JOURNAL
de Théorie des Nombres
de BORDEAUX
anciennement Séminaire de Théorie des Nombres de Bordeaux

Johannes BUCHMANN et Ulrich VOLMER

A Terr algorithm for computations in the infrastructure of real-quadratic number fields

Tome 18, n° 3 (2006), p. 559-572.

<http://jtnb.cedram.org/item?id=JTNB_2006___18_3_559_0>

Journal de Théorie des Nombres
de Bordeaux **18** (2006), 559-572

A Terr algorithm for computations in the infrastructure of real-quadratic number fields

par Johannes BUCHMANN et Ulrich VOLMER

Dedicated to Michael Pohst on the occasion of his 60th birthday

Résumé. Nous montrons comment adapter la variante due à Terr de l'algorithme "baby-step giant-step" de Shanks pour le calcul du régulateur et des générateurs des idéaux principaux des corps quadratiques réels. La complexité du pire cas de l'algorithme obtenu dépend uniquement de la racine carrée du régulateur, et est plus petite que toutes celles des algorithmes inconditionnels et déterministes connus précédemment pour ce problème.

Abstract. We show how to adapt Terr's variant of the baby-step giant-step algorithm of Shanks to the computation of the regulator and of generators of principal ideals in real-quadratic number fields. The worst case complexity of the resulting algorithm depends only on the square root of the regulator, and is smaller than that of all other previously specified unconditional deterministic algorithm for this task.

1. Introduction

The baby-step giant-step idea goes back to Shanks, see [Sha71]. It can be applied to group order and discrete logarithm computations in general groups. Shanks basic goal, however, was to apply it to computations in class groups and the infrastructure of quadratic number fields, the latter also having been a discovery he made. His original algorithms assume the knowledge of an upper bound B for the group order, or the regulator, respectively. Their complexity (in terms of group operations) is in $O(\sqrt{B})$. The algorithms can be adapted to the case were no such bound is known, by taking an arbitrary small bound and squaring (or, respectively, doubling) it in case of a failure of the algorithm, cf. e.g. [BW88] and [BJT97].

More elegant is another variant of the baby-step giant-step algorithm given by Terr for the element order problem in [Ter00]. Its basic idea is not to compute all baby-steps at once, but to alternate baby- and giant-steps. The length of each giant-step equals the current size of the baby-step

Manuscrit reçu le 31 décembre 2005.

table. Hence, the length of the path covered by all giant-steps together grows quadratically. Thus, the algorithm has complexity proportional to the square root of the order of the element. Space and time requirements do not depend as strongly on the instance as is the case for the squaring technique. The latter may compute close to double the minimally necessary number of baby-steps, while the cost of the former is increased by a constant factor of $\sqrt{2}$, both in comparison to a run of the original baby-step giant-step algorithm with a good known bound B. The complexity advantage of the Terr variant is largest when the first bound B used in the doubling approach is very small and in the penultimate round the giant steps just barely fail to hit the baby-step table causing an unnecessary large number of baby-steps elements to be computed.

Here we show how to translate Terr's idea to the second domain it was intended for: computations in the infrastructure of a real-quadratic order. The resulting algorithm computes the fundamental unit ε of such an order $\mathcal{O}$ with discriminant Δ and regulator $R = \log \varepsilon$ in time $\mathrm{O}((\log \Delta + \sqrt{R})(\log \Delta)^2) = |\Delta|^{1/4+\mathrm{o}(1)}$. An extension can be used to decide equivalence of $\mathcal{O}$-ideals and to calculate generators of principal $\mathcal{O}$-ideals. If the input ideals are reduced, then the equivalence algorithm admits the same running time bound as the regulator algorithm.

In sections 2 and 3 we will introduce basic notions from the theory of quadratic number fields as well as the infrastructure in the set of reduced principal ideals of real-quadratic fields. More background on this topic can be found in [Len82] and [BV07]. For a generalization in the framework of Arakelov theory to number fields of higher degree, see [Sch04]. In section 4 we give an outline of the regulator algorithm. Details are provided in section 5 and 6 including an example for the quadratic order with $\Delta = 2521$. The analysis of the regulator algorithm is given in section 7. Throughout we have made an effort to present our algorithms in such a fashion that all operations can be performed exactly. Finally, section 8 is devoted to the extension of the regulator algorithm to the computation of generators of principal ideals, extended discrete logarithms and the structure of the class group of the order.

2. Infrastructure

Let $\mathcal{O}$ be a real quadratic order, let Δ be the discriminant of $\mathcal{O}$, and let R be the regulator of $\mathcal{O}$. We let F be the field of fractions of $\mathcal{O}$, and σ be the non-trivial automorphism of F. By $\mathcal{I}$ we denote the group of fractional invertible $\mathcal{O}$-ideals and by $\mathcal{P}$ the group of principal fractional $\mathcal{O}$-ideals. Also, ε is the fundamental unit of $\mathcal{O}$.

The set $\mathcal{I}$ can be endowed with a topology using the following notion of length for elements in F^*.

Definition 2.1. *For $\alpha \in F^*$ we set* $\operatorname{Log}\alpha = \log|\sigma(\alpha)/\alpha|$.

Indeed, it is easy to show that this length map is homomorphic and the image of the group of units of $\mathcal{O}$ is $R\mathbb{Z}$.

Proposition 2.2. *(1) The map $F^* \to \mathbb{R}$, $\alpha \mapsto \operatorname{Log}\alpha$ is a homomorphism of the multiplicative group F^* into the additive group $\mathbb{R}$. The kernel of that homomorphism is $\mathbb{Q}^* \cup \mathbb{Q}^*\sqrt{\Delta}$.*

(2) If η is a unit in $\mathcal{O}$, then $\operatorname{Log}\eta = -\log|\eta|$. In particular, we have $\operatorname{Log}\varepsilon = -R$.

(3) For any $\alpha \in F^*$ we have $\operatorname{Log}\alpha = -\operatorname{Log}\sigma(\alpha)$. □

In consequence, we may pull-back the natural topology on $\mathbb{R}/R\mathbb{Z}$ to $\mathcal{P}$ by using the following map.

Proposition 2.3. *The map*

$$d : \mathcal{P} \to \mathbb{R}/R\mathbb{Z}, \quad \alpha\mathcal{O} \mapsto \operatorname{Log}\alpha + R\mathbb{Z} \tag{1}$$

is a well defined homomorphism of the multiplicative group $\mathcal{P}$ into the additive group $\mathbb{R}/R\mathbb{Z}$.

Proof. We show that the map is well defined. Let $\alpha, \beta \in F^*$ with $\alpha\mathcal{O} = \beta\mathcal{O}$. Then α/β is a unit in $\mathcal{O}$. Hence α/β is a power of the fundamental unit ε of $\mathcal{O}$, and $\operatorname{Log}\alpha/\beta = \operatorname{Log}\alpha - \operatorname{Log}\beta$ is an integer multiple of the regulator R.

By Proposition 2.2 the map is a homomorphism. □

From Proposition 2.2 it is also clear that in the topology we introduced ideals in the same orbit under the natural operation of $\mathbb{Q}^*$ on $\mathcal{P}$ are inseparable, as are any pairs $\mathfrak{a}$ and $\sqrt{\Delta}\cdot\mathfrak{a}$. For the definition of the correct topological group that embeds into the circle group, see [Len82].

The topology can be extended to all of $\mathcal{I}$ by fixing for any non-principal equivalence classes $\mathcal{C}$ in $\mathcal{I}$ some ideal $\mathfrak{a} \in \mathcal{C}$, and using the map

$$d(\mathfrak{a}, \cdot) : \mathcal{C} \to \mathbb{R}/R\mathbb{Z} : \mathfrak{b} \mapsto d(\mathfrak{b}\mathfrak{a}^{-1}).$$

Definition 2.4. *Let $\mathfrak{a}$ and $\mathfrak{b}$ be equivalent $\mathcal{O}$-ideals. Then we call $d(\mathfrak{a}, \mathfrak{b})$ the* distance *from $\mathfrak{a}$ to $\mathfrak{b}$.*

3. Cycles of reduced $\mathcal{O}$-ideals

Any fractional $\mathcal{O}$-ideal $\mathfrak{a}$ can be written in standard representation as

$$\mathfrak{a} = q\left(\mathbb{Z}a + \mathbb{Z}\frac{b+\sqrt{\Delta}}{2}\right) \tag{2}$$

with a positive rational number q and integers (a, b) such that $\gcd(a, b, (b^2 - \Delta)/(4a)) = 1$, and

$$-a < b \leq a \quad \text{if} \quad a \geq \sqrt{\Delta} \ , \text{ or}$$

$$\sqrt{\Delta} - 2a < b < \sqrt{\Delta} \quad \text{if} \quad a < \sqrt{\Delta} \ .$$

A fractional ideal is called *reduced* if it does not contain a number α for which both $|\alpha|$ and $|\sigma(\alpha)|$ is smaller than the smallest positive number in $\mathfrak{a} \cap \mathbb{Z}$. In terms of the standard representation (2) this means that $q = 1$, and

$$|\sqrt{\Delta} - a| < b < \sqrt{\Delta}.$$

As can be seen from this condition, the set of all reduced $\mathcal{O}$-ideals $\mathcal{R}_\Delta$ is finite. We have an injective map

$$d(\mathcal{O}, \cdot) : \mathcal{R}_\Delta \cap \mathcal{P} \longrightarrow \mathbb{R}/R\mathbb{Z} : \mathfrak{a} \longmapsto d(\mathcal{O}, \mathfrak{a}).$$

The image of this map for the case that $\mathcal{O}$ has discriminant 1001 is depicted in Figure 1. Here, ideals in standard representation (2) are denoted by (a, b).

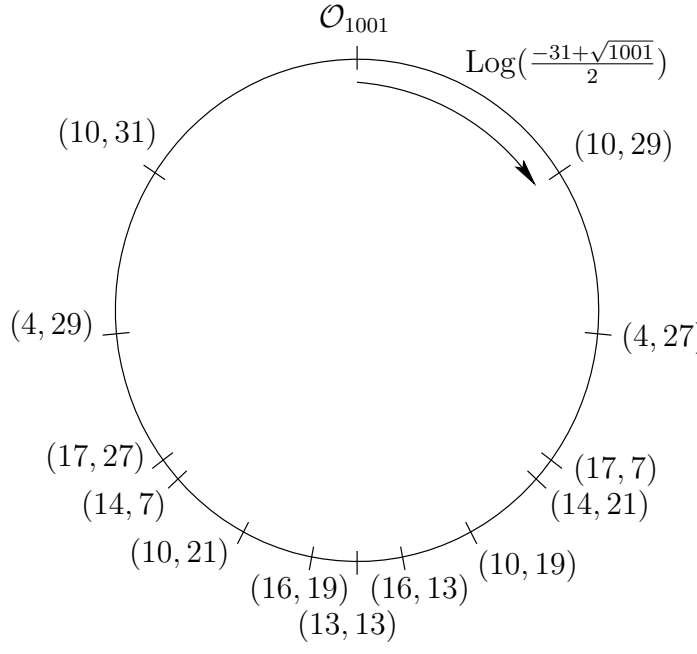

Figure 1. Embedding the principle cycle of $\mathcal{O}_{1001}$ into $\mathbb{R}/R\mathbb{Z}$

Definition 3.1. *For any fractional ideal* $\mathfrak{a}$ *we let* $\gamma(\mathfrak{a})$ *denote the number of minimal positive length in* F^* *such that*

$$\rho(\mathfrak{a}) = \gamma(\mathfrak{a})\mathfrak{a}$$

is reduced. The number $\gamma(\mathfrak{a})$ *is called the* reducing number *of* $\mathfrak{a}$.

For any equivalence class $\mathcal{C}$, the map ρ permutes $\mathcal{R}_\Delta \cap \mathcal{C}$. This set is called the *cycle of reduced ideals in* $\mathcal{C}$. For the principal class the cycle is called the *principal cycle*. Let $\mathfrak{a}_0 = \mathfrak{a}$ be reduced and set

$$\mathfrak{a}_{i+1} = \rho(\mathfrak{a}_i), \quad \mathfrak{a}_{-i-1} = \rho^{-1}(\mathfrak{a}_{-i}), \quad i \geq 0.$$

Moreover, we define the reducing numbers

$$\gamma_i = \gamma(\mathfrak{a}_i) = \frac{-2c_i}{b_i + \sqrt{\Delta}}, \quad i \in \mathbb{Z} \ .$$

There are easily proved upper and lower bounds on the length of γ_i and a lower bound for the product of two consecutive reducing numbers. For a proof, see [Len82].

Lemma 3.2. *(1)* $1/\sqrt{\Delta} < \operatorname{Log} \gamma_i < \frac{1}{2} \log \Delta$, $i \in \mathbb{Z}$.
(2) $\operatorname{Log} \gamma_i + \operatorname{Log} \gamma_{i+1} > \log 2$, $i \in \mathbb{Z}$. □

Corollary 3.3. *For any fractional ideal* $\mathfrak{a}$ *we have* $0 < \gamma(\mathfrak{a}) < \frac{1}{2} \log \Delta$. □

Lemma 3.4. *There is an algorithm that computes for any given fractional ideal* $\mathfrak{a}$ *both* $\rho(\mathfrak{a})$ *and* $\gamma(\mathfrak{a})$ *in quadratic time.*

Proof. In [BB99], it is shown that the classical reduction algorithm of Gauss can be executed in quadratic time. This yields reduced ideal $\rho_0(\mathfrak{a})$ and relative generator $\gamma_0(\mathfrak{a})$. In [Len82], Lenstra notes (albeit without explicit proof) that $\rho(\mathfrak{a})$ can be obtained from $\rho_0(\mathfrak{a})$ by no more than two reduction steps. □

4. Outline of the algorithm

Embedding the set of reduced principal ideals into the circle $\mathbb{R}/R\mathbb{Z}$ of circumference R is similar to embedding a cyclic group of order n into the circle $\mathbb{R}/n\mathbb{Z}$ of circumference n. (Cf. Figure 1.) While two neighboring group elements on the second circle have a fixed distance of 1, two reduced ideals have a distance of at least $1/\sqrt{\Delta}$ and at most $\frac{1}{2} \log \Delta$ (see Lemma 3.2). This indicates computing the regulator is similar to computing the order of a cyclic group. Also, computing the logarithm of a generator of a reduced ideal is like computing the discrete logarithm of a group element.

In this section we will use those analogies to develop an algorithm for computing the regulator R and for solving the equivalence problem.

The situation is the following. We are given reduced $\mathcal{O}$-ideals $\mathfrak{a}$ and $\mathfrak{b}$. The goal is to decide whether or not $\mathfrak{a}$ and $\mathfrak{b}$ are equivalent. Also, if $\mathfrak{a}$ and $\mathfrak{b}$ are equivalent, then we want to find $\lambda \in F$ with

$$\mathfrak{b} = \lambda\mathfrak{a}, \quad 0 < \operatorname{Log} \lambda \leq R.$$

In order to obtain the regulator, we will choose $\mathfrak{a} = \mathfrak{b} = \mathcal{O}$.

The algorithm uses two sequences of reduced $\mathcal{O}$-ideals. The first sequence $\mathfrak{a}_0, \mathfrak{a}_1, \ldots$ starts at

$$\mathfrak{a}_0 = \mathfrak{a}$$

and is defined by

$$\mathfrak{a}_{i+1} = \rho(\mathfrak{a}_i), \quad i \geq 0.$$

It is called the *baby-step sequence* because the distance between two consecutive elements of the sequence is very small, cf. Lemma 3.2. Set

$$\alpha_i = \prod_{j=0}^{i-1} \gamma_i, \quad i \geq 0$$

with γ_i from Section 3. Then

$$\mathfrak{a}_i = \alpha_i \mathfrak{a}, \quad i \geq 0.$$

Also, it follows from Lemma 3.2 that

$$\operatorname{Log} \alpha_{i+1} > \operatorname{Log} \alpha_i > 0 \quad i \geq 0,$$

$$\operatorname{Log} \alpha_{i+2} \geq \operatorname{Log} \alpha_i + \log 2, \quad i \geq 0, \tag{3}$$

and

$$\lim_{i \to \infty} \operatorname{Log} \alpha_i = \infty. \tag{4}$$

Initially, the baby-step sequence is calculated until $L \geq 0$ is found with

$$\operatorname{Log} \alpha_{2(L+1)} \geq \frac{1}{2} \log \Delta. \tag{5}$$

Define

$$s_i = \operatorname{Log} \alpha_{2(L+i)} \quad i \geq 1. \tag{6}$$

The second sequence $\mathfrak{b}_0, \mathfrak{b}_1, \ldots$ starts at

$$\mathfrak{b}_0 = \mathfrak{b}.$$

It is called the *giant-step sequence* since the distances between the consecutive elements of that sequence become larger and larger. More precisely, the algorithm determines positive numbers $\delta_i \in F$ such that

$$\mathfrak{b}_i = (\delta_i / \alpha_{2(L+i)}) \mathfrak{b}_{i-1}, \quad i \geq 1$$

is reduced. If

$$\beta_i = \delta_i / \alpha_{2(L+i)}, \quad i \geq 1$$

then we require

$$s_i - \frac{1}{2}\log \Delta < -\operatorname{Log} \beta_i \le s_i, \quad i \ge 1. \tag{7}$$

The algorithm is based on the following proposition.

Proposition 4.1. *Assume that* $\mathfrak{a}$ *and* $\mathfrak{b}$ *are equivalent. Then the following are true.*

(1) There are e, f *such that* $e \ge 0$ *and*

$$\mathfrak{b}_e = \mathfrak{a}_f, \quad f \in \{1, \dots, 2(e + L + 1)\}. \tag{8}$$

(2) If (e, f) *is the lexicographically smallest pair that satisfies* (8)*, then for*

$$\lambda = \alpha_f / \prod_{i=1}^{e} \beta_i \tag{9}$$

we have $\mathfrak{b} = \lambda\mathfrak{a}$ *and*

$$0 < \operatorname{Log} \lambda \le R. \tag{10}$$

Also,

$$e < \lceil E_\Delta \rceil \tag{11}$$

where

$$E_\Delta = \log_2 \Delta + \sqrt{2R/\log 2}.$$

(3) If $\mathfrak{a} = \mathfrak{b} = \mathcal{O}$ *and* e, f*, and* λ *are as in 2., then* $\operatorname{Log} \lambda = R$ *and the fundamental unit of* $\mathcal{O}$ *is* $1/\lambda$*.*

For $e = 1, 2, \dots$ the algorithm calculates the baby-step sequence $(\mathfrak{a}_i)_{1 \le i \le 2(e+L+1)}$ and the giant-step $\mathfrak{b}_e$. If a *match* (8) is found, then the equivalence of $\mathfrak{a}$ and $\mathfrak{b}$ is proved and with λ from (9) we have $\mathfrak{b} = \lambda\mathfrak{a}$ and $0 < \operatorname{Log} \lambda \le R$.

Note that

$$\lambda = \prod_{i=0}^{f-1} \gamma_i \cdot \prod_{i=0}^{2L+1} \gamma_i^e \cdot \prod_{i=1}^{e-1} (\gamma_{2(L+i)}\gamma_{2(L+i)+1})^{e-i} \cdot \prod_{i=1}^{e} \delta_i^{-1}. \tag{12}$$

Since $\gamma_i = \gamma(\mathfrak{a}_i)$ can be determined easily from $\mathfrak{a}_i$, it suffices to store $\mathfrak{a}_i$, $1 \le i \le \max\{f, 2(e + L)\}$, and δ_i, $1 \le i \le e$ in order to compute the power product representation of λ in (12). This will actually be the representation of λ output by our algorithm.

If for no $e < \lceil E_\Delta \rceil$ a match (8) is found, then it is established that $\mathfrak{a}$ and $\mathfrak{b}$ are not equivalent. It follows from (11) that e and f are of the order of magnitude $\sqrt{2R} = \Delta^{1/4+o(1)}$. We will see that the running time of the algorithm is of the same order of magnitude.

We prove Proposition 4.1. Choose positive $\lambda \in F$ with

$$\mathfrak{b} = \lambda \mathfrak{a}$$

and

$$0 < \operatorname{Log} \lambda \leq R. \tag{13}$$

Define

$$\lambda_k = \lambda \prod_{i=1}^{k} \beta_i, \quad k \geq 0.$$

Then

$$\mathfrak{b}_k = \lambda_k \mathfrak{a}, \quad k \geq 0.$$

To show 1. we prove the following result.

Lemma 4.2. *There is some $k \geq 0$ with $\operatorname{Log} \lambda_k \leq s_{k+1}$. Also, if e is the smallest such k, then $\operatorname{Log} \lambda_e > 0$.*

Proof. It follows from (7), the definition of s_i in (6), and (4) that

$$\lim_{i \to \infty} \operatorname{Log} \beta_i = -\infty.$$

This proves the existence of k. Let e be the smallest such k. Assume that $\operatorname{Log} \lambda_e \leq 0$. Then $\operatorname{Log} \lambda_{e-1} = \operatorname{Log} \lambda_e - \operatorname{Log} \beta_e \leq s_e$ by (7). This contradicts the minimality of e. □

Proof of 1. Let e be as in Lemma 4.2. Then Lemma 4.2 and (13) imply $0 < \operatorname{Log} \lambda_e \leq R$.

Let f be the smallest positive index such that $\mathfrak{b}_e = \mathfrak{a}_f$. Note that f exists since $\mathfrak{b}_e$ is reduced and sits on the cycle $\{\mathfrak{a}_i\}$ of reduced ideals in the class of $\mathfrak{a}$. Then $0 < \operatorname{Log} \alpha_f \leq R$. It follows that $\operatorname{Log} \alpha_f = \operatorname{Log} \lambda_e \leq \operatorname{Log} \alpha_{2(e+L+1)}$. Hence we also have $0 < f \leq 2(e + L + 1)$.

This concludes the proof of 1.

Proof of 2. Let (e, f) be the lexicographically smallest pair that satisfies (8). Let e' be the value of e in Lemma 4.2. Then the proof of 1. shows that

$$e \leq e' \tag{14}$$

and we have

$$0 < \operatorname{Log} \alpha_f = \operatorname{Log} \lambda_e + kR \tag{15}$$

for some integer k. Now $\operatorname{Log} \lambda_e = \operatorname{Log} \lambda + \sum_{i=1}^{e} \operatorname{Log} \beta_i \leq R$ since $\operatorname{Log} \lambda \leq R$ and $\operatorname{Log} \beta_i < 0$ by (7) and (5). Hence, (15) implies $k \geq 0$ and $\operatorname{Log} \alpha_f \geq \operatorname{Log} \lambda_e$. Since $f \leq 2(e+L+1)$, it follows that $s_{e+1} = \operatorname{Log} \alpha_{2(e+L+1)} \geq \operatorname{Log} \lambda_e$. This implies $e \geq e'$. Together with (14) we have $e = e'$ and $\operatorname{Log} \lambda_e > 0$ by Lemma 4.2. The minimality of f implies $k = 0$. Hence $\operatorname{Log} \lambda = \operatorname{Log} \alpha_f - \sum_{i=1}^{e} \operatorname{Log} \beta_i$. Indeed, Proposition 2.2 implies $\lambda = \alpha_f / \prod_{i=1}^{e} \beta_i$ so that λ

chosen at the beginning of the proof coincides with the one defined in the Proposition and (10) holds.

We prove the upper bound on e. Set $E = \lceil E_\Delta \rceil$. Then (7) and Lemma 3.2 imply

$$\begin{aligned}\log \lambda_E &= \operatorname{Log} \lambda + \sum_{i=1}^{E} \operatorname{Log} \beta_i \\ &\le \operatorname{Log} \lambda - \sum_{i=1}^{E} (\operatorname{Log} \alpha_{2(L+i)} - \frac{1}{2} \log \Delta) \\ &\le \operatorname{Log} \lambda + (E/2) \log \Delta - \frac{E(E+1)}{2} \log 2.\end{aligned}$$

Now we have $E(E+1)/2 \cdot \log 2 > E^2/2 \cdot \log 2 > E/2 \cdot \log \Delta + R$. Hence, $\lambda_E < 0$. Since $\operatorname{Log} \lambda_e > 0$ by Lemma 4.2 it follows that $e < E$.

Proof of 3. Let $\mathfrak{a} = \mathfrak{b} = \mathcal{O}$. Let λ be the number defined in 2. Then $\mathcal{O} = \lambda \mathcal{O}$. So λ is a unit in $\mathcal{O}$. Also $0 < \operatorname{Log} \lambda \le R$. Finally, all factors of λ given in (12) are positive, hence so is λ. Proposition 2.2 then says that $1/\lambda$ is the fundamental unit, as desired.

5. Construction of the the giant-steps

We explain how a giant-step is computed. We let $\mathfrak{b}_0 = \mathfrak{b}$. For some $e \ge 0$ we are given the giant-step ideal $\mathfrak{b}_e$ and the baby-step ideal $\mathfrak{a}_{2(L+e+1)}$. We compute

$$\mathfrak{b}_{e+1} \leftarrow \rho(\mathfrak{b}_e \mathfrak{a}_{2(L+e+1)}^{-1}), \quad \delta_{e+1} \leftarrow \gamma(\mathfrak{b}_e \mathfrak{a}_{2(L+e+1)}^{-1}).$$

Note that this computation only involves one ideal multiplication and one reduction of an ideal with norm bounded by Δ. Then

$$\beta_{e+1} = \delta_{e+1} / \alpha_{2(L+e+1)}.$$

That number is not stored explicitely, as this is too space consuming. It suffices to store δ_{e+1}.

Lemma 5.1. *We have* $s_{e+1} - \frac{1}{2} \log \delta < -\operatorname{Log} \beta_{e+1} \le s_{e+1}$.

Proof. By construction we have $-\operatorname{Log} \beta_{e+1} = s_{e+1} - \operatorname{Log} \delta_{e+1}$. Since $-\frac{1}{2} \log \Delta < -\operatorname{Log} \delta_{e+1} \le 0$ by Corollary 3.3, we have $s_{e+1} - \frac{1}{2} \log \Delta < \operatorname{Log} \beta_{e+1} \le s_{e+1}$ as asserted. □

6. The complete algorithm

We now present the algorithms for computing the fundamental unit of $\mathcal{O}$ and for deciding equivalence between two $\mathcal{O}$-ideals. For the first, called `TerrUnit`, see Algorithm 1 on the following page. The second, called `Terr-Equivalent`, is obtained from the first by (1) initializing $\mathfrak{a}_0$ and $\mathfrak{b}_0$ with the

Algorithm 1 `TerrUnit`$(\mathcal{O})$

Input: The order $\mathcal{O}$
Output: $\mathfrak{a}_1, \ldots, \mathfrak{a}_{2(e+L)}$, $\delta_1, \ldots, \delta_e$, f such that λ from (12) is the fundamental unit of $\mathcal{O}$

Initial Baby-steps $\mathfrak{a}_0 \leftarrow \mathcal{O}$, $L \leftarrow -1$
repeat
  $L \leftarrow L+1$, $\mathfrak{a}_{2L+1} \leftarrow \rho(\mathfrak{a}_{2L})$, $\mathfrak{a}_{2L+2} \leftarrow \rho(\mathfrak{a}_{2L+1})$
until $\operatorname{Log} \alpha_{2L+2} > \frac{1}{2} \log \Delta$
if $\mathcal{O} = \mathfrak{a}_f$ for some $1 < f \leq 2(L+1)$ **then**
  return $\mathfrak{a}_1, \ldots, \mathfrak{a}_f, f$
$e \leftarrow 0$, $\mathfrak{b}_0 \leftarrow \mathcal{O}$
loop
  Giant Step
  $\mathfrak{c} \leftarrow \mathfrak{b}_e \mathfrak{a}_{2(L+e+1)}^{-1}$, $(\mathfrak{b}_{e+1}, \delta_{e+1}) \leftarrow (\rho(\mathfrak{c}), \gamma(\mathfrak{c}))$, $e \leftarrow e+1$
  Baby Steps
  $\mathfrak{a}_{2L+2e+1} \leftarrow \rho(\mathfrak{a}_{2L+2e})$, $\mathfrak{a}_{2L+2e+2} \leftarrow \rho(\mathfrak{a}_{2L+2e})$
  Table look-up
  if $\mathfrak{b}_e = \mathfrak{a}_f$ for some $f \leq 2(L+e+1)$ **then**
    return $\mathfrak{a}_1, \ldots, \mathfrak{a}_{2(e+L+1)}, \delta_1, \ldots, \delta_e, f$

given ideals, and (2) terminating the loop once $e > E_\Delta$. In those algorithms the baby-step ideals $\mathfrak{a}_i$, $i \geq 1$, are stored in a hash table. Then deciding whether a given reduced $\mathcal{O}$-ideal is in that table takes time $\mathrm{O}(\log \Delta)$.

Note that `TerrEquivalent` presumes that an approximation to the regulator has been computed in advance. This is only used to obtain an integer close to and larger than E_Δ. (The rounding in the calculation of E need not be exact.) The pre-computation of R can be avoided by computing two giant step sequences one beginning at $\mathfrak{b}$ and one at $\mathfrak{a}$ and terminating with result `nil` when there is a match of the second one with the baby step table.

Example 6.1. Table 6.1 lists the ideals computed in the course of the execution of `TerrUnit` for $\Delta = 2521$. Ideals in standard representation $a\mathbb{Z} + \mathbb{Z}(b + \sqrt{\Delta})/2$ are listed as (a, b). Distances given are distances to the unit ideal $\mathcal{O}$. We set $\omega = (1+\sqrt{\Delta})/2$. Finally, note that $1/2 \cdot \log \Delta \approx 3.916$.

The table shows that $\mathfrak{a}_{11} = \mathfrak{b}_6$ (connected by an arrow). Using (12) the data in the table yields the fundamental unit. We obtain $R \approx 85.768$.

L	e	i	$\mathfrak{a}_i$	distance	$\mathfrak{b}_e$	distance	δ_e
0		1	(30, 11)	2.203			
		2	(20, 29)	2.426			
1		3	(21, 13)	3.085			
		4	(28, 43)	3.350			
2		5	(6, 41)	4.630			
		6	(35, 29)	5.776			
	1	7	(12, 43)	6.435	(35,41)	−5.776	35
		8	(14, 41)	7.715			
	2	9	(15, 49)	8.861	(10,31)	−13.491	2
		10	(2, 47)	11.065			
	3	11	(39, 31)	12.770	(5,41)	−24.555	1
		12	(10, 49)	13.491			
	4	13	(3, 47)	15.694	(12,37)	−37.833	$1-\omega/5$
		14	(26, 5)	17.400			
	5	15	(24, 43)	17.500	(9,35)	−54.379	$6-\omega/6$
		16	(7, 41)	18.779			
	6				(39,31)	−72.998	$(-7+2\omega)/9$

TABLE 1. Baby-step ideals $\mathfrak{a}_i$ and giant-step ideals $\mathfrak{b}_e$ computed by `TerrUnit` for $\Delta = 2521$

7. Analysis of the algorithm

Proposition 7.1. *Algorithms `TerrUnit` and `TerrEquivalent` require time and space* $\mathrm{O}((\log\Delta+\sqrt{R})(\log\Delta)^2)$.

Proof. It follows from (3) that $L = \mathrm{O}(\log\Delta)$. Also, it follows from Proposition 4.1 that Algorithms `TerrUnit` and `TerrEquivalent` both terminate with the correct output and $e = \mathrm{O}(\log\Delta+\sqrt{R})$.

In each iteration of the precomputation, the algorithms apply the reduction operator twice to reduced $\mathcal{O}$-ideals. That application has running time $\mathrm{O}((\log\Delta)^2)$. In each iteration of the main loop, both algorithms apply ρ once to the quotient of two reduced $\mathcal{O}$-ideals and at most twice to two reduced $\mathcal{O}$-ideals. By Lemma 3.4 each application of ρ takes time $\mathrm{O}((\log\Delta)^2)$. This proves the running time estimate.

Both algorithms store $\mathrm{O}(\log\Delta+\sqrt{R})$ reduced $\mathcal{O}$-ideals and numbers δ_i. The size of a reduced $\mathcal{O}$-ideal is $\mathrm{O}(\log\Delta)$, and by Lemma 3.4 each δ_i requires space $\mathrm{O}((\log\Delta)^2)$. This implies the space estimate. □

8. Computing discrete logarithms and the class group

In this section, we briefly touch upon the problems of computing discrete logarithms in the class group of a real-quadratic field, and of computing the class group itself.

Extended Discrete Logarithm Problem. Given two ideals $\mathfrak{a}$ and $\mathfrak{b}$, find $n \in \mathbb{Z}$ and $\alpha \in F$ such that

$$\mathfrak{a} = \alpha \cdot \mathfrak{b}^n.$$

This problem is most conveniently solved by combining Terr's original DL algorithm with the classical baby-step giant-step algorithm in the infrastructure.

More precisely, we let $\mathfrak{a}_k, \mathfrak{b}_k$, and $\mathfrak{c}_k$ be defined recursively as follows:

$$\begin{aligned} \mathfrak{a}_0 = \mathcal{O}, \qquad \mathfrak{a}_k &= \gamma_{a,k} \cdot \mathfrak{a}_{k-1}/\mathfrak{b} &&= \rho(\mathfrak{a}_{k-1}/\mathfrak{b}), \\ \mathfrak{b}_k &= \gamma_{b,k} \cdot \mathfrak{c}_k/\mathfrak{a} &&= \rho(\mathfrak{c}_k/\mathfrak{a}), \\ \mathfrak{c}_0 = \mathcal{O}, \qquad \mathfrak{c}_k &= \gamma_{c,k} \cdot \mathfrak{c}_{k-1}/\mathfrak{a}_k &&= \rho(\mathfrak{c}_{k-1}/\mathfrak{a}_k). \end{aligned}$$

Obviously, the ideals $\mathfrak{a}_k$, $\mathfrak{b}_k$ are reduced, and $\mathfrak{a}_k \sim \mathfrak{b}^{-k}$, and $\mathfrak{b}_k \sim \mathfrak{b}^{\frac{k(k+1)}{2}}/\mathfrak{a}$. More precisely,

$$\begin{aligned} \mathfrak{a}_k &= \prod_{i=1}^{k} \gamma_{a,i} \ \cdot \ \mathfrak{b}^{-k}, \\ \mathfrak{b}_k &= \gamma_{b,k} \cdot \prod_{i=1}^{k} \gamma_{c,i}\gamma_{a,i}^{-k+i-1} \ \cdot \ \mathfrak{b}^{k(k+1)/2}/\mathfrak{a}, \\ \mathfrak{c}_k &= \prod_{i=1}^{k} \gamma_{c,i}\gamma_{a,i}^{-k+i-1} \ \cdot \ \mathfrak{b}^{k(k+1)/2}. \end{aligned}$$

We compute baby-step sequences starting at $\mathfrak{a}_k$, and giant-step sequences starting at $\mathfrak{b}_k$ and $\mathfrak{c}_k$. All baby-step sequences have the same number of elements (or same approximate length) which is derived from the output of a previous run of `TerrUnit`. Regarding the giant-steps, care has to be taken that they are shorter than the shortest sequence of baby-steps computed before.

Once the algorithm finds a match, i.e. the current giant-step ideal occurs in the baby-step table, it has found indices $k \leq l$ such that $\mathfrak{a}_k \sim \mathfrak{b}_l$, or $\mathfrak{a}_k \sim \mathfrak{c}_l$, respectively. In the first case, $n = l(l+1)/2 + k$ is the sought exponent. In the second case n equals the order of $[\mathfrak{b}]$, and $[\mathfrak{a}] \notin \langle [\mathfrak{b}] \rangle$. Once the exponent is found, it is easy to compute the generator of $\mathfrak{a}$ relative to $\mathfrak{b}^n$ on the basis of the data computed so far.

The complexity of the algorithm is bounded by $O(\sqrt{n}(\log \Delta + \sqrt{R}) \cdot (\log \Delta)^{2+\epsilon})$. It can be re-arranged in such a manner that no approximations

of real numbers need to be computed and the ϵ can be dropped from the exponent. Details can be found in [Vol03].

In [BS05], Buchmann and Schmidt have explained how to compute the structure of a finite Abelian group from generators and relations. That algorithm cannot be immediately used in the context of real quadratic class groups since deciding equality of real quadratic ideal classes is more difficult. Moreover, the algorithm presumes that we have a generating set for the group available. Finding such a generating set for the class group of a number field is difficult unless one assumes an Extended Riemann Hypothesis. If one does assume this hypothesis, then one can use the Bach bounds [Bac90] to obtain a generating set of quadratic size. Using this generating set and adapting Buchmann and Schmidt's algorithm the same way we adapted Terr's Discrete Logarithm algorithm we can easily construct a deterministic class group algorithm for real quadratic orders that runs in time $\Delta^{1/4+o(1)}$.

References

[Bac90] Eric Bach, *Explicit bounds for primality testing and related problems.* Mathematics of Computation **55** (1990), no. 191, 355–380.

[BB99] Ingrid Biehl, Johannes Buchmann, *An analysis of the reduction algorithms for binary quadratic forms.* In Peter Engel and Halyna M. Syta, editors, *Voronoi's Impact on Modern Science, Kyiv, Ukraine 1998*, pages 71–98. National Academy of Sciences of Ukraine, 1999.

[BJT97] Johannes Buchmann, Michael J. Jacobson, Jr., Edlyn Teske, *On some computational problems in finite abelian groups.* Math. Comp. **66** (1997), no. 220, 1663–1687.

[BS05] Johannes Buchmann, Arthur Schmidt, *Computing the structure of a finite abelian group.* Mathematics of Computation **74** (2005), no. 252, 2017–2026.

[BV07] Johannes Buchmann, Ulrich Vollmer *Binary Quadratic Forms – An Algorithmic Approach.* Algorithms and Computation in Mathematics, vol. **20**. Springer 2007.

[BW88] Johannes Buchmann, Hugh C. Williams, *On the infrastructure of the principal ideal class of an algebraic number field of unit rank one.* Math. Comp. **50** (1988), no. 182, 569–579.

[Len82] Hendrik W. Lenstra, Jr., *On the calculation of regulators and class numbers of quadratic fields.* In J. V. Armitage, editor, *Journees Arithmetiques, Exeter 1980*, London Mathematical Society Lecture Notes Series, vol. **56**, pages 123–150. Cambridge University Press, 1982.

[Sch04] René Schoof, *Computing Arakelov class groups.* `http://axp.mat.uniroma2.it/~schoof/infranew2.pdf`, October 2004.

[Sha71] Daniel Shanks, Class number, a theory of factorization, and genera. In *1969 Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Brook, N.Y., 1969)*, pages 415–440. Amer. Math. Soc., Providence, R.I., 1971.

[Ter00] David C. Terr *A modification of Shanks' baby-step giant-step algorithm.* Mathematics of Computation **69** (2000), no. 230, 767–773.

[Vol03] Ulrich Vollmer, *Rigorously Analyzed Algorithms for the Discrete Logarithm Problem in Quadratic Number Fields.* PhD thesis, Technische Universität Darmstadt, Fachbereich Informatik, 2003.

Johannes Buchmann and Ulrich Volmer
Technische Universität Darmstadt
Department of Computer Science
Hochschulstr. 10, 64289 Darmstadt, Germany
E-mail: buchmann@cdc.informatik.tu-darmstadt.de
E-mail: uvollmer@cdc.informatik.tu-darmstadt.de