the problem of securely transmitting the key between communicants. In 1976, Diffie and Hellman [8] described a possible solution to this problem by making use of the multiplicative group $\mathbb{F}_p{}^*$ of integers relatively prime to a large prime p. More generally, we can let G be any group such that $|G|$ $(= n)$ is large. Consider two communicants, Alice and Bob, who first select publicly an element g of large order in G. Alice selects at random a positive integer y $(< n)$ and Bob selects at random a positive integer z $(< n)$. (y and z are kept secret.) Alice then transmits $h_1 = g^y$ to Bob and Bob transmits $h_2 = g^z$ to Alice. At this point, Alice evaluates $k = h_2^y = g^{zy}$ and Bob evaluates $k = h_1^z = g^{yz}$. Because there is no fast method known for determining k, given only h_1 and h_2, Alice and Bob can now use some same aspects of k to produce their secret communication key. The security of this scheme is based on the presumed difficulty of the *discrete logarithm problem* (DLP) in G; that is, given some g and some $h = g^y$, find y. A fast algorithm for solving the DLP will lead to the discovery of the key k; unfortunately, it is unknown whether it is really necessary to solve an instance of the DLP in order to break the system. The Diffie-Hellman technique

* Research supported by NSERC of Canada Grant #A7649

and all of its extensions make use of this idea, only the choice of G varies. Of course, G here should be selected such that the DLP in this structure is a hard problem.

Recently, Scheidler, Buchmann and Williams [12] were able, for the first time, to exhibit a secure key exchange protocol, similar in concept to that of Diffie-Hellman, which does not make use of a group as the underlying structure. This scheme is based on the infrastructure [15] of the principal ideal class of a real quadratic number field. Unfortunately, this technique possesses a number of disadvantages not shared by the standard Diffie-Hellman protocol: increased bandwidth, a need to deal with high precision approximations to certain algebraic numbers, and an ambiguity problem which necessitates a short, second round of communication. In this paper we show how to eliminate all of these disadvantages, and yet maintain the same type of (non-group) structure, by implementing the basic idea of [12] in a real quadratic congruence function field over a finite field $\mathbb{F}_q$ of constants, where q is odd, instead of in a real quadratic number field. This appears to be the first time that the theory of algebraic function fields has been applied to cryptography.

Stein [16] has shown that Shanks' infrastructure idea also applies to the set of reduced principal ideals in a real quadratic congruence function field. Thus, many of the techniques needed to produce the scheme in [12] can also be used in these function fields. Furthermore, because the distances between reduced principal ideals in real quadratic congruence function fields are rational integers instead of logarithms of algebraic numbers, as they are in the real quadratic number field case, we are able to eliminate the difficulties mentioned earlier. It may even be that the security of this new system is better than that of [12]. At the moment the only methods known for solving the problem analogous to the DLP in the set of reduced ideals in a real quadratic congruence function field are of exponential complexity, whereas techniques of subexponential complexity are known for solving the same problem in real quadratic number fields.

In Sections 2 and 3 of this paper, we outline the basic properties of real quadratic congruence function fields and, in particular, describe the arithmetic of reduced ideals. In Section 4, we make use of these ideals to develop the algorithms that we require and analyze their complexity. The overall protocol is presented in Section 5 and security issues are discussed in Section 6. The paper concludes with a brief mention of some computer implementation issues and some timings for a certain set of examples.

2. Real Quadratic Congruence Function Fields

2.1. *Basic Definitions*

In this section, we present the situation as described in [18], [16] and [19]. Basic references for this subject are [3], [7] and [20].

Let $K/\mathbb{F}_q$ be a *quadratic congruence function field* over a finite field $\mathbb{F}_q$ of *constants* of odd characteristic with q elements. Then K is a quadratic extension of the rational function field $\mathbb{F}_q(x)$ with a transcendental element $x \in K$. We say that K is a *real* quadratic congruence function field (of odd characteristic) if K is of the form $K = \mathbb{F}_q(x)(\sqrt{D}) = \mathbb{F}_q(x) + \mathbb{F}_q(x)\sqrt{D}$, where $D \in \mathbb{F}_q[x]$ is a squarefree polynomial of even degree whose leading coefficient is a square in $\mathbb{F}_q{}^* = \mathbb{F}_q \setminus \{0\}$. (This is in analogy to the case of a real quadratic number field $\mathbb{Q}(\sqrt{\Delta})$, where Δ is a positive, squarefree integer). The *ring of integers of K* is $\mathcal{O} = \mathbb{F}_q[x][\sqrt{D}] = \mathbb{F}_q[x] + \mathbb{F}_q[x]\sqrt{D}$. For $\alpha = u + v\sqrt{D} \in K (u, v \in \mathbb{F}_q(x))$, denote by $\overline{\alpha} = u - v\sqrt{D}$ its *conjugate*.

In contrast to the number field case, there are two places of K at infinity. We know from [19] that the place at infinity $\mathfrak{P}_\infty$ of $\mathbb{F}_q(x)$ with respect to x splits in K as $\mathfrak{P}_\infty = \mathfrak{P}_1 \cdot \mathfrak{P}_2$. Furthermore, the completions of K with respect to $\mathfrak{P}_1$ and $\mathfrak{P}_2$, $K_{\mathfrak{P}_1}$, and $K_{\mathfrak{P}_2}$, respectively, are isomorphic to $\mathbb{F}_q(x)_{\mathfrak{P}_\infty} = \mathbb{F}_q((1/x))$, the field of power series in $1/x$. By explicitly taking square roots of D, we see that K is a subfield of $\mathbb{F}_q((1/x))$. Let $\mathfrak{P}_1$ be the place which corresponds to the case where $\sqrt{1} = 1$. Then we consider elements of K as Laurent series at $\mathfrak{P}_1$ in the variable $1/x$. Let $\alpha \in \mathbb{F}_q((1/x))$ be a non-zero element. Then $\alpha = \sum_{i=-\infty}^{m} c_i x^i$ with $c_m \neq 0$. Denote by

$$\begin{aligned}
\deg(\alpha) &= m \quad \text{the } \textit{degree} \text{ of } \alpha,\\
|\alpha| &= q^m \quad \text{the } \textit{absolute value} \text{ of } \alpha,\\
\operatorname{sgn}(\alpha) &= c_m \quad \text{the } \textit{sign} \text{ of } \alpha,\\
\lfloor \alpha \rfloor &= \sum_{i=0}^{m} c_i x^i \quad \text{the } \textit{principal part} \text{ of } \alpha.
\end{aligned}$$

If m is negative, then $\lfloor \alpha \rfloor = 0$. We set $\deg(0) = -\infty$ and $|0| = 0$.

In analogy to the case of a real quadratic number field, the *unit group E* of $K/\mathbb{F}_q$ is of the form $E = \mathbb{F}_q{}^* \times \langle \epsilon \rangle$, where $\epsilon \in K$ is a *fundamental unit* of K. The positive integer $R = \deg(\epsilon)$ is called the *regulator* of $K/\mathbb{F}_q$ with respect to $\mathcal{O}$. Denoting by h' the *ideal class number* and by h the *divisor class number*, we know from [13] that $h = Rh'$.

2.2. *Reduced Ideals and Distances*

A subset $\mathfrak{a}$ of $\mathcal{O}$ is an *(integral) ideal* if both $\mathfrak{a}+\mathfrak{a}$ and $\mathcal{O} \cdot \mathfrak{a}$ are subsets of $\mathfrak{a}$. A *principal* ideal $\mathfrak{a}$ of $\mathcal{O}$ is an ideal of the form $\mathfrak{a} = \alpha \cdot \mathcal{O}$ where $\alpha \in \mathcal{O}$. We say that α *generates* the ideal $\mathfrak{a}$ and write $\mathfrak{a} = (\alpha)$. If the product of two principal ideals (α), (β) is defined to be $(\alpha\beta)$, then the set $\mathcal{P}$ of non-zero principal ideals is a monoid under multiplication with identity $\mathcal{O}$. In our scheme, we will only be considering principal ideals.

As in the case of real quadratic number fields, there is a finite subset $\mathcal{R}$ of $\mathcal{P}$, the *reduced* (principal) ideals, and a natural ordering $\mathfrak{r}_1 = (1) < \mathfrak{r}_2 < \ldots < \mathfrak{r}_m$ of the ideals in $\mathcal{R}$. The exact definition of a reduced ideal as well as a procedure for generating the entire sequence $(\mathfrak{r}_j)_{1 \leq j \leq m}$ are given in Section 3.1.

With each reduced ideal $\mathfrak{r}_j = (\rho_j)$, we associate a *distance*

$$\delta_j = \delta(\mathfrak{r}_j) = \deg(\rho_j).$$

Note that in contrast to the number field case, the distance is a nonnegative integer. δ is unique modulo R, where R is the regulator of K. Furthermore, if $0 \leq \delta_j < R$, then δ_j is uniquely determined and is strictly increasing with j.

For any $z \in [0, \infty)$, there exists a unique index $j \in \mathbb{N}$ such that $\delta_j \leq z \leq \delta_{j+1}$. If $\delta_j = \delta(\mathfrak{r}_j)$, then $\mathfrak{r}_j$ is called the *reduced ideal closest to the left of z*.

2.3. *Outline of the Protocol*

The key space for our Diffie-Hellman protocol is the set $\mathcal{R}$ of reduced ideals in a real quadratic congruence function field. Assume that two communications partners, Alice and Bob, wish to exchange a secret cryptographic key. Then they publicly agree on an odd prime power q and a squarefree polynomial $D \in \mathbb{F}_q[x]$ which defines a real quadratic congruence function field $K = \mathbb{F}_q(x)(\sqrt{D})$. Furthermore, they publicly determine a reduced ideal $\mathfrak{c}$ with small distance $\delta = \delta(\mathfrak{c})$. Now Alice secretly generates an "exponent" $a \in \mathbb{N}$ and computes the reduced ideal $\mathfrak{a}$ closest to the left of $a\delta$ and its distance $\delta(\mathfrak{a})$. She transmits the ideal $\mathfrak{a}$ to Bob. Similarly, Bob chooses $b \in \mathbb{N}$ and computes the reduced ideal $\mathfrak{b}$ closest to $b\delta$ and its distance $\delta(\mathfrak{b})$. He sends $\mathfrak{b}$ to Alice. From $\delta(\mathfrak{a})$ and the ideal $\mathfrak{b}$ received from Bob, Alice computes the reduced ideal closest to the left of $\delta(\mathfrak{a})\delta(\mathfrak{b})$. Similarly, Bob uses $\delta(\mathfrak{b})$ and the ideal $\mathfrak{a}$ he got from Alice to compute the reduced ideal closest to the left of $\delta(\mathfrak{b})\delta(\mathfrak{a})$. Then both parties will have computed the same reduced ideal $\mathfrak{k}$ which can be used to establish a common cryptographic key.

Compared to the analogous protocol in real quadratic number fields, this scheme is much simpler. Since all distances are integers, no approximations are required. Furthermore, both parties obtain the same ideal at the end of their computations. In the number field case, the final ideal is one of two possible candidates, and another round (or at least half a round) of communicating one bit is necessary to establish a unique key ideal. Finally, we will see in Section 4 that we only step in a "forward" direction through the set $\mathcal{R}$ of reduced ideals (i.e. from $\mathfrak{r}_j$ to $\mathfrak{r}_{j+1}$), whereas in the number field situation, it was necessary to step "backwards" through $\mathcal{R}$ in occasional, although rare cases.

The crucial point here is the ability of both parties to compute for $n \in \mathbb{N}$ and $\mathfrak{r} \in \mathcal{R}$ with distance $\delta(\mathfrak{r})$ the reduced ideal closest to the left of $n\delta(\mathfrak{r})$. Before we solve this problem in Section 4, we present the notion of reduced ideals and their distances.

3. Arithmetic of Reduced Ideals

3.1. *Ideals and Continued Fractions*

We now illustrate how ideal arithmetic can be performed in terms of polynomials over the finite field $\mathbb{F}_q$. The connection between ideals and the continued fraction expansion of elements of the form $\alpha = \frac{P+\sqrt{D}}{Q}$ $(Q, P \in \mathbb{F}_q[x])$ where $Q|(D - P^2)$, is shown in [16]. The following base representations go back to Artin [3]. Let $\mathfrak{a}$ be an (integral) ideal. Then there exist polynomials $S, P, Q \in \mathbb{F}_q[x]$ with $Q|(D - P^2)$ such that $\mathfrak{a} = [SQ, SP + S\sqrt{D}] = SQ\mathbb{F}_q[x] + (SP + S\sqrt{D})\mathbb{F}_q[x]$. The set $\{SQ, SP + S\sqrt{D}\}$ is called $\mathbb{F}_q[x]$-*base* of $\mathfrak{a}$. S and Q are unique up to constant factors. If we set $\operatorname{sgn}(S) = \operatorname{sgn}(Q) = 1$, then both are unique. Furthermore, SQ is the greatest common divisor of all polynomials in $\mathbb{F}_q[x]$ which belong to $\mathfrak{a}$.

An ideal is called *primitive* if it has no prime divisors in $\mathbb{F}_q[x]$, or equivalently, if S in the $\mathbb{F}_q$-base can be chosen to be 1. Hence, we associate with each primitive ideal $\mathfrak{a}$ a pair (Q, P) of polynomials in $\mathbb{F}_q[x]$. In particular, for $\mathcal{O}$, we have $Q = 1$ and $P = 0$. Throughout our computations, we will only be using primitive principal ideals, which will be represented by their $\mathbb{F}_q[x]$-bases (Q, P).

Let $\mathfrak{a} = [Q, P + \sqrt{D}]$ be a primitive ideal. Set $d = \lfloor\sqrt{D}\rfloor$, $Q_0 = Q$, $P_0 = P$, $\alpha_0 = \frac{P+\sqrt{D}}{Q}$ and $a_0 = \lfloor\alpha_0\rfloor$. We calculate the continued fraction expansion of α_0 by using the formulas

$$\alpha_i = \frac{1}{\alpha_{i-1} - a_{i-1}} \qquad a_i = \lfloor\alpha_i\rfloor \qquad (i \in \mathbb{N}). \tag{3.1}$$

Then α_i is given by $\alpha_i = \frac{P_i+\sqrt{D}}{Q_i}$, where $0 \neq Q_i, P_i, a_i \in \mathbb{F}_q[x]$, are polynomials such that $Q_i | \left(D - P_i^2\right)$. They can be recursively computed as follows.

$$\left\{\begin{array}{ll} P_i = a_{i-1}Q_{i-1} - P_{i-1} & \\ Q_i = \dfrac{D - P_i^2}{Q_{i-1}} & \\ a_i = (P_i + d) & (\operatorname{div} Q_i) \end{array}\right\} \qquad (i \in \mathbb{N}). \tag{3.2}$$

Setting $\mathfrak{a}_1 = \mathfrak{a}$, and

$$\mathfrak{a}_i = \left[Q_{i-1}, P_{i-1} + \sqrt{D}\right] \qquad (i \in \mathbb{N}), \tag{3.3}$$

we see that each $\mathfrak{a}_i$ is a primitive ideal. The iterative steps of obtaining $\mathfrak{a}_i$ from $\mathfrak{a}_{i-1}$ for $i \in \mathbb{N}$ are called *Baby steps*.

A primitive ideal $\mathfrak{a}$ is called *reduced*, if there exists an $\mathbb{F}_q[x]$-base for $\mathfrak{a}$ of the form $\{Q, P + \sqrt{D}\}$ with polynomials $Q, P \in \mathbb{F}_q[x]$, $Q|(D - P^2)$ such that $|P - \sqrt{D}| < |Q| < |P + \sqrt{D}|$. This *reduced base* representation is unique up to constant factors of Q. Choosing $\operatorname{sgn}(Q) = 1$ makes it unique. From [18] and [16], we know that the polynomials in the reduced base can be characterized by the following Lemmata.

LEMMA 3.1 *Let $\mathfrak{a}$ be a primitive ideal with $\mathbb{F}_q[x]$-base $\{Q, P + \sqrt{D}\}$. Then $\mathfrak{a}$ is reduced if and only if $|Q| < |\sqrt{D}|$.*

LEMMA 3.2 *Let $\mathfrak{a}$ be a reduced ideal and $\{Q, P + \sqrt{D}\}$ be its reduced $\mathbb{F}_q[x]$-base. Then the following properties hold:*

a) $|P| = |P + \sqrt{D}| = |\sqrt{D}| = |d|$.

b) $sgn(P) = sgn(\sqrt{D})$. *Even the two highest coefficients of P and $\sqrt{D}$ are equal.*

c) *Let* $a = (P + d) \quad (\operatorname{div} Q)$. *Then* $|aQ| = |\sqrt{D}|$. *In particular,* $1 < |a| \leq |\sqrt{D}|$ *and* $1 \leq |Q| < |\sqrt{D}|$.

LEMMA 3.3 *Let $\mathfrak{a} = [Q, P + \sqrt{D}]$ be a primitive ideal, and let $\mathfrak{a}_1 = \mathfrak{a}, \mathfrak{a}_2, \mathfrak{a}_3, \ldots$ be the sequence of primitive ideals given by the formulas in (3.3) with the quantities in (3.2). Then the following holds:*

a) $\mathfrak{a}_i$ is reduced for $i > \max\{1, \frac{1}{2}\deg(Q) - \frac{1}{4}\deg(D) + 2\}$.

b) If $\mathfrak{a}_j$ is reduced for some $j \in \mathbb{N}$, then $\mathfrak{a}_i$ is reduced for all $i \geq j$, and the reduced $\mathbb{F}_q[x]$-base of $\mathfrak{a}$ is exactly that given in (3.3).

Because of the bounds and reduction criteria given in the three lemmas, it is clear that reduced ideals exhibit a periodic behavior. In contrast to the case of real quadratic number fields, we have to investigate the quasi-period of certain elements and not only the period. For an element $\alpha \in K$, we say that the sequence $(\alpha_i)_{i\geq 0}$ defined as in (3.1) is *quasi-periodic* if there are integers $m, i_0 \geq 0$, and an element $c \in \mathbb{F}_q{}^*$ such that

$$\alpha_{i_0+m} = c \cdot \alpha_{i_0}. \tag{3.4}$$

The smallest positive integer m, for which (3.4) holds, is called the *quasi-period* of the continued fraction expansion of α. The smallest m, satisfying (3.4) with $c = 1$, is called the *period* of $(\alpha_i)_{i\geq 0}$.

Let $\mathfrak{a} = [Q, P + \sqrt{D}]$ be a primitive ideal, and let $\mathfrak{a}_1 = \mathfrak{a}, \mathfrak{a}_2, \mathfrak{a}_3, \ldots$ be the sequence of primitive ideals given by the formulas in (3.3). As in [16], we conclude from the above lemmas that the continued fraction expansion of $\alpha = \frac{P+\sqrt{D}}{Q}$ is quasi-periodic with quasi-period m. In fact, the corresponding ideal sequence $(\mathfrak{a}_i)_{i\in\mathbb{N}}$ is periodic with period m, i.e. there exists a minimal $i_0 \geq 0$ such that $\mathfrak{a}_{m+i} = \mathfrak{a}_i$ $(i \geq i_0)$.

In the continued fraction expansion of α, we define $\theta_1 = 1$, and $\theta_{i+1} = \prod_{j=1}^{i-1} \frac{1}{\alpha_j}$ for $i \geq 2$. Then $Q_0\theta_i, Q_0\overline{\theta}_i \in \mathcal{O}$ and $\theta_i\overline{\theta}_i = (-1)^{i-1}\frac{Q_{i-1}}{Q_0}$ for $i \geq 1$. Furthermore, we have

$$(Q_0\theta_i)\,\mathfrak{a}_i = (Q_{i-1})\,\mathfrak{a}_1. \tag{3.5}$$

Since $\deg(\alpha_i) = \deg(a_i)$, it follows that

$$\deg\left(\overline{\theta}_i\right) = \deg(Q_{i-1}) - \deg(Q_0) + \sum_{j=1}^{i-1}\deg(a_j) \qquad (i \in \mathbb{N}). \tag{3.6}$$

If we set $\mathfrak{r}_1 = \mathcal{O} = [1, \sqrt{D}]$, then $\mathfrak{r}_1$ is reduced by Lemma 3.1. If we compute $\mathfrak{r}_2, \mathfrak{r}_3, \ldots$ by applying repeated Baby steps, starting at $\mathfrak{r}_1$ (or any reduced ideal $\mathfrak{r}_i$, $i \in \mathbb{N}$), then by Lemma 3.3 b), we obtain a sequence of reduced ideals $(\mathfrak{r}_i)_{i\in\mathbb{N}}$. By (3.5), we get $\mathfrak{r}_i = (\overline{\theta}_i)$ for $i \in \mathbb{N}$. Since we previously observed that this sequence is periodic, we can generate the entire sequence $(\mathfrak{r}_i)_{1\leq i\leq m}$ of reduced ideals in this manner. Thus, the set of reduced ideals is $\mathcal{R} = \{\mathfrak{r}_1, \ldots, \mathfrak{r}_m\}$ and $|\mathcal{R}| = m$, where m denotes the quasi-period of $\alpha = \sqrt{D}$.

Let $\mathfrak{r}_i \in \mathcal{R}$ $(i \in \mathbb{N})$. We define the *distance* of $\mathfrak{r}_i$ to be

$$\delta_i = \deg(\overline{\theta}_i).$$

Note that the distance δ_i is an integer-valued function which is only defined for reduced ideals and strictly increases as i increases. By Lemma 3.2 a) and (3.6), we have

$$\delta_1 = 0, \qquad \delta_i = \deg(d) - \deg(Q_0) + \sum_{j=1}^{i-2} \deg(a_j) \qquad (i \geq 2). \tag{3.7}$$

It follows from Lemma 3.2 c) that

$$1 \leq \delta_{i+1} - \delta_i \leq \deg(d) \qquad (i \in \mathbb{N}). \tag{3.8}$$

Therefore, $\delta_{k+i} \geq \delta_i + k$ for all $k \geq 0$, $i \in \mathbb{N}$. Finally, we get for the regulator: $R = \delta_{m+1}$. If D is chosen appropriately (see Section 6.1), then m might be as large as $O(|\sqrt{D}|) = O(q^{1/2 \deg(D)})$.

We also define the *distance between two reduced ideals* $\mathfrak{r}_i, \mathfrak{r}_j$ $(1 \leq j \leq i)$ to be

$$\delta(\mathfrak{r}_i, \mathfrak{r}_j) = \delta_i - \delta_j.$$

Hence $\delta_i = \delta(\mathfrak{r}_i, \mathcal{O})$ for $i \in \mathbb{N}$.

Before we describe the Baby step method in more detail, let us explain how we will analyse the performance of all our algorithms. We will give the time complexity of an algorithm in terms of polynomial operations over $\mathbb{F}_q$ (additions, subtractions, multiplications, divisions with remainder, degree comparisons, and assignments). We do not consider the computation time each such operations requires. In particular, this means that we will largely ignore the dependence of the running times of polynomial arithmetic on q; however, this dependence is the same for fixed q, regardless of the polynomial D used for our real quadratic congruence function field $K = \mathbb{F}_q(x)(\sqrt{D})$. The space requirement for an algorithm is given in terms of the degrees of the computed polynomials and in terms of the binary length of integers if the algorithm uses rational integers.

The following algorithm computes one Baby step for a reduced principal ideal, and its distance. For the continued fraction algorithm, we use an optimized version due to Tenner [16].

Algorithm *BABYSTEP*

Precomputed: $\mathfrak{a}_{i-1} = (Q_{i-2}, P_{i-2}) \in \mathcal{R}$, $\mathfrak{a}_i = (Q_{i-1}, P_{i-1}) \in \mathcal{R}$, $r_{i-2} \equiv (P_{i-2} + d) \pmod{Q_{i-2}}$, $d_i = \delta(\mathfrak{a}_i, \mathfrak{a}_1)$ $(i \geq 2)$.

Input: $(Q_{i-2}, Q_{i-1}, P_{i-1}, r_{i-2}, d_i)$.

Output: $(Q_{i-1}, Q_i, P_i, r_{i-1}, d_{i+1})$.

Algorithm:

```
$a_{i-1} := (P_{i-1} + d) \quad (\mathrm{div}\ Q_{i-1})$;
$r_{i-1} := (P_{i-1} + d) \quad (\mathrm{mod}\ Q_{i-1})$;
$P_i := d - r_{i-1}$;
$Q_i := Q_{i-2} + a_{i-1}(r_{i-1} - r_{i-2})$;
$d_{i+1} := d_i + \deg(a_{i-1})$.
```

Clearly $\mathfrak{a}_{i+1} = (Q_i, P_i) \in \mathcal{R}$ and $d_{i+1} = \delta(\mathfrak{a}_{i+1}, \mathfrak{a}_1)$.

Note that each iteration of BABYSTEP requires only a fixed number of polynomial operations, and by Lemma 3.2, the degree of all occurring polynomials is bounded by $\deg(D)/2$. As in the case for real quadratic number fields, $\deg(a_i)$ will generally be very small, mostly 1.

For our protocol, we need to find for $\mathfrak{r} \in \mathcal{R}$ with distance $\delta(\mathfrak{r})$ and $n \in \mathbb{N}$ the reduced ideal $\mathfrak{s}$ closest to the left of $n\delta(\mathfrak{r})$. To compute $\mathfrak{s}$, we could perform repeated Baby steps, starting at $\mathfrak{r}_1 = \mathcal{O}$, until we obtain $\mathfrak{r}_j \in \mathcal{R}$ such that $\delta_j \leq n\delta(\mathfrak{r}) \leq \delta_{j+1}$, i.e. $\mathfrak{r}_j = \mathfrak{s}$. However, since by (3.8) each Baby step advances us at most $\deg(D)/2$ in distance, this requires exponential computation time if n is polynomial in $|D|$. In order to move through $\mathcal{R}$ at a much more rapid pace and thus find $\mathfrak{s}$ in time polynomial in $\deg(D)$, we make use of Shanks' *infrastructure* concept.

3.2. *Giant Steps*

As in the case of real quadratic number fields [12], we define an operation $*$ (*multiply & reduce*) on $\mathcal{R}$ under which $\mathcal{R}$ is closed. For $\mathfrak{r}_i, \mathfrak{r}_j \in \mathcal{R}$ with respective distances δ_i and δ_j $(i, j \in \mathbb{N})$, the ideal $\mathfrak{r}_i * \mathfrak{r}_j$ is reduced and satisfies $\delta(\mathfrak{r}_i * \mathfrak{r}_j) \approx \delta_i + \delta_j$, so $\mathfrak{r}_i * \mathfrak{r}_j = \mathfrak{r}_k$ for some $k \in \mathbb{N}$ where $k \approx i + j$. $\mathfrak{r}_i * \mathfrak{r}_j$ is defined as follows. Compute the ideal product $(S)\mathfrak{c} = \mathfrak{r}_i\mathfrak{r}_j$ as defined in Section 2.2, where $S \in \mathbb{F}_q[x]$ is such that $\mathfrak{c}$ is primitive. $\mathfrak{c}$ need not be reduced, but by Lemma 3.3 b), applying $\max\{1, \frac{1}{2}\deg(Q) - \frac{1}{4}\deg(D) + 2\}$ Baby step operations to the ideal $\mathfrak{c} = (Q, P)$ produces a reduced ideal $\mathfrak{r}_k$, which we define to be $\mathfrak{r}_i * \mathfrak{r}_j$, such that $\delta_k = \delta_i + \delta_j + \epsilon$ and $2 - \deg(D) \leq \epsilon \leq 0$ (see Theorem II.5.1 in [16]), so in general, ϵ is very small compared to δ_i and δ_j. The computation of $\mathfrak{r}_k$ from $\mathfrak{r}_i$ and $\mathfrak{r}_j$ is called a *Giant step*. $\mathfrak{r}_k$ may not yet be the ideal $\mathfrak{r}$ closest to the left of $\delta_i + \delta_j$, but we will see in Section 4 that we can obtain $\mathfrak{r}$ by applying $O(\deg(D))$ many Baby steps to $\mathfrak{r}_k$.

We can apply the method described above repeatedly to compute for any $n \in \mathbb{N}$ and $\mathfrak{r} \in \mathcal{R}$ with distance $\delta(\mathfrak{r})$ the reduced ideal $\mathfrak{s}$ closest to the left of $n\delta(\mathfrak{r})$. The number of iterations required is approximately n. By adapting a well-known exponentiation technique based on repeated squaring (see for example Algorithm 1.2.3 in [4]), this can be reduced to $O(\log n)$. We will now describe the ideal multiplication and reduction process in more detail.

Algorithm *MULT*

Input: $\mathfrak{a} = (Q_a, P_a)$, $\mathfrak{b} = (Q_b, P_b) \in \mathcal{R}$.

Output: $\mathfrak{c} = (Q_c, P_c) \in \mathcal{P}$, $S \in \mathbb{F}_q[x]$ *such that* $(S)\mathfrak{c} = \mathfrak{a}\mathfrak{b}$.

Algorithm:

```
1. Solve $S_1 = \gcd(Q_a, Q_b) \equiv X_1 Q_a \pmod{Q_b}$ for $S_1, X_1 \in \mathbb{F}_q[x]$;
```

```
2. Solve $S = \gcd(S_1, P_a + P_b) = X_2S_1 + Y_2(P_a + P_b)$ for $S, X_2, Y_2 \in \mathbb{F}_q[x]$;
   (If $S_1 = 1$, then set $X_2 := 1$, $Y_2 := 0$, $S := 1$);
3. Set $Q_c := \frac{Q_aQ_b}{S^2}$;
4. Set $P_c :\equiv P_a + \frac{Q_a}{S}(X_2X_1(P_b - P_a) + Y_2\frac{D-P_a^2}{Q_a})$ (mod $Q_c$);
```

THEOREM 3.4 *The parameters* $\mathfrak{c} \in \mathcal{P}$ *and* $S \in \mathbb{F}_q[x]$ *computed by Algorithm MULT satisfy* $(S)\mathfrak{c} = \mathfrak{ab}$. *Furthermore,* $\deg(S) < \deg(D)/2$ *and* $\deg(P_c) < \deg(Q_c) < \deg(D)$, *and Algorithm MULT performs* $O(\deg(D))$ *polynomial arithmetic operations.*

Proof. $(S)\mathfrak{c} = \mathfrak{ab}$ follows from Section II.2 in [16]. Since $\mathfrak{a}$ and $\mathfrak{b}$ are reduced, $\deg(S) \leq \deg(Q_a), \deg(Q_b) < \deg(D)/2$ by Lemma 3.2 c). Furthermore, $\deg(P_c) < \deg(Q_c) \leq \deg(Q_a) + \deg(Q_b) < \deg(D)$.

The algorithm performs a fixed number of polynomial operations and two applications of the Extended Euclidean Algorithm for polynomials in $\mathbb{F}_q[x]$. The number of polynomial operations required by the Extended Euclidean Algorithm is linear in the degree of the largest polynomial, i.e. $O(\deg(D))$. ■

We can now move through the set $\mathcal{R}$ of reduced ideals in Giant steps.

Algorithm *GIANTSTEP*

Input: $\mathfrak{a} = (Q_a, P_a) \in \mathcal{R}$, $\mathfrak{b} = (Q_b, P_b) \in \mathcal{R}$.

Output: $\mathfrak{r} = (Q, P) \in \mathcal{R}$, $\epsilon \in \mathbb{Z}_{\leq 0}$ *such that* $\epsilon = \delta(\mathfrak{r}) - \delta(\mathfrak{a}) - \delta(\mathfrak{b})$.

Algorithm:

```
1. $(\mathfrak{c}, S) := MULT(\mathfrak{a}, \mathfrak{b})$, so $(S)\mathfrak{c} = \mathfrak{ab}$, $\mathfrak{c} = (Q_c, P_c) \in \mathcal{R}$, $S \in \mathbb{F}_q[x]$;
2. If $\deg(Q_c) < \frac{1}{2}\deg(D)$, then set $\mathfrak{r} = \mathfrak{c}$, $\epsilon = 0$, and STOP ;
3. { Initial Baby step }    Set

      $j := 1$;        $Q'_{j-1} := Q_c$;        $P'_{j-1} := P_c$;
      $a'_{j-1} := (P'_{j-1} + d)$ (div $Q'_{j-1}$);     $r'_{j-1} := (P'_{j-1} + d)$ (mod $Q'_{j-1}$);
      $P'_j := a'_{j-1}Q'_{j-1} - P'_{j-1}$;       $Q'_j := \frac{D - P'^2_j}{Q'_{j-1}}$;
      $d_{j+1} := -\deg(Q'_{j-1})$;
```

```
4. While $\deg(Q'_j) \geq \frac{1}{2}\deg(D)$ do        { Perform Baby steps }
       $j := j + 1;$
       $a'_{j-1} := (P'_{j-1} + d) \pmod{\text{div } Q'_{j-1}};$     $r'_{j-1} := (P'_{j-1} + d) \pmod{Q'_{j-1}};$
       $P'_j := d - r'_{j-1};$     $Q'_j := Q'_{j-1} + a'_{j-1}(r'_{j-1} - r'_{j-2});$
       $d_{j+1} := d_j + \deg(a'_{j-1});$
   end while
5. Set $Q := Q'_j$;   $P := P'_j$;   $\mathfrak{r} = (Q, P)$;   $\epsilon := d_{j+1} - \deg(S) + \deg(Q'_j)$;
```

Note that the computation in step 4) is exactly the same as the Baby step arithmetic in Algorithm BABYSTEP. However, we write P'_{j-1}, Q'_{j-1}, etc. rather than P_{j-1}, Q_{j-1} to indicate that the Baby steps are performed on base polynomials of non-reduced ideals. We also point out that the distances $\delta(\mathfrak{a})$, $\delta(\mathfrak{b})$, and $\delta(\mathfrak{r})$ are not explicitly used in the algorithm, so knowledge of these quantities is not required.

THEOREM 3.5 *The ideal* $\mathfrak{r}$ *computed by Algorithm GIANTSTEP is reduced. Furthermore, we have* $2 - \deg(D) \leq \epsilon \leq 0$ *and* $|d_j| \leq \deg(D)$ *throughout steps 3) and 4). All polynomials computed in steps 3) and 4) have degrees bounded by* $\deg(D)$, *and the number of polynomial operations performed by the algorithm is* $O(\deg(D))$.

Proof. Let j be the first index such that $\deg(Q'_j) < \deg(D)/2$, i.e. the value of the index j where the loop in step 4) is exits. Then by Lemma 3.1, $\mathfrak{r} = (Q'_j, P'_j)$ is reduced. Now $d_{j+1} = -\deg(Q'_0) + \sum_{i=1}^{j-1}\deg(a'_i)$. If we set $\mathfrak{a}_1 = \mathfrak{c}$, then $\mathfrak{r} = \mathfrak{a}_{j+1}$ where $(Q'_0\theta_{j+1})\mathfrak{a}_{j+1} = (Q'_j)\mathfrak{a}_1$ by (3.6), so $(SQ'_0)\mathfrak{r} = (Q'_j)\mathfrak{a}\mathfrak{b}$. Then $\epsilon = d_{j+1} - \deg(S) + \deg(Q'_j) = \deg(Q'_j) - \deg(Q'_0) + \sum_{i=1}^{j-1}\deg(a'_i) = \deg(\overline{\theta}_{j+1}) - \deg(S)$ by (3.6). Therefore by Theorem II.5.1 in [16], $\epsilon = \delta(\mathfrak{r}) - \delta(\mathfrak{a}) - \delta(\mathfrak{b})$ and $2 - \deg(D) \leq \epsilon \leq 0$. Furthermore, for all $i \in \{2, 3, \ldots, j\}$:

$$\begin{aligned} -\deg(D) < -\deg(Q'_0) &= d_2 \leq d_i < d_{j+1} \\ &= \epsilon - \deg(Q'_j) + \deg(S) \leq \deg(S) < \frac{1}{2}\deg(D), \end{aligned}$$

so $|d_i| < \deg(D)$ for all $i \in \{2, 3, \ldots, j + 1\}$. From Theorem 3.4, $\deg(P'_0) < \deg(Q'_0) < \deg(D)$. From Theorem II.4.6 of [16], we see that for all $i \in \{1, 2, \ldots, j - 1\}$, $|Q'_i|/|Q'_0| \leq |\theta_{i+1}| \leq 1$, so $\deg(Q'_i) \leq \deg(Q'_0)$. Furthermore, from step 3) in the algorithm, $P'_i = d - r'_{i-1}$ where $|r'_{i-1}| < |Q'_{i-1}| \leq |Q'_0|$, so $\deg(P'_i) \leq \max\{\deg(Q'_0), \deg(d)\}$. Finally, $\deg(Q'_j) < \deg(P'_j) = \deg(D)/2$, since $\mathfrak{a}_{j+1}$ is reduced.

Now step 1) requires $O(\deg(D))$ polynomial operations by Theorem 3.4. By Lemma 3.3 a), we must have $\deg(Q'_j) < \deg(D)/2$ after at most $\max\{1, \frac{1}{2}\deg(Q'_0) - \frac{1}{4}\deg(D) + 2\} = O(\deg(D))$ iterations. ■

Now if $\mathfrak{r}_i$ and $\mathfrak{r}_j$ are reduced ideals, then Algorithm GIANTSTEP computes a reduced ideal $\mathfrak{r}_k$ such that $\delta_k = \delta_i + \delta_j + \epsilon$ where $2 - \deg(D) \leq \epsilon \leq 0$, so $\delta_i + \delta_j - (\deg(D) - 2) \leq \delta_k \leq \delta_i + \delta_j$, While the ideal $\mathfrak{r}_k$ is already "considerably close" to the left of $\delta_i + \delta_j$, it need not be the closest such ideal. Since by (3.8), an advance in distance of at least 1 is gained in each Baby step, it requires no more than $O(\deg(D))$ Baby steps to compute the ideal closest to the left of $\delta_i + \delta_j$. Repeated application of this technique to $\mathfrak{r}_i$ will then enable us to find the ideal closest to the left of $n\delta_i$ for any $n \in \mathbb{N}$. The details of this computation are given in the next section.

4. Computing Closest Ideals

The following algorithm advances from a given reduced ideal $\mathfrak{r}$ a certain length $k > 0$ in the set $\mathcal{R}$ of reduced ideals.

Algorithm *CLOSESTINT*

```
Input: $\mathfrak{r} = (Q, P) \in \mathcal{R}$, $k \in \mathbb{Z}_{\geq 0}$

Output: $\mathfrak{s} \in \mathcal{R}$, $f \in \mathbb{Z}_{\leq 0}$ such that $\delta(\mathfrak{s}) \leq \delta(\mathfrak{r}) + k$ and $f = \delta(\mathfrak{s}) - \delta(\mathfrak{r}) - k$ is maximal.

Algorithm:

1. (a) Set $d_2 := \frac{1}{2}\deg(D) - \deg(Q)$;
   (b) If $d_2 > k$, then set $\mathfrak{s} := \mathfrak{r}$, $f := -k$, and STOP ;
2. { Initial Baby step }    Set
       $j := 1$;    $Q_{j-1} := Q$;    $P_{j-1} := P$;
       $a_{j-1} := (P_{j-1} + d) \pmod{\text{div } Q_{j-1}}$;    $r_{j-1} := (P_{j-1} + d) \pmod{Q_{j-1}}$;
       $P_j := a_{j-1}Q_{j-1} - P_{j-1}$;    $Q_j := \frac{D - P_j^2}{Q_{j-1}}$;
3. While $d_{j+1} \leq k$ do    { perform Baby steps }
       $j := j + 1$;
       $(Q_{j-1}, Q_j, P_j, r_{j-1}, d_{j+1}) := BABYSTEP(Q_{j-2}, Q_{j-1}, P_{j-1}, r_{j-2}, d_j)$;
   end while
4. Set $\mathfrak{s} := (Q_{j-1}, P_{j-1})$;    $f := d_j - k$;
```

Note that $\mathfrak{s}$ is the reduced ideal such that the distance $\delta(\mathfrak{s}, \mathfrak{r})$ between $\mathfrak{s}$ and $\mathfrak{r}$ is maximal and $\delta(\mathfrak{s}, \mathfrak{r}) \leq k$. Furthermore, $\delta(\mathfrak{r})$ and $\delta(\mathfrak{s})$ are not explicitly used and thus need not be known for this algorithm.

THEOREM 4.1 *The ideal $\mathfrak{s}$ computed by Algorithm CLOSESTINT is the ideal closest to the left of $\delta(\mathfrak{r}) + k$, i.e. $\delta(\mathfrak{s}) \leq \delta(\mathfrak{r}) + k$ and $\delta(\mathfrak{s})$ is maximal. Furthermore, $-k \leq f \leq 0$ and $0 < d_j \leq k$ for all $j \geq 2$, except for the value of d_{j+1} computed in the last iteration of* *step 3) which satisfies* $0 < d_{j+1} \leq k + \deg(D)/2$. *Finally, the total number of polynomial operations performed by the Algorithm is* $O(k)$.

Proof. Let $\mathfrak{a}_1 = \mathfrak{r}, \mathfrak{a}_2, \ldots, \mathfrak{a}_s$ be the sequence of reduced ideals computed by Algorithm CLOSESTINT. Then by (3.7), in steps 2) and 3) we have $d_i = \delta(\mathfrak{a}_i, \mathfrak{a}_1)$ $(2 \leq i \leq s)$, and the algorithm computes the ideal $\mathfrak{s} = \mathfrak{a}_s$ where $d_s \leq k < d_{s+1}$, so $\delta(\mathfrak{a}_s) \leq \delta(\mathfrak{r}) + k < \delta(\mathfrak{a}_{s+1})$. Hence, $\mathfrak{s}$ is the reduced ideal closest to the left of $d(\mathfrak{r})+k$. Now $0 \geq f = \delta(\mathfrak{a}_s, \mathfrak{a}_1) - k \geq -k$ and $0 < d_2 = \frac{1}{2}\deg(D) - \deg(Q) \leq d_i \leq d_s \leq k$ for $2 \leq i \leq s$, $d_{s+1} \leq d_s + \deg(D)/2 \leq k + \deg(D)/2$. Finally, since $1 \leq d_s \leq k$ and $d_{i+1} - d_i \geq 1$ for $2 \leq i \leq s$, the loop in step 3) is executed $k - 1$ times. ■

We can now compute for $\mathfrak{a}, \mathfrak{b} \in \mathcal{R}$ the ideal closest to the left of $\delta(\mathfrak{a}) + \delta(\mathfrak{b})$. Note that again, this does not explicitly use or require knowledge of $\delta(\mathfrak{a})$ or $\delta(\mathfrak{b})$.

Algorithm *CLOSESTSUM*

Input: $\mathfrak{a}, \mathfrak{b} \in \mathcal{R}$.

Output: $\mathfrak{c} \in \mathcal{R}$, $f \in \mathbb{Z}_{\leq 0}$ *such that* $\delta(\mathfrak{c}) \leq \delta(\mathfrak{a}) + \delta(\mathfrak{b})$ *and* $f = \delta(\mathfrak{c}) - \delta(\mathfrak{a}) - \delta(\mathfrak{b})$ *is maximal.*

Algorithm:

```
1. $(\mathfrak{r}, \epsilon) := GIANTSTEP(\mathfrak{a}, \mathfrak{b})$, so $\epsilon = \delta(\mathfrak{r}) - \delta(\mathfrak{a}) - \delta(\mathfrak{b})$;
2. $(\mathfrak{c}, f) := CLOSESTINT(\mathfrak{r}, -\epsilon)$, so $f = \delta(\mathfrak{c}) - \delta(\mathfrak{r}) + \epsilon = \delta(\mathfrak{c}) - \delta(\mathfrak{a}) - \delta(\mathfrak{b})$ and
   $f$ is maximal;
```

Here, f corresponds to the number of Baby steps required to obtain the ideal closest to the left of $\delta(\mathfrak{a}) + \delta(\mathfrak{b})$ from the (primitive) product ideal $\mathfrak{c}$ where $(S)\mathfrak{c} = \mathfrak{a}\mathfrak{b}$.

THEOREM 4.2 *The ideal* $\mathfrak{c}$ *computed by Algorithm CLOSESTSUM is the reduced ideal closest to the left of* $\delta(\mathfrak{a}) + \delta(\mathfrak{b})$, *i.e.* $\delta(\mathfrak{c}) \leq \delta(\mathfrak{a}) + \delta(\mathfrak{b})$ *and* $\delta(\mathfrak{c})$ *is maximal. Furthermore,* $2 - \deg(D) \leq \epsilon \leq f \leq 0$, *and the algorithm performs* $O(\deg(D))$ *polynomial operations.*

Proof. By the previous theorem, $f \leq 0$, so $\delta(\mathfrak{c}) \leq \delta(\mathfrak{a}) + \delta(\mathfrak{b})$, and since f is maximal, the algorithm generates the desired ideal. Now $2 - \deg(D) \leq \epsilon \leq 0$ by Theorem 3.5 and $\epsilon \leq f \leq 0$ by Theorem 4.1. Finally, step 1) requires $O(\deg(D))$ polynomial operations by Theorem 3.5 and step 2) requires $O(-\epsilon) = O(\deg(D))$ polynomial operations. ■

Using repeated applications of Algorithm CLOSESTSUM, we can adapt the repeated squaring exponentiation technique mentioned earlier to compute for $\mathfrak{a} \in \mathcal{R}$ and $n \in \mathbb{N}$ the reduced ideal closest to the left of $n\delta(\mathfrak{a})$.

Algorithm *BINARY*

Input: $i \in \{0, 1\}$, $\mathfrak{a}, \mathfrak{b} \in \mathcal{R}$, $f \in \mathbb{Z}_{\leq 0}$ *such that* $\delta(\mathfrak{b}) \leq s\delta(\mathfrak{a})$ *for some* $s \in \mathbb{N}$ *and* $f = \delta(\mathfrak{b}) - s\delta(\mathfrak{a})$ *is maximal.*

Output: $\mathfrak{c} \in \mathcal{R}$, $l \in \mathbb{Z}_{\leq 0}$ *such that* $\delta(\mathfrak{c}) \leq (2s+i)\delta(\mathfrak{a})$ *and* $l = \delta(\mathfrak{c}) - (2s+i)\delta(\mathfrak{a})$ *is maximal.*

Algorithm:

1. $(\mathfrak{m}, g) :=$ *CLOSESTSUM*$(\mathfrak{b}, \mathfrak{b})$, *so* $\delta(\mathfrak{m}) \leq 2\delta(\mathfrak{b})$ *and* $g = \delta(\mathfrak{m}) - 2\delta(\mathfrak{b})$ *is maximal* ;
2. $(\mathfrak{n}, h) :=$ *CLOSESTINT*$(\mathfrak{m}, -(g+2f))$, *so* $\delta(\mathfrak{n}) \leq \delta(\mathfrak{m}) - (g+2f)$ *and* $h = \delta(\mathfrak{n}) - \delta(\mathfrak{m}) + g + 2f$ *is maximal* ;
3. *If* $i = 0$, *then set* $\mathfrak{c} := \mathfrak{n}$, $l = h$, *and STOP* ;
4. { Now $i = 1$ }
   $(\mathfrak{q}, k) :=$ *CLOSESTSUM*$(\mathfrak{a}, \mathfrak{n})$, *so* $\delta(\mathfrak{q}) \leq \delta(\mathfrak{a}) + \delta(\mathfrak{n})$ *and* $k = \delta(\mathfrak{q}) - \delta(\mathfrak{a}) - \delta(\mathfrak{n})$ *is maximal* ;
5. $(\mathfrak{c}, l) :=$ *CLOSESTINT*$(\mathfrak{q}, -(k+h))$, *so* $\delta(\mathfrak{c}) \leq \delta(\mathfrak{q}) - (k+h)$ *and* $l = \delta(\mathfrak{c}) - \delta(\mathfrak{q}) + k + h$ *is maximal* ;

Note that s is not explicitly used in this algorithm.

THEOREM 4.3 *The ideal* $\mathfrak{c}$ *computed by Algorithm BINARY is the ideal closest to the left of* $(2s+i)\delta(\mathfrak{a})$, *i.e.* $\delta(\mathfrak{c}) \leq (2s+i)\delta(\mathfrak{a})$ *and* $\delta(\mathfrak{c})$ *is maximal. Furthermore,* $|g|, |h|, |k|, |l| = O(\max\{\deg(D), |f|\})$ *and the algorithm performs* $O(\max\{\deg(D), |f|\})$ *polynomial operations.*

Proof. To prove that $\mathfrak{c}$ is the desired ideal, it suffices to show that l as defined in steps 3) and 5) is equal to $\delta(\mathfrak{c}) - (2s+i)\delta(\mathfrak{a})$.

If $i = 0$, then from step 3) we obtain:

$$\begin{aligned} l &= h = \delta(\mathfrak{n}) - \delta(\mathfrak{m}) + g + 2f \\ &= \delta(\mathfrak{n}) - \delta(\mathfrak{m}) + (\delta(\mathfrak{m}) - 2\delta(\mathfrak{b})) + 2(\delta(\mathfrak{b}) - s\delta(\mathfrak{a})) \\ &= \delta(\mathfrak{c}) - 2s\delta(\mathfrak{a}). \end{aligned}$$

If $i = 1$, the from step 5) we obtain:

$$\begin{aligned} l &= \delta(\mathfrak{c}) - \delta(\mathfrak{q}) + k + h \\ &= \delta(\mathfrak{c}) - \delta(\mathfrak{q}) + (\delta(\mathfrak{q}) - \delta(\mathfrak{a}) - \delta(\mathfrak{n})) + (\delta(\mathfrak{n}) - \delta(\mathfrak{m}) + g + 2f) \\ &= \delta(\mathfrak{c}) - \delta(\mathfrak{a}) - \delta(\mathfrak{m}) + (\delta(\mathfrak{m}) - 2\delta(\mathfrak{b})) + 2(\delta(\mathfrak{b}) - s\delta(\mathfrak{a})) \\ &= \delta(\mathfrak{c}) - (2s+1)\delta(\mathfrak{a}). \end{aligned}$$

Next, we check whether all quantities satisfy the requirements for the input parameters of the algorithms CLOSESTINT and CLOSESTSUM and establish the bounds on g, h, k, and l.

From Theorem 4.2, $2 - \deg(D) \leq g \leq 0$. Now $f \leq 0$, so $g + 2f \leq 0$ and the inputs for CLOSESTSUM in step 2) are well-defined. Furthermore, by Theorem 4.1, $g + 2f \leq h \leq 0$, so $|h| \leq \deg(D) - 2 + 2|f|$. As before, $2 - \deg(D) \leq k \leq 0$, so $h + k \leq 0$, and the input parameters for CLOSESTINT in step 5) are again well-defined. Finally, by Theorem 4.1, $h+k \leq l \leq 0$, so $|l| \leq \deg(D) - 2 + |h| \leq 2(\deg(D) - 2 + |f|)$.

Now steps 1) and 4) of the algorithm require $O(\deg(D))$ polynomial operations by Theorem 4.2. Steps 3) and 5) perform $O(|g+2f|)$ and $O(|k+h|)$ operations, respectively. By Theorem 4.2, $|g|, |k| = O(\deg(D))$, so $|g+2f| = O(\max\{\deg(D), |f|\})$, and since $|h| \leq |g+2f|$, also $|k+h| = O(\max\{\deg(D), |f|\})$. ■

Algorithm *POWER*

Input: $\mathfrak{a} \in \mathcal{R}$, $n \in \mathbb{N}$.

Output: $\mathfrak{b} \in \mathcal{R}$ *such that* $\delta(\mathfrak{b}) \leq n\delta(\mathfrak{a})$ *and* $f = \delta(\mathfrak{b}) - n\delta(\mathfrak{a})$ *is maximal.*

Algorithm:

```
1. Compute the binary representation $n = \sum_{i=0}^{t} b_i 2^{t-i}$ of $n$ where $b_0 = 1$, $b_i \in \{0, 1\}$
   for $1 \leq i \leq t$;
2. Set $\mathfrak{b}_0 := \mathfrak{a}$;   $s_0 := 1$;   $f_0 := 0$;
3. For $i := 1$ to $n$ do
   { At this point $\delta(\mathfrak{b}_{i-1}) \leq s_{i-1}\delta(\mathfrak{a})$ and $f_{i-1} = \delta(\mathfrak{b}_{i-1}) - s_{i-1}\delta(\mathfrak{a})$ is maximal }
   (a) $s_i := 2s_{i-1} + b_i$;
   (b) $(\mathfrak{b}_i, f_i) := BINARY(b_i, \mathfrak{a}, \mathfrak{b}_{i-1}, f_{i-1})$, so $\mathfrak{b}_i \in \mathcal{R}$, $f_i \in \mathbb{Z}_{\leq 0}$ such that
       $\delta(\mathfrak{b}_i) \leq s_i\delta(\mathfrak{a})$ and $f_i = \delta(\mathfrak{b}_i) - s_i\delta(\mathfrak{a})$ is maximal;
   endfor
4. Set $\mathfrak{b} := \mathfrak{b}_t$;
```

THEOREM 4.4 *The ideal computed by Algorithm POWER is the reduced ideal closest to the left of* $n\delta(\mathfrak{a})$, *i.e.* $\delta(\mathfrak{b}) \leq n\delta(\mathfrak{a})$ *and* $\delta(\mathfrak{b})$ *is maximal. Furthermore,* $1 \leq s_i \leq n$, $|f_i| = O(\deg(D))$ *for* $0 \leq i \leq t$, *and the algorithm performs* $O(\deg(D)\log n)$ *polynomial operations.*

Proof. $\delta(\mathfrak{b}) = \delta(\mathfrak{b}_t) \leq s_t\delta(\mathfrak{a}) = n\delta(\mathfrak{a})$ and $f_t = f$ is maximal, so $\mathfrak{b}$ is the desired ideal. Now $s_i = \sum_{j=0}^{i} b_i 2^{t-i}$, so $1 \leq s_{i-1} \leq s_i \leq n$ for $1 \leq i \leq t$. Furthermore, $f_0 = 0$ and by Theorem 4.3, $|f_i| = O(\max\{\deg(D), |f_{i-1}|\}) = O(\deg(D))$ for $0 \leq i \leq t$.

Now each call of BINARY in step 3b) uses $O(\deg(D))$ polynomial operations, so the entire loop requires $O(\deg(D)t) = O(\deg(D)\log n)$ polynomial operations. ■

If n is polynomially bounded by $|D|$, then we can compute the ideal closest to the left of $n\delta(\mathfrak{a})$ in $O((\deg(D))^2 \log q)$ polynomial operations. Hence both communication partners must bound their respective "exponents" by a polynomial in $|D|$, say $|D|^r$. We consider a choice of $r = 1/4$ sufficiently secure if $|D|$ is large, say $|D| \approx 10^{200}$ (see Section 6.1).

Finally, in order to exchange a reduced ideal that is to be used as a cryptographic key, the two parties require one more algorithm, which is an extension of the previous algorithm.

Algorithm *POWERDIST*

```
Input: $\mathfrak{a} \in \mathcal{R}$, $n \in \mathbb{N}$, $\delta_a \in \mathbb{N}$ where $\delta_a = \delta(\mathfrak{a})$.
Output: $\mathfrak{b} \in \mathcal{R}$, $\delta_b \in \mathbb{N}$ such that $\delta_b = \delta(\mathfrak{b}) \leq n\delta(\mathfrak{a})$ and $\delta_b$ is maximal.
Algorithm:
1. $\mathfrak{b} := POWER(\mathfrak{a}, n)$, so $\delta(\mathfrak{b}) \leq n\delta(\mathfrak{a})$ and $f = \delta(\mathfrak{b}) - n\delta(\mathfrak{a})$ is maximal ;
2. $\delta_b := n\delta_a + f$;
```

5. The Protocol

Precomputation: Alice and Bob

1. generate an odd prime power q
2. generate a random squarefree polynomial $D \in \mathbb{F}_q[x]$ of even degree whose leading coefficient is a square in $\mathbb{F}_q$
3. compute $d = \lfloor \sqrt{D} \rfloor$
4. generate a reduced ideal $\mathfrak{c} = (Q, P)$ with small distance $\delta = \delta(\mathfrak{c})$ by applying Algorithm BABYSTEP to the ideal $\mathcal{O} = (1, 0)$ a few times
5. publicize (q, D, d, P, Q, δ)

Protocol:

1. Alice
   (a) secretly generates $a \in \mathbb{N}$, $a < |D|^{1/4}$
   (b) computes $(\mathfrak{a}, \delta_a) := POWERDIST(\mathfrak{c}, \delta, a)$, $\mathfrak{a} = (Q_a, P_a)$
   (c) transmits (Q_a, P_a) to Bob
2. Bob
   (a) secretly generates $b \in \mathbb{N}$, $b < |D|^{1/4}$
   (b) computes $(\mathfrak{b}, \delta_b) := POWERDIST(\mathfrak{c}, \delta, b)$, $\mathfrak{b} = (Q_b, P_b)$
   (c) transmits (Q_b, P_b) to Alice
3. Alice computes $\mathfrak{k} := POWER(\mathfrak{b}, \delta_a)$
4. Bob computes $\mathfrak{k} := POWER(\mathfrak{a}, \delta_b)$

Both partners compute the reduced ideal $\mathfrak{k}$ closest to the left of $\delta(\mathfrak{a})\delta(\mathfrak{b})$. However, their respective basis polynomials need not be the same. By multiplying $\mathfrak{k}$ by a suitable element in $\mathbb{F}_q$ to achieve $\text{sgn}(Q_k) = 1$ and then reducing P_k modulo Q_k such that $\deg(P_k) < \deg(Q_k)$, the base representation (Q_k, P_k) of the ideal $\mathfrak{k}$ is unique. The coefficients of these polynomials (or any substring thereof) can then be used as the key. The protocol requires one round of communicating two polynomials of degree at most $\deg(D)/2$ each, hence the number of bits that must be transmitted is at most $(\deg(D) + 2)\log q$.

6. Security

6.1. Choice of Parameters

To prevent an exhaustive key-search attack, we need to ensure that the number m of reduced principal ideals in K is large. Since $R = \delta_{m+1}$, we have $R \leq m \cdot \deg(d)$ by Lemma 3.2 c) and (3.7), or equivalently

$$m \geq \frac{2R}{\deg(D)}.$$

Thus, to get a lower bound on m, we require a lower bound on R.

By using standard results on zeta functions for function fields (see Eichler [9], pp. 299–307), we can bound the value of the divisor class number h by

$$\left(\sqrt{q} - 1\right)^{2g} \leq h \leq \left(\sqrt{q} + 1\right)^{2g},$$

where, in this case, $g = \frac{1}{2}\deg(D) - 1$. Since $h = Rh'$, we see that for fixed h, R will be large as long as h' is small. We can use a result of Zhang [21] to ensure that h' is odd. Namely, if D is prime or a product of two even degree prime polynomials, then $2 \nmid h'$.

In order to examine the odd part of the class group G of K, we can apply the same heuristic arguments that Cohen and Lenstra [5], [6] used for real quadratic number fields. This is possible because of the complete analogy that exists between the infrastructures of the ideal classes in K and in a real quadratic number field. For example, under the same heuristic assumptions as those used in [5] and [6], we can derive the following result for real quadratic congruence function fields over $\mathbb{F}_q$. Let r be any odd prime and let G_r be the r-Sylow subgroup of the ideal class group of K. Then, if H is any fixed finite abelian r-group, we have

$$\lim_{n \to \infty} \frac{\sum\limits_{\substack{D \\ \deg(D) \leq 2n \\ G_r \cong H}} 1}{\sum\limits_{\substack{D \\ \deg(D) \leq 2n}} 1} = |H|^{-1} \, |\mathrm{Aut}(H)|^{-1} \prod_{j=2}^{\infty} \left(1 - r^{-j}\right).$$

This kind of result can be extended to show that, if $\hat{G}$ is the odd part of G, then we would have

$$\Pr\left(\left|\hat{G}\right| = k\right) = C\frac{w(k)}{k}, \tag{6.1}$$

where, in the notation of [5],

$$C = (2\eta_\infty(2)C_\infty)^{-1} \approx .754458173$$

and

$$w(k) = \prod_{p^\alpha \| k} \left(p^\alpha (1 - 1/p)\left(1 - 1/p^2\right) \dots (1 - 1/p^\alpha)\right)^{-1}.$$

Here, we have assumed that the characteristic of $\mathbb{F}_q$ behaves like any other odd prime p with respect to the p-component of G. In their investigation of the structure of the divisor class group of K, Friedman and Washington [10] excluded this prime because the p-rank of the divisor class group for this prime cannot be as large as that for other primes; however, there seems to be no *a priori* reason for excluding it in an investigation of the ideal class group.

From (6.1) it can be shown that

$$\Pr\left(\left|\hat{G}\right| > \kappa\right) = \frac{1}{2\kappa} + O\left(\frac{\log(\kappa)}{\kappa^2}\right).$$

Thus, we would expect the probability that $|\hat{G}|$ is small to be very close to 1. Indeed, this is what has actually been observed in extensive computations of h' carried out by one of the authors (A. Stein). Thus, if D is a randomly selected prime polynomial, it is most likely that our scheme will be secure against an exhaustive search attack.

6.2. *The DLP for Real Quadratic Congruence Function Fields*

By analogy to the number field case, we can define the *discrete logarithm problem* (DLP) *for real quadratic congruence function fields* as follows: For any $\mathfrak{r} \in \mathcal{R}$, find $\delta(\mathfrak{r})$, $0 \leq \delta(\mathfrak{r}) < R$.

Note that we can solve any instance of the DLP by applying $O(\delta(\mathfrak{r}))$ Baby steps to the ideal $\mathfrak{r}_1 = \mathcal{O}$. For large distances $\delta(\mathfrak{r})$, this is exponential in $\deg(D)$. As in the number field case, we can conclude that a cryptanalyst can break our scheme, if he is able to solve the DLP. We formulate our results in terms of polynomial time solutions, but our conclusions are not restricted to polynomial-time algorithms.

LEMMA 6.1 *If there is a polynomial time solution of the DLP, then the key exchange protocol can be broken in polynomial time.*

Proof. Suppose A is a polynomial-time algorithm that solves any instance of the DLP. Then an eavesdropper intercepting the reduced ideal $\mathfrak{a}$ sent by Alice can use A to compute $\delta(\mathfrak{a})$. Then he intercepts the reduced ideal $\mathfrak{b}$ sent by Bob and uses Algorithm POWER on inputs $\mathfrak{b}$ and $\delta(\mathfrak{a})$ to compute the reduced ideal $\mathfrak{k}$ closest to the left of $\delta(\mathfrak{b})\delta(\mathfrak{a})$. This ideal is the secret key. ■

It is not known whether the DLP is in fact equivalent to the difficulty of breaking the protocol in the sense that any fast method for breaking the scheme gives rise to a fast algorithm for solving the DLP.

In [1], Abel shows that the DLP in a real quadratic number field $\mathbb{Q}(\sqrt{\Delta})$ can be solved in time subexponential in $\log \Delta$. Also, any algorithm for solving the DLP can be used to find the regulator of this field. Knowledge of the regulator together with a technique due to Schoof [14] can then in turn be used to factor Δ. Hence the DLP for real quadratic number fields is at least as difficult as the problem of factoring the integer Δ.

The situation in real quadratic congruence function fields is somewhat different. Here, the only known algorithm for solving the DLP is exponential. Just as in the case of a real quadratic number field, Shanks' Baby step-Giant step technique [11] can be used to compute the distance of a reduced ideal. This method has complexity $O(q^{\frac{1}{4}\deg(D)})$.

In [17], it is shown that the DLP in real quadratic congruence function fields $\mathbb{F}_q(x)(\sqrt{D})$ where $\deg(D) = 4$, (the simplest non-trivial case, since it is known that $R = 1$ if $\deg(D) = 2$) is equivalent to the DLP for elliptic curves; that is, given an elliptic curve $E/\mathbb{F}_q$, an $\mathbb{F}_q$-rational point P and an $\mathbb{F}_q$-rational point Q such that $Q = k \cdot P$, find the integer k. Furthermore, the set of reduced principal ideals forms a group in this special case; that is, $\mathfrak{r}_s * \mathfrak{r}_t = \mathfrak{r}_{s+t}$ for $\mathfrak{r}_s, \mathfrak{r}_t \in \mathcal{R}$ and $s + t \leq m$. We now sketch the main ideas.

THEOREM 6.2 *If the DLP for real quadratic congruence function fields can be solved in polynomial time, then the DLP for elliptic curves can be solved in polynomial time.*

Proof. Let E be the elliptic curve defined by the equation

$$E \, : \, w^2 = v^3 + Av + B \, ,$$

where $A, B \in \mathbb{F}_q$ and $\Delta = -4A^3 - 27B^2 \neq 0$. Denote by $K = \mathbb{F}_q(E) = \mathbb{F}_q(v, w)$ its corresponding function field. Let O be the point at infinity which is the identity in the usual group law on E. Let $P = (x_P, y_P) \neq O$ be any $\mathbb{F}_q$-rational point on E ($x_P, y_P \in \mathbb{F}_q$). Under a suitable birational transformation (dependent on x_P, y_P) we can construct a plane quartic model for E, i.e.

$$E_P \, : \, y^2 = D_P(x) \, ,$$

where $D_P(x)$ is a monic squarefree polynomial of degree 4, E and E_P are birationally equivalent and $K = \mathbb{F}_q(x, y)$. In fact, $\mathbb{F}_q(x, y)$ is a real quadratic congruence function field. If we set $\mathfrak{r}_1 = [1, \sqrt{D_P(x)}]$ and apply repeated Baby steps to $\mathfrak{r}_1$ using the formulas of (3.2) and (3.3), we obtain a sequence of reduced ideals $(\mathfrak{r}_i)_{i \in \mathbb{N}}$. From Lemma 3.2 c) and (3.7), we deduce that $\delta_i = \delta(\mathfrak{r}_i) = i$ ($i \geq 2$). In [17], it is pointed out that there is a one-to-one correspondence between the subgroup of E generated by P and the set $(\mathfrak{r}_i)_{i \in \mathbb{N}}$. Let $Q = (x_Q, y_Q) \neq P$ be an $\mathbb{F}_q$-rational point on E such that $Q = kP$ ($k \in \mathbb{N}, k \geq 2$). Then the corresponding reduced ideal $\mathfrak{r}_Q$ can easily be computed from x_Q and y_Q. Furthermore, it is true that $\mathfrak{r}_Q = \mathfrak{r}_k$. If there is a polynomial time method for finding $\delta(\mathfrak{r}_Q)$, we are able to determine k in polynomial time, because $k = \delta_k = \delta(\mathfrak{r}_Q)$. ■

This means that the DLP for real quadratic congruence function fields is at least as difficult as the DLP for elliptic curves. So far, the only known algorithm for solving the DLP for elliptic curves is exponential (except for the supersingular case). If it should turn out that the subexponential methods of the number field case can be applied to function fields, then the DLP for elliptic curves should be of subexponential complexity. We also sketch the converse direction.

THEOREM 6.3 *If the DLP for elliptic curves can be solved in polynomial time, then the DLP for real quadratic congruence function fields* $\mathbb{F}_q(x)(\sqrt{D})$ *where* $\deg(D) = 4$ *can be solved in polynomial time.*

Proof. Let $\mathbb{F}_q(x)(\sqrt{D})$ be a real quadratic congruence function field, where $D \in \mathbb{F}_q[x]$ is a squarefree polynomial of degree 4 and $\mathrm{sgn}(D)$ is a square in $\mathbb{F}_q{}^*$. Without loss of generality, we can assume that D is monic. By applying the inverse of the birational transformation mentioned in the proof of Theorem 6.2, we obtain an elliptic curve E and an $\mathbb{F}_q$-rational point P on E. Let $\mathfrak{r}_1 = [1, \sqrt{D}]$ and let $(\mathfrak{r}_i)_{i \in \mathbb{N}}$ be the sequence of reduced ideals defined by (3.2) and (3.3). If $\mathfrak{r} \neq \mathfrak{r}_1$ is any reduced ideal, then we know that $\mathfrak{r} = \mathfrak{r}_k$ for some index $k \geq 2$. Because $\delta(\mathfrak{r}_k) = \delta_k = k$, we see that $\delta(\mathfrak{r}) = k$. As in the proof of the above theorem, we use the result of [17] that there is a one-to-one correspondence between multiples of P on E and the sequence $(\mathfrak{r}_i)_{i \in \mathbb{N}}$. The corresponding point on E, $Q_{\mathfrak{r}}$, can be determined from $\mathfrak{r}$, and we know that $Q_{\mathfrak{r}} = kP$. Hence, if we are able to solve the DLP for elliptic curves in polynomial time, then we can determine k and $\delta(\mathfrak{r}) = k$ in polynomial time. ■

7. Implementation

7.1. Implementation Issues

Our key computations were run on a Silicon Graphics Challenge workstation using the Computer Algebra System SIMATH which is based on the programming language C. SIMATH was developed by the research group of Prof. H. G. Zimmer at the Universität des Saarlandes in Saarbrücken, Germany. All our computations were done over prime fields $\mathbb{F}_p$, i.e. $q = p$ prime. For arithmetic in finite fields $\mathbb{F}_p$, where $p < 2^{30} - 1$, single precision arithmetic was sufficient. For primes $p > 2^{30} - 1$, we used multiple precision arithmetic.

Our computations were significantly faster than those for key exchange in real quadratic number fields using parameters of the same order of magnitude. This is partially due to the fact that our implementation involves rational integers only and requires no rational approximations. Furthermore, in our setting, there are two parameters which can be varied, namely the prime p and the degree of D (in the number field case, only the size of the radicand was variable). In order to achieve optimal performance for a fixed size of the key space, the sizes of the two parameters need to be weighed against each other according to computation time requirements of polynomial arithmetic vs. arithmetic in finite fields. From the table in Section 7.2, one can determine the best choice of the size of $q = p$ and $\deg(D)$. The case $\deg(D) = 4$, in which we have a direct correspondence to elliptic curves, performed best.

Since our implementation is algorithmic and computes keys for arbitrary parameters q and D, it is slower than some implementations of elliptic curve cryptosystems such as [2]. However, we believe that a significant speed-up could be achieved by making use of features such as special-purpose arithmetic, hardware implementation, and code optimization.

For our procedures, we used two optimized versions of Algorithm MULT, since our computations only require squaring of a reduced ideal and multiplication of an arbitrary reduced ideal by a given fixed one. We also noticed that in most cases the gcd in the first step of MULT produces $S_1 = 1$, in which case the second gcd calculation need not be performed.

7.2. *Numerical Examples*

The table below gives some numerical examples and their computation times. As mentioned above, we always chose $q = p$ to be a prime. For security reasons, we selected our parameters so that $p^{\frac{1}{2}\deg(D)}$ is of order of magnitude 10^{100}, although we believe that smaller

Table 1. Common Key Computations

deg(D)	p	time
4	99999999999999999999999999999999999999999999999943	3.76 sec
6	2154434690031883721759293566519517	6.58 sec
8	10000000000000000000000013	9.39 sec
10	100000000000000000039	13.10 sec
12	46415888336127803	15.66 sec
14	193069772888329	19.67 sec
16	3162277660169	25.50 sec
18	129154966537	31.70 sec
20	10000000019	33.70 sec
22	1232846819	26.33 sec
24	215443483	8.69 sec
26	49238857	9.77 sec
28	13894961	16.51 sec
30	4641631	13.11 sec
32	1778279	15.12 sec
34	762721	9.95 sec
36	359389	11.37 sec
38	183299	12.47 sec
40	100003	14.31 sec
42	57793	8.24 sec
44	35111	9.00 sec
46	22277	9.89 sec
48	14683	10.27 sec
50	10007	11.52 sec

parameters (such as $p^{\frac{1}{2}\deg(D)} \approx 10^{50}$) still provide sufficient security while resulting in a significant performance increase. If D is a random squarefree polynomial in $\mathbb{F}_p[x]$ of even degree, we used a prime-generating routine to find a prime $p \approx \exp(\ln(10) \cdot 200/\deg(D))$. For example, if $\deg(D) = 4$, then $p \approx 10^{50}$. In our table, we only give the degree of D rather than D itself. In accordance with Section 5, we chose exponents of order of magnitude $p^{\frac{1}{4}\deg(D)} \approx 10^{50}$.

Each computation time given in the table is the total time for each party to compute the common key. This is equal to the sum of the computation times required for POWER and POWERDIST, respectively, as described in the protocol. Time is recorded in seconds. The results in the table document the changing point from single to multiple precision. If p is a prime less than $2^{30} - 1$, we can see from $p^{\frac{1}{2}\deg(D)} \approx 10^{100}$ that $\deg(D) > 22$. For comparison, we computed the largest prime less than $2^{30} - 1$, $p = 1073741789$, and ran another example for $\deg(D) = 22$. Here, the running time was 7.09 seconds, which is significantly less than that for the degree 22 example given in the table.

References

1. C. S. Abel, *Ein Algorithmus zur Berechnung der Klassenzahl und des Regulators reellquadratischer Ordnungen*, Dissertation, Universität des Saarlandes, Saarbrücken (1994).
2. G. B. Agnew, R. C. Mullin and S. A. Vanstone, An implementation of elliptic curve cryptosystems over $\mathbb{F}_{2^{155}}$, *IEEE J. Selected Areas in Communications*, Vol. 11 (1993) pp. 804–813.
3. E. Artin, Quadratische Körper im Gebiete der höheren Kongruenzen I, II, *Math. Zeitschr.*, Vol. 19 (1924) pp. 153–206.
4. H. Cohen, *A Course in Computation Algebraic Number Theory*, Springer, Berlin (1994).
5. H. Cohen and H. W. Lenstra, Heuristics on class groups, in Number Theory (H. Jager, ed.) (Noordwijkerhout, 1983), Lecture Notes in Mathematics, Springer, New York, 1052 (1984) pp. 26–36.
6. H. Cohen and H. W. Lenstra, Heuristics on class groups of number fields, in Number Theory (H. Jager, ed.) (Noordwijkerhout, 1983), Lecture Notes in Mathematics, Springer, New York, 1068 (1984) pp. 33–62.
7. M. Deuring, Lectures on the Theory of Algebraic Functions of One Variable, Lecture Notes in Mathematics, Berlin 314 (1973).
8. W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Trans. Inform. Theory*, Vol. 22, No. 6, (1976) pp. 644–654.
9. M. Eichler, *Introduction to the Theory of Algebraic Numbers and Functions*, Academic Press, New York (1966).
10. E. Friedman and L. C. Washington, On the distribution of divisor class groups of curves over finite fields, *Theorie des Nombres, Proc. Int. Number Theory Conf. Laval*, 1987, Walter de Gruyter, Berlin and New York (1989) pp. 227–239.
11. H. W. Lenstra Jr., On the calculation of regulators and class numbers of quadratic fields, *London Math. Soc. Lec. Note Ser.*, Vol. 56, (1982) pp. 123–150.
12. R. Scheidler, J. A. Buchmann and H. C. Williams, A key exchange protocol using real quadratic fields, *J. Cryptology*, Vol. 7, (1994) pp. 171–199.
13. F. K. Schmidt, Analytische Zahlentheorie in Körpern der Charakteristik p, *Math. Zeitschr.*, Vol. 33 (1931) pp. 1–32.
14. R. J. Schoof, Quadratic fields and factorization, Computational Methods in Number Theory (H. W. Lenstra and R. Tijdemans, eds.), Math. Centrum Tracts, Part II, Amsterdam, 155 (1983) pp. 235–286.
15. D. Shanks, The infrastructure of a real quadratic field and its applications, *Proc. 1972 Number Theory Conf.*, Boulder, Colorado (1972) pp. 217–224.
16. A. Stein, *Baby step-Giant step-Verfahren in reell-quadratischen Kongruenzfunktionenkörpern mit Charakteristik ungleich 2*, Diplomarbeit, Universität des Saarlandes, Saarbrücken (1992).
17. A. Stein, *Equivalences between Elliptic Curves and Real Quadratic Congruence Function Fields*, in preparation.

18. A. Stein and H. G. Zimmer, An algorithm for determining the regulator and the fundamental unit of a hyperelliptic congruence function field, Proc. 1991 Int. Symp. on Symbolic and Algebraic Computation, Bonn, ACM Press, July 15-17 (1991) pp. 183–184.
19. B. Weis and H. G. Zimmer, Artin's Theorie der quadratischen Kongruenzfunktionenkörper und ihre Anwendung auf die Berechnung der Einheiten-und Klassengruppen, *Mitt. Math. Ges. Hamburg, Sond.*, Vol. XII, No. 2 (1991) pp. 261–286.
20. E. Weiss, *Algebraic Number Theory*, McGraw-Hill, New York (1963).
21. X. Zhang, Ambiguous classes and 2-rank of class groups of quadratic function fields, *J. of China University of Science and Technology*, Vol. 17, No. 4, (1987) pp. 425–431.