FAST MULTIQUADRATIC S-UNIT COMPUTATION AND APPLICATION TO THE CALCULATION OF CLASS GROUPS

JEAN-FRANÇOIS BIASSE AND CHRISTINE VAN VREDENDAAL

Abstract. Let \( L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n}) \) be a real multiquadratic field and \( S \) be a set of prime ideals of \( L \) that does not contain any divisors of 2. In this paper, we present a heuristic algorithm for the computation of the \( S \)-class group and the \( S \)-unit group that runs in time \( \text{Poly}(\log(\Delta), \text{Size}(S)) e^{\tilde{O}(\sqrt{\ln d})} \) where \( d = \max_{1 \leq i \leq n} d_i \) and \( \Delta \) is the discriminant of \( L \). We use this method to compute the ideal class group of the maximal order \( \mathcal{O}_L \) of \( L \) in time \( \text{Poly}(\log(\Delta)) e^{\tilde{O}(\sqrt{\log d})} \).

1. Introduction

Let \( L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n}) \) be a real multiquadratic number field, and \( S \) be a set of prime ideals of \( L \). The \( S \)-unit group \( U_S \) of \( L \) is the set of elements \( \alpha \in L \) such that there is \( \vec{e} \in \mathbb{Z}^{|S|} \) satisfying \( \alpha \mathcal{O}_L = \prod_{p \in S} p^{e_p} \) where \( \mathcal{O}_L \) is the maximal order of \( L \). The computation of the \( S \)-unit group is a fundamental problem in computational number theory with many applications.

In this paper, we present an original algorithm for the computation of certain \( S \)-unit groups in real multiquadratic fields. The main motivation for the development of this algorithm is the computation of the ideal class group of \( \mathcal{O}_L \). The computation of \( \text{Cl}(\mathcal{O}_L) \) can be trivially deduced from the knowledge of an \( S \)-unit group where the classes of the elements of \( S \) generate \( \text{Cl}(\mathcal{O}_L) \). The computation of the ideal class group is one of the four major tasks in computational number theory postulated by Zassenhaus [23, p. 2] (together with the computation of the unit group, the Galois group and the ring of integers). In 1968, Shanks [25, 26] proposed an algorithm relying on the baby-step giant-step method to compute the class number and the regulator of a quadratic number field in time \( O \left( |\Delta|^{1/4+\epsilon} \right) \), or \( O \left( |\Delta|^{1/5+\epsilon} \right) \) under the extended Riemann hypothesis [22]. Then, a subexponential strategy for the computation of the group structure of the class group of an imaginary quadratic field was described in 1989 by Hafner and McCurley [21]. The expected running time of this method is

\[
L_\Delta(1/2, \sqrt{2} + o(1)) = e^{(\sqrt{2}+o(1))\sqrt{\ln |\Delta| \ln \ln |\Delta|}}.
\]

Buchmann [15] generalized this result to the case of infinite classes of number fields with fixed degree. Practical improvements to Buchmann’s algorithm were presented in [19] by Cohen, Diaz Y Diaz and Olivier. Biasse [5] described an algorithm for computing the ideal class group and the unit group of \( \mathcal{O} = \mathbb{Z}[\theta] \) in heuristic complexity bounded by \( L_\Delta(1/3, c) \) for some \( c > 0 \) valid in certain classes of number fields.

This work was supported by NIST under grant 60NANB17D184 and by the Simons Foundation under grant 430128.
fields. In [6, 9], Biasse and Fieker showed that there was a heuristic subexponential algorithm for the computation of the ideal class group in all classes of number fields. The methods of [9] can be specialized to the case of cyclotomic fields for a better asymptotic complexity [7]. The computation of the ideal class group is also the subject of study in the context of quantum computing. It was recently proved (under the GRH) by Biasse and Song that there is a quantum polynomial time algorithm for the computation of the ideal class group of an arbitrary field [13]. The most efficient practical implementations of algorithms for the computation of the ideal class group are either based on the quadratic sieve [12, 4, 11, 10] for quadratic fields and on the number field sieve [8] for number fields of higher degree.

The computation of $S$-units is also instrumental in the resolution of norm equations [27]. Indeed, it is the bottleneck of the resolution in $x$ of $N_{L/K}(x) = a$ for a given $a \in K$ where $L/K$ is a Galois extension. This computational problem is closely related to Hilbert’s 10th problem, for which there is no efficient general solution.

Contributions. Let $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ be a real multiquadratic number field, and $S$ be a set of prime ideals of $L$ that does not contain any ideals above 2. We define $d = \max_{i \leq n} d_i$ and $\Delta = \text{disc}(L)$.

- We describe an algorithm for the computation of $\text{Cl}(\mathcal{O}_L)$ in heuristic complexity $\text{Poly}(\log(\Delta))\tilde{e}O(\sqrt[5]{\log(\Delta)})$.
- We describe a heuristic algorithm for the computation of the $S$-class group and the $S$-unit group of $L$ in time $\text{Poly}(\log(\Delta), \text{Size}(S))\tilde{e}O(\sqrt[5]{\log(\Delta)})$.
- We report on the performance of an implementation of our algorithms.

Our recursive approach is based on the unit group computation of [3] which we extended to the more general problem of the computation of the $S$-unit group. In the case where $d$ is small compared to $\Delta$, our method for computing class groups, $S$-class groups and $S$-unit groups runs in heuristic polynomial time in $\log(\Delta)$ (and in the size of $S$) where $\log(x)$ is the bit size of the integer $x$. This is ensured when $\log(d) \leq \log(\log(\Delta))^c$ for some constant $c < 2$. For example, this is the case when the $d_i$ are the first $n$ consecutive primes. This is the first non-quantum algorithm that runs in polynomial time on infinite classes of number fields. The main ingredient of our recursion strategy is not restricted to multiquadratic fields. We can take advantage of computations in subfields whenever there are two different $\sigma, \tau \in \text{Gal}(L/\mathbb{Q})$ of order two. General subfields might not enjoy the same general recursive structure as multiquadratic fields, but we expect that the reduction to the computation in subfields will improve the performance of class group algorithms.

The application of these methods to more general fields was left for future work.

2. Preliminaries

2.1. Number fields. A number field $K$ is a finite extension of $\mathbb{Q}$. Its ring of integers $\mathcal{O}_K$ has the structure of a lattice of degree $n = [K : \mathbb{Q}]$. A number field has $r_1 \leq n$ real embeddings $(\sigma_i)_{1 \leq i \leq r_1}$ and $2r_2$ complex embeddings $(\sigma_i)_{r_1 < i \leq 2r_2}$ (coming as pairs of conjugates). The pair $(r_1, r_2)$ is the signature of $K$. The field $K$ is isomorphic to $\mathcal{O}_K \otimes \mathbb{Q}$. The norm of an element $x \in K$ is defined by $\mathcal{N}(x) = \prod \sigma_i(x)$. Let $(\alpha_1, \ldots, \alpha_n)$ such that $\mathcal{O}_K = \mathbb{Z}[\alpha_i]$, then the discriminant of $K$ is $\Delta(K) := \det^2(T_2(\alpha_i, \alpha_j))$, where $T_2$ is defined by $T_2(x, x') := \sum_i \sigma_i(x)\sigma_i(x')$. When there is no ambiguity, we simply denote it by $\Delta$. 
2.2. Units of $\mathcal{O}_K$. Elements $u \in \mathcal{O}_K$ that are invertible in $\mathcal{O}_K$ are called units. Equivalently, they are the elements $u \in K$ such that $(u) := (u)\mathcal{O}_K = \mathcal{O}_K$. The unit group of $\mathcal{O}_K$ where $K$ is a real multiquadratic field has rank $r = n - 1$ and has the form $\mathcal{O}_K^* = \mu \times \langle \epsilon_1 \rangle \times \cdots \times \langle \epsilon_r \rangle$ where $\mu$ are roots of unity (torsion units) and the $\epsilon_i$ are non-torsion units. Such $(\epsilon_i)_{i \leq r}$ are called a system of fundamental units of $\mathcal{O}_K$. Units generate a lattice $\mathcal{L}$ of rank $r$ in $\mathbb{R}^{n+1}$ via the embedding $x \in K \mapsto \log(x) := (\log(|\sigma_1(x)|), \ldots, \log(|\sigma_{r+1}(x)|))$. The volume $R$ of $\mathcal{L}$ is an invariant of $K$ called the regulator. The regulator $R$ and the class number $h$ satisfy $hR = \frac{|\mu\sqrt{\Delta}|}{2\pi (2\pi)^{\frac{r}{2}}} \lim_{s \to 1} ((s-1)\zeta_K(s))$, where $\zeta_K(s) = \sum_{a \in \mathcal{O}_K^*} \frac{1}{N(a)^s}$ is the usual $\zeta$-function associated to $K$ and $|\mu|$ is the cardinality of $\mu$ the group of torsion units. This allows us to derive a bound $h^*$ in polynomial time under GRH that satisfies $h^* \leq hR < 2h^*$ ([2]).

2.3. Multiquadratic fields. In this paper, we focus on towers of quadratic extensions.

**Definition 2.1.** Let $d_1, \ldots, d_n$ be squarefree integers that are multiplicatively independent modulo squares (i.e., they are independent in $\mathbb{Q}^*/((\mathbb{Q}^*)^2)$. Then $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ is called a multiquadratic field and $N := [L : \mathbb{Q}] = 2^n$. Its Galois group $\text{Gal}(L/\mathbb{Q}) := \{\text{Automorphisms of } L \text{ that fix } \mathbb{Q}\}$ is isomorphic to $(\mathbb{Z}/2\mathbb{Z})^n$.

When $n = 1$, the field $L = \mathbb{Q}(\sqrt{d_1})$ is simply called a quadratic field. In this paper, we focus on real multiquadratic fields, that is, those that satisfy $\forall i \leq n, d_i > 0$. The discriminant of a real multiquadratic field is given to us by an explicit formula. This is useful for the computation of its maximal order.

**Lemma 2.2.** Let $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ a multiquadratic field as given above and $\prod_{i=1}^{r_m} p_i^{m_i}$ with $p_1 < p_2 < \ldots < p_s$ be the factorization of $\prod_{i=1}^{n} d_i$. Then $\Delta(L) = (2^n p_1 \cdot p_2 \cdots p_s)^{2n-1}$ where

$$a = \begin{cases} 0, & d_i \equiv 1 \text{ mod } 4 \ (\forall \ 1 \leq i \leq n) \\ 2, & p_1 = 2 \text{ and } p_i \equiv 1 \text{ mod } 4 \ (\forall \ 2 \leq i \leq n) \text{ or} \\ p_1 \neq 2 \text{ and } \exists \ i \text{ s.t. } p_i \equiv 3 \text{ mod } 4 \\ 3, & \text{otherwise} \end{cases}$$

**Proof.** This follows from Theorem 2.1 of [24].

If we take $d_1, d_2, \ldots, d_n$ to be the first $n$ primes, then their product is the primorial $p_n# = e^{(1+o(1))n \log n}$. Combining this with Lemma 2.2 gives $\ln \Delta(L) = \frac{1}{2} N n \log n = \frac{1}{4} N \log N \log \log N$.

2.4. Class groups. Elements of the form $\mathfrak{a}$ where $\mathfrak{a} \subseteq \mathcal{O}_K$ is an ideal of the ring of integers of $K$ and $d > 0$ are called fractional ideals. Ideals of $\mathcal{O}_K$ are also referred to as integral ideals. Fractional ideals have the structure of a $\mathbb{Z}$-lattice of degree $n = [K : \mathbb{Q}]$, and they form a multiplicative group $\mathcal{I}$. Elements of $\mathcal{I}$ admit a unique decomposition as a product of non-zero prime ideals of $\mathcal{O}_K$ (with possibly negative exponents). The norm of integral ideals is given by $N(\mathfrak{a}) := [\mathcal{O}_K : \mathfrak{a}]$, which extends to fractional ideals by $N(\mathfrak{a}/\mathfrak{b}) := N(\mathfrak{a})/N(\mathfrak{b})$. The norm of a principal (fractional) ideal agrees with the norm of its generator $N(x_1 \mathcal{O}_K) = |N(x)|$. The principal fractional ideals $\mathcal{P}$ of $K$ are a subgroup of $\mathcal{I}$ and the ideal class group of $\mathcal{O}_K$ is defined by $\text{Cl}(\mathcal{O}_K) := \mathcal{I}/\mathcal{P}$. We denote by $[\mathfrak{a}]$ the class of a fractional $\mathfrak{a}$ in
The best asymptotic algorithms to compute the ideal class group of \( \mathcal{O}_K \) follow the general framework derived from the algorithm of Hafner and McCurley [21] (subsequently generalized by Buchmann [15] and Biasse-Fieker [9]). Let \( B > 0 \) be a bound and define a factor base as \( \mathcal{B} := \{ \text{non-zero prime ideals } p \text{ with } N(p) \leq B \} \). We refer to \( B \) as the smoothness bound. We compute a generating set of the lattice \( \Lambda \) of all the vectors \((e_1, \ldots, e_m) \in \mathbb{Z}^m \) with \( m := |\mathcal{B}| \) such that \( \exists \alpha \in K, \ (\alpha) = p_1^{e_1} \cdots p_m^{e_m} \).

**Definition 2.3** (relations). Let \( S = \{ p_1, \ldots, p_s \} \) be a set of non-zero prime ideals of \( K \). For each \( S\)-unit \( \alpha \in K \) with \( \bar{e} = (e_1, \ldots, e_s) \) such that \( (\alpha) = \prod_i p_i^{e_i} \), we define the relation associated with \( \alpha \) by \( \mathcal{R}_{S,K}(\alpha) := (\alpha, \bar{e}) \). The relations of \( K \) for the set \( S \) form a group denoted by \( \mathcal{R}el_S(K) \).

When \( B > 12 \log^2 |\Delta| \), the classes of ideals in \( \mathcal{B} \) generate \( \text{Cl}(\mathcal{O}_K) \) under the GRH [1, Th. 4]. Therefore, \((\mathcal{B}, \Lambda)\) is a presentation of the group \( \text{Cl}(\mathcal{O}_K) \) and the search for a generating set of the relations \( \mathcal{R}el_S(K) \) for \( S = \mathcal{B} \) is equivalent to computing the group structure of \( \text{Cl}(\mathcal{O}_K) \). Indeed, the morphism

\[
\begin{align*}
\mathbb{Z}^m & \xrightarrow{\varphi} \mathcal{I} \xrightarrow{\pi} \text{Cl}(\mathcal{O}_K) \\
(e_1, \ldots, e_m) & \mapsto \prod_i p_i^{e_i} \mapsto \prod_i [p_i]^{e_i}
\end{align*}
\]

is surjective, and the class group \( \text{Cl}(\mathcal{O}_K) \) is isomorphic to \( \mathbb{Z}^m / \ker(\pi \circ \varphi) = \mathbb{Z}^m / \Lambda \).

**2.6. \( S \)-class groups and \( S \)-unit groups.** Let \( S = \{ p_1, \ldots, p_s \} \) be a finite set of prime ideals of the number field \( K \). We say that \( x \in K \) is an \( S \)-integer if \( \nu_p(x) \geq 0 \) for all \( p \notin S \). The set of \( S \)-integers is a ring denoted by \( \mathcal{O}_{K,S} \). We define the \( S \)-unit group \( U_{K,S} \) (or \( U_S \) if the field of definition is understood) as the elements \( x \in K \) such that \( \nu_p(x) = 0 \) for all \( p \notin S \). The group of \( S \)-units is finitely generated: \( U_S = \mu(K) \times \langle \eta_1 \rangle \times \cdots \times \langle \eta_{r+s} \rangle \) where \( \mu(K) \) is the set of the roots of unity of \( K \), and \( \eta_1, \ldots, \eta_{r+s} \) are torsion free generators. The rank of its torsion-free part equals \( r + s \) where \( r \) is the rank of the torsion free part of the unit group \( U_K \). Let \( \mathcal{I}_S \) be the group of fractional ideals of \( \mathcal{O}_{K,S} \), and \( \mathcal{P}_S \) its subgroup of principal ideals. We define the \( S \)-class group by \( \text{Cl}_S(\mathcal{O}_{K,S}) = \mathcal{I}_S / \mathcal{P}_S \).

**3. \( S \)-units of quadratic fields**

In this section, we assume that \( L = \mathbb{Q}(\sqrt{d}) \) for \( d > 0 \) a squarefree integer. The calculation of the \( S \)-unit group for \( S \) a set of prime ideals of \( L \) is done by using the approach of Simon [27, Sec. 1.1.2]. Together with the subexponential strategy for computing the ideal class group derived from the Hafner-McCurley algorithm [21], the \( S \)-unit group of \( L \) can be computed in time \( \text{Poly}(\text{Size}(S)) \cdot e^{O(\sqrt{\log d})} \). These algorithms have been extensively studied, in particular in [21, 15, 9, 27]. Therefore, we only give a brief sketch of the algorithm and will focus on the run time and the format of the output.
3.1. Computing the class group. First, let $B \in e^{\tilde{O}(\sqrt{\log d})}$ be a large enough smoothness bound such that the non-zero prime ideals $p_1, \ldots, p_k$ of $L$ with norm less than $B$ generate $\text{Cl}(\mathcal{O}_L)$. Note that $k \in e^{\tilde{O}(\sqrt{\log d})}$. The computation of $\text{Cl}(\mathcal{O}_L)$ starts with the collection of $\delta_i, \ldots, \delta_l$ for some $l \in \tilde{O}(k)$ such that for all $i \leq l$ there exist $(a_{i,1}, \ldots, a_{i,k})$ with $(\delta_i) = \prod_j p_j^{a_{i,j}}$. The $\delta_i$ and the $a_{i,j}$ are all polynomial size in $\log(d)$. Then there are unimodular matrices $U \in \text{GL}_d(\mathbb{Z})$ and $V \in \text{GL}_k(\mathbb{Z})$ such that

$$\text{SNF}(A) = UAV = \begin{pmatrix}
 d_1 & (0) \\
 \vdots & \ddots \\
 (0) & \cdots & d_k \\
 (0) & \cdots & \cdots & (0)
\end{pmatrix},$$

where $\text{SNF}(A)$ denotes the Smith Normal Form of $A$. The unimodular matrices $U, V$ can be found in polynomial time [28] (in the dimension and the bit size of the entries of $A$), and their entries have polynomial size in the dimension of $A$ and the bit size of its coefficients. This means that $\log(|U|), \log(|V|) \in e^{\tilde{O}(\sqrt{\log d})}$, where $|U|$ denotes a bound on the absolute values of the entries generated by $U$. Let $\mathcal{L} \subseteq \mathbb{Z}^k$ be the lattice generated by the rows of $A$. Then $\text{Cl}(\mathcal{O}_L) \simeq \mathbb{Z}^k / \mathcal{L} \simeq \mathbb{Z}/d_1 \mathbb{Z} \oplus \cdots \oplus \mathbb{Z}/d_k \mathbb{Z}$. Let $g_j := \prod_{i \leq k} p_i^{a_{i,j}}$, we have $\text{Cl}(\mathcal{O}_L) \simeq \langle \langle g_1 \rangle \rangle \times \cdots \times \langle \langle g_k \rangle \rangle$. In addition, let $\beta_i := \prod_{j \leq l} g_j^{\beta_{i,j}}$, for $i \leq k$. We do not evaluate this product. We have $g_i^{\beta_i} = (\beta_i)$. Overall, the complexity of this calculation is in $e^{\tilde{O}(\sqrt{\log d})}$.

3.2. Computing the S-unit group. Let $S$ be a set of primes $q_1, \ldots, q_s$ of $L$. To get the $S$-class group and the $S$-unit group we add extra relations to $\mathcal{L}$. More specifically, we need to identify the classes of $\text{Cl}(\mathcal{O}_L)$ that are represented by a product of primes in $S$ with the trivial class of $\text{Cl}_S(\mathcal{O}_{L,S})$. The ideal class of each of the elements of $S$ can be represented as a product of the classes of the $g_i$. In time $e^{\tilde{O}(\sqrt{\log d})}$ (and polynomial in $\log(\mathcal{N}(q_i))$), one can find polynomial size $x_1, \ldots, x_k$ and $\beta_{i+k} \in L$ such that $q_i = (\beta_{i+k}) \prod_j p_j^{x_{i,j}}$ with standard methods derived from [21].

Then for each $j$, $p_j = \prod_{i \leq k} g_i^{v_{i,j}}$, where the $v_{i,j}$ are the coefficients of $V^{-1}$, we readily find vectors $\vec{e}_i \in \mathbb{Z}^k$ with entries having polynomial size in $k$ (that is in $e^{\tilde{O}(\sqrt{\log d})}$) such that $q_i = (\beta_{i+k}) \prod_{j \leq k} g_j^{v_{i,j}}$. The vectors $\vec{e}_i$ are precisely the new additions needed to expand $\mathcal{L}$. We get a new relation matrix

$$B = \begin{pmatrix}
 d_1 & (0) \\
 \vdots & \ddots \\
 (0) & \cdots & d_k \\
 e_{1,1} & \cdots & e_{1,k} \\
 \vdots & \vdots & \vdots \\
 e_{s,1} & \cdots & e_{s,k}
\end{pmatrix}.$$ 

As for the computation of $\text{Cl}(\mathcal{O}_L)$, the SNF of $B$ gives the elementary divisors of the cyclic decomposition of $\text{Cl}_S(\mathcal{O}_{K,S})$. Meanwhile, let $\vec{w}_1, \ldots, \vec{w}_{1+s}$ be a basis for the left kernel of $B$ (in general the dimension is $r+s$ where $r$ is the rank of the unit group of $L$). This kernel is found in polynomial time in the dimension of $B$ and the size of its entries, that is in time $\text{Poly}(s) \cdot e^{\tilde{O}(\sqrt{\log d})}$. The entries of the kernel
vectors have size in \( \text{Poly}(s) \cdot e^{O(\sqrt{\log d})} \), and \( \mathbb{U}_S = \mu \times \langle \gamma_1 \rangle \times \ldots \times \langle \gamma_{1+s} \rangle \) where \( \mu = \{ \pm 1 \} \) are the torsion units of \( \mathcal{O}_L \) and \( \gamma_i := \prod_{j \leq k+i} \delta_{j}^{\delta_{j-i}} \).

**Proposition 3.1.** Let \( d > 0 \) be a squarefree integer, \( L = \mathbb{Q}(\sqrt{d}) \) and \( S \) be a set of prime ideals of \( L \) with \( |S| = s \). Then the \( S \)-unit group algorithm of [27, Sec. I.1.2] returns \( \ell \in e^{O(\sqrt{\log d})} \) polynomial size elements \( \delta_i \in L \) and \( s + 1 \) vectors \( \tilde{c}_i \) with entries of size in \( \text{Poly}(s) \cdot e^{O(\sqrt{\log d})} \) such that the \( s + 1 \) elements \( \gamma_i := \prod_{j \leq k+i} \delta_{j}^{\delta_{j-i}} \) generate the \( S \)-unit group of \( L \). The overall complexity of this procedure is in \( \text{Poly}(\text{Size}(S)) \cdot e^{O(\sqrt{\log d})} \) where the size of \( S \) is in \( O(s \cdot \max_{p \in S} \log(N(p))) \).

4. **Recursive computation of \( S \)-units**

Let \( S \) be a set of non-zero prime ideals in \( L \) that is invariant under the action of \( \text{Gal}(L/\mathbb{Q}) \) (that is, \( \forall p \in S, \forall \sigma \in \text{Gal}(L/\mathbb{Q}), p^{\sigma} \in S \)). In this section, we introduce a recursive method for finding a generating set of \( \text{Rel}_S(L) \) which is the group of elements of the form \( \mathbb{R}_{S,L}(\alpha) = (\alpha, \tilde{e}) \) such that \( (\alpha) = \prod_{p \in S} p^{e_p} \). Our strategy consists in deriving the \( S \)-unit group in \( L \) from that of three subfields of \( L \). When we reach the leaves of this recursion tree, we use the methods of Section 3 for computing the \( S \)-unit group directly on the quadratic field.

### 4.1. High level description of the algorithms

Let \( L \) be a multiquadratic number field and let \( \sigma, \tau \) be two distinct non-trivial automorphisms of \( L \). Let \( \sigma \tau := \sigma \circ \tau \) and \( K_\ell \) be the subfield of \( L \) fixed by \( \ell \in \{ \sigma, \tau, \sigma \tau \} \). Let \( S \) be a set of prime ideals of the ring of integers \( \mathcal{O}_L \) of \( L \) stable by the action of \( \text{Gal}(L/\mathbb{Q}) \), and for each \( \ell \in \{ \sigma, \tau, \sigma \tau \} \) let us define \( S_\ell := \{ p \cap K_\ell \mid p \in S \} \). We recover a generating set of \( \text{Rel}_S(L) \) from generating sets of \( \text{Rel}_{S_{\sigma}}(K_{\sigma}), \text{Rel}_{S_{\tau}}(K_{\tau}), \) and \( \sigma(\text{Rel}_{S_{\sigma \tau}}(K_{\sigma \tau})) \).

Our result follows from two crucial observations:

1. The subgroup \( U \) of \( \text{Rel}_S(L) \) generated by the lifts of \( \text{Rel}_{S_{\sigma}}(K_{\sigma}), \text{Rel}_{S_{\tau}}(K_{\tau}), \) and \( \sigma(\text{Rel}_{S_{\sigma \tau}}(K_{\sigma \tau})) \) contains all the squares of relations in \( \text{Rel}_S(L) \).

2. There is an algorithm that efficiently produces elements of \( U \) that are square of relations in \( \text{Rel}_S(L) \), and then computes their square root.

When the recursive tree reaches a quadratic subfield \( K_\ell \) of \( L \), it uses the subexponential algorithm of Simon [27, Sec. I.1.2] to return the \( S_\ell \)-unit group. The high level description of this strategy is summarized in Algorithm 4.1. Note that the ring of integers \( \mathcal{O}_L \) is part of the input. In general, the computation of \( \mathcal{O}_L \) is as hard as the factorization of the discriminant of \( L \), but in the particular case of multiquadratic fields, there is an efficient algorithm for this task [18].

### 4.2. Lifting relations

To compute \( \text{Rel}_S(L) \), we use relations from \( \text{Rel}_{S_{\sigma}}(K_{\sigma}), \text{Rel}_{S_{\tau}}(K_{\tau}), \) and \( \sigma(\text{Rel}_{S_{\sigma \tau}}(K_{\sigma \tau})) \) where \( \sigma, \tau \in \text{Gal}(L/\mathbb{Q}) \) and \( S_{\sigma}, K_{\sigma} \) are defined in Section 4.1.

Therefore, given relations in a subfield \( K_\sigma \) of \( L \), we need to be able to efficiently compute the corresponding relations in \( L \).

**Theorem 4.1.** Let \( L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n}) \) be a multiquadratic field. Let \( K_{\sigma} \) be the \( (\text{multi}) \)-quadratic subfield of \( L \) fixed by \( \sigma \in \text{Gal}(L/\mathbb{Q}) \), \( S_{\sigma} = \{ p_i \}_{1 \leq i \leq s} \) where \( p_i \) are prime ideals of \( K_{\sigma} \) and \( S = \{ \mathbb{P}_k \subset L \mid \exists i \leq s, \mathbb{P}_k \cap K_{\sigma} = p_i \} \). Let \( \mathbb{R}_{S_{\sigma}, K_{\sigma}}(\alpha) = (\alpha, \tilde{e}) \) be a relation in \( \text{Rel}_{S_{\sigma}}(K_{\sigma}) \). Then \( (\alpha, \tilde{e}_L) := \mathbb{R}_{S,L}(\alpha) \in \text{Rel}_S(L) \) with \( \tilde{e}_L = (e_1\tilde{f}_1|e_2\tilde{f}_2|\ldots|e_s\tilde{f}_s) \), where \( \tilde{f}_i \) satisfy \( p_i \mathcal{O}_L = \prod_{j \leq k+i} \mathbb{P}_{k+i,j}^{e_{i,j}} \).
Lemma 4.2. Let $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ be a multiquadratic field and let $S$ be a set of prime ideals of $L$ that is invariant under the action of $\text{Gal}(L/\mathbb{Q})$. Let $\sigma, \tau \in \text{Gal}(L/\mathbb{Q})$ be two different non-identity isomorphisms, and define $S_\ell, K_\ell$ of $\ell \in \{\sigma, \tau, \sigma \tau\}$ as in Section 4.1. Let $U$ be the group generated by $\mathcal{R}_{S_\ell}(K_\sigma) \cup \mathcal{R}_{S_\ell}(K_\tau) \cup \sigma(\mathcal{R}_{S_\ell}(K_{\sigma \tau}))$ where

$$\sigma(\mathcal{R}_{S_\ell}(K_{\sigma \tau})) := \{ \mathcal{R}_{S_\ell, K_\ell}(\sigma(\alpha)) \mid \exists \bar{e}, (\alpha, \bar{e}) \in \mathcal{R}_{S_\ell, K_\ell}(K_{\sigma \tau}) \}$$

Then $(\mathcal{R}_{S_\ell}(L))^2 \subseteq U \subseteq \mathcal{R}_{S_\ell}(L)$, where $(\mathcal{R}_{S_\ell}(L))^2$ denotes the relations of the form $(\alpha^2, 2\bar{e})$ where $(\alpha, \bar{e}) \in \mathcal{R}_{S_\ell}(L)$.

Proof. From Theorem 4.1, we know that the relations in $\mathcal{R}_{S_\ell}(K_\sigma)$, $\mathcal{R}_{S_\ell}(K_\tau)$ and $\mathcal{R}_{S_\ell}(K_{\sigma \tau})$ lift naturally to relations in $\mathcal{R}_{S_\ell}(L)$. Moreover, $\sigma$ maps elements of $K_{\sigma \tau}$ to $K_{\tau}$, and since $S$ is invariant under the action of $\sigma$, a relation is mapped to another relation (modulo a permutation of the coefficients of the exponent vector). So the action of $\sigma$ on $\mathcal{R}_{S_\ell}(K_{\sigma \tau})$ is well defined, and $U \subseteq \mathcal{R}_{S_\ell}(L)$.

For the other inclusion, let $(\alpha, \bar{e}) \in \mathcal{R}_{S_\ell}(L)$. For each $\ell \in \{\sigma, \tau, \sigma \tau\}$, $\alpha \cdot \ell(\alpha)$ decomposes as a product of ideals in $S_\ell$. Therefore, there are vectors $\bar{e}_\ell$ such that

\[ (\alpha, \bar{e}) \in \mathcal{R}_{S_\ell}(L) \Rightarrow (\alpha^2, 2\bar{e}_\ell) \in (\mathcal{R}_{S_\ell}(L))^2 \]
for each \( \ell, (\alpha \cdot \ell(\alpha), \bar{e}_\ell) \in \mathcal{R}_{\mathcal{L}}(K_\ell) \). Moreover,
\[
\frac{N_{L/K_\ell}(\alpha)N_{L/K_\ell}(\alpha)}{\sigma(N_{L/K_\ell}(\alpha))} = \frac{\alpha \cdot \sigma(\alpha) \cdot \alpha \cdot \tau(\alpha)}{\sigma(\alpha \cdot \sigma(\alpha))} = \alpha^2,
\]
hence \((\alpha^2, 2\bar{e}) = (\sigma(\alpha), \bar{e}_\sigma) + (\tau(\alpha), \bar{e}_\tau) - \sigma((\sigma \tau)(\alpha), \bar{e}_{\sigma \tau})\) is a linear combination of relations in \( \mathcal{R}_{\mathcal{L}}(K_\ell), \mathcal{R}_{\mathcal{L}}(K_\ell) \) and \( \sigma(\mathcal{R}_{\mathcal{L}}(K_{\sigma \tau})) \), so \((\mathcal{R}_{\mathcal{L}}(L))^2 \subseteq U\).

4.3. Representation of elements and square roots. The lifting \( U \) of the relations in three different subfields yield a set of relations containing all the squares of the relations in \( \mathcal{R}_{\mathcal{L}}(L) \). We need to solve two tasks:

1. Identification of a generating set of the squares of \( U \).
2. For each square \((\alpha^2, 2\bar{e})\) found in (1): computation of \((\alpha, \bar{e})\).

\( p \)-th roots with saturation. Let us identify \( U \subseteq \mathcal{R}_{\mathcal{L}}(L) \) with the elements \( \alpha \in U_S \) such that \( \exists \bar{e}, (\alpha, \bar{e}) \in U \). Let \( b > 0 \) such that \( (U_S : U) = b \). For any prime \( \ell \mid b \) there is some \( \alpha \in U_S \setminus U \) such that \( \alpha^b \in U \). The saturation technique of Biasse and Fieker [8] can be used to find elements in \( U_S \) that are not in \( U \). Let us fix the prime \( p \). For any residue degree 1 prime ideal \( \mathfrak{Q} \not\in S \) with \( Q := \mathcal{N}(\mathfrak{Q}) \) such that \( \ell pQ - 1 \) we define the map \( \phi_{\mathfrak{Q}} : U \rightarrow \mathbb{F}_q^*/(\mathbb{F}_q^*)^p \) mapping \( S \)-units into the multiplicative group of the residue class field \( \mathbb{F}_q := \mathbb{O}_L/\mathfrak{Q} \) modulo \( p \)-th powers. The Cebotarev theorem [17] guarantees that if \( \alpha \in U \) is not a \( p \)-th power, there will be some \( \mathfrak{Q} \) such that \( \phi_{\mathfrak{Q}}(\alpha) \) is non-trivial, i.e. \( \alpha \) is not a \( p \)-th power modulo \( Q \). To find \( p \)-th powers, we now simply intersect \( \ker \phi_{\mathfrak{Q}} \) for sufficiently many \( \mathfrak{Q} \).

The elements \( \alpha \in U/\langle \ker \phi_{\mathfrak{Q}} \rangle \) will have a \( p \)-th root in \( U_S \) but not in \( U \). Suppose \((\alpha, \bar{e}) \in U \) with \( \alpha \in U/\langle \ker \phi_{\mathfrak{Q}} \rangle \), then \((\sqrt[p]{\alpha}, \bar{e}/p)\) is a new relation that reduces the index of the lattice of currently found relations in \( \mathcal{R}_{\mathcal{L}}(L) \).

Using quadratic characters for \( p = 2 \). When looking for square roots, we can use quadratic characters to find elements in elements \( \alpha \in U/\langle \ker \phi_{\mathfrak{Q}} \rangle \) by following the approach of [3]. More specifically, in [3, Sec. 4.1], the map
\[
\phi_{\mathfrak{Q}} : \mathbb{Z}[x_1, \ldots, x_n]/(x_1^2 - d_1, \ldots, x_n^2 - d_n) \cong \mathbb{Z} \left[ \sqrt{d_1}, \ldots, \sqrt{d_n} \right] \rightarrow \mathbb{F}_q,
\]
where \( \mathfrak{Q} \) is a residue degree 1 prime ideal and \( Q := \mathcal{N}(\mathfrak{Q}) \), is defined by \( x_i \mapsto s_i \) where \( s_i \) is a square root of \( d_i \) modulo \( Q \). Elements of \( U_S \) have non-negative valuation at \( \mathfrak{Q} \) since it satisfies \( \mathfrak{Q} \not\in S \). We can use the characters defined in [3, Sec. 4.1] by \( \chi_{\mathfrak{Q}}(\alpha) := \left( \frac{\phi_{\mathfrak{Q}}(\alpha)}{Q} \right) \in \{-1, 0, 1\} \). When \( \alpha \) is a square, we have \( \chi_{\mathfrak{Q}}(\alpha) = 1 \).

To find squares, we find the \( \alpha \in U \) such that \( \chi_{\mathfrak{Q}}(\alpha) = 1 \) for \( i \leq m \) where \( m \) is large enough. This boils down to the search for a kernel element of the linear map
\[
U_S \xrightarrow{X} \mathbb{Z}/2\mathbb{Z} \times \cdots \times \mathbb{Z}/2\mathbb{Z} \quad \alpha \quad \mapsto \quad (\log_{-1}(\chi_{\mathfrak{Q}}(\alpha)), \ldots, \log_{-1}(\chi_{\mathfrak{Q}}(\alpha)))^T,
\]
where for each \( x \in \{-1, 1\} \), \( \log_{-1}(x) \) denotes the discrete logarithm of \( x \) in base \(-1\). If \( \alpha \) is a square, then necessarily \( X(\alpha) = (0, \ldots, 0) \). On the other hand, if \( X(\alpha) = (0, \ldots, 0) \), there is a non-zero probability that \( \alpha \) might not be a square. Given generators \( \alpha_1, \ldots, \alpha_k \) of \( U \), we can find a generating set of the squares of elements of \( U_S \). This contains the squares of elements \( \alpha_{k+1}, \ldots, \alpha_{k+l} \) of \( U_S \) such that \( \alpha_1, \ldots, \alpha_{k+l} \) generate \( U_S \). We obtain these squares by finding the kernel of the matrix \( A = (X(\alpha_i)) \in \mathbb{Z}^{k \times m} \).
**Representation of the elements.** We compute $S$-units in the quadratic fields by directly applying the subexponential algorithm of [27, Sec. 1.1.2]. As we saw in Section 3, the output of the computation in each quadratic field $K_l := \mathbb{Q}(\sqrt{d_l})$ for $l \leq 2^n := N$ is a set of $s+1$ elements $\gamma_i$ that are represented by vectors of exponents $\vec{c}_i$ and $k$ elements $\alpha_j$ such that $\gamma_i = \prod_{j \leq k} \delta_j^{\vec{c}_i}$. The $\delta_j$ have polynomial size in $\log(d_i)$, while $\vec{c}_i \in \mathbb{Z}^N$ and the entries of $\vec{c}_i$ have size in $\operatorname{Poly}(s) \cdot O(\sqrt{\log(d_i)})$. In our algorithm these products are never evaluated in $L$. By linearity, one can evaluate $X(\gamma_i) = \sum_{j \leq k} e_{i,j} X(\delta_j)$ in time $k \cdot \operatorname{Poly}(\max_{i,j} \operatorname{Size}(e_{i,j})) \cdot \operatorname{Poly}(\max_i \operatorname{Size}(X(\delta_j)))$. As $\operatorname{Size}(X(\delta_j))$ is bounded by $m \cdot \max_i \log(\mathcal{N}(\Omega_i))$, the resulting complexity is in \( \operatorname{Poly}(s,m,\log Q) \cdot e^{\tilde{O}(\sqrt{\log(d)})} \) where $Q := \max_i \mathcal{N}(\Omega_i)$ and $d := \max_i d_i$.

When working in a subfield $K$ of $L$ of degree $N_i = 2^i$ for $2 \leq i \leq n$, we represent the elements $\alpha \in U_{K,S}$ as products of the $\gamma_i$ for $\sigma \in \text{Gal}(L/Q)$ and $i \leq (s+1)N_i$ where $\gamma_{j+1}, \ldots, \gamma_{j+(s+1)}$ generate the $S$-unit group of the $j$-th quadratic subfield. Each lifting involves square roots. We do not evaluate the product of the $\gamma_i$, nor the square roots. To represent $\alpha \in U_{K,S}$, we use the vector $\vec{c} \in \mathbb{Z}^{N_i}$ such that $\alpha = \left( \prod_{i,\sigma} (\gamma_i^\sigma)^{\vec{c}} \right)^{\frac{1}{2}}$. Under this representation, the product of two elements, the image under a morphism $\sigma \in \text{Gal}(L/Q)$ and the computation of the square root are straightforward operations with complexity in $\operatorname{Poly}(s,N,C) \cdot e^{\tilde{O}(\sqrt{\log(d)})}$ where $C$ is an upper bound on the size of the coefficients of the exponent vectors $\vec{c}$. On the other hand, it is more delicate to compute $X(\alpha)$. For each $\Omega_i$, we can compute $\phi_{\Omega_i} \left( \prod_{i,\sigma} (\gamma_i^\sigma)^{\vec{c}} \right)$ in time $\operatorname{Poly}(s,N,\log Q) \cdot e^{\tilde{O}(\sqrt{\log d})}$. Evaluating a $2^{i-1}$-th root of this value in $\mathbb{F}_{Q_i}$ has the same complexity. However, there are $2^{i-1}$ different possible roots, and it is impossible to tell which one is the image of $\alpha$ under the map $\phi_{\Omega_i}$ without actually evaluating the product (and square roots) defining $\alpha$ (something we cannot afford for complexity reasons). To circumvent this, we choose the $Q_i := \mathcal{N}(\Omega_i)$ such that all of the different $2^{i-1}$-th roots in $\mathbb{F}_{Q_i}$ have the same Legendre symbol for $2 \leq i \leq n$. To do this, we need to ensure that $-1$ is a $2^{m-1}$-th power in $\mathbb{F}_{Q_i}$. This is ensured when there is a primitive $2^n$-th root of unity in $\mathbb{F}_{Q_i}$, or equivalently, when $2^n \mid Q_i - 1$.

---

**Algorithm 4.2: S-UnitsGivenSubgroup**

**Input:** Real multiquadratic field $K \subseteq \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$, $\alpha_1, \ldots, \alpha_k$ such that $U_{K,S} \subseteq (\alpha_1, \ldots, \alpha_k)$

**Result:** Generators of $U_{K,S} / \{ \pm 1 \}$

1. $\chi_1, \ldots, \chi_m \leftarrow$ characters defined by $\Omega_i$ with $2^n \mid \mathcal{N}(\Omega_i) - 1$ for $i \leq m$.
2. $A \leftarrow \left[ \log_{-1} (\chi_i(\alpha_j)) \right]_{i \leq m, j \leq k} \in \mathbb{F}_{2^{m \times k}}$.
3. $V \leftarrow$ Basis of the left kernel of $A$
4. for $i = 1, \ldots, \#V$
5. \[ v_i \leftarrow \prod_j \alpha_j^{V_{ij}}. \]
6. \[ b_i \leftarrow \sqrt{v_i}. \]
7. return $\alpha_1, \ldots, \alpha_k, b_1, \ldots, b_{\#V}$

In the description of Algorithm 4.2, we identify field elements and their representation described above. As previously mentioned, all squares must map to elements...
of \text{LeftKernel}(A)$, but there is a chance that elements from \text{LeftKernel}(A)$ do not arise as the map of a square in $K$. In this case, the element $s_i$ calculated in Step 5 is not a square, and the (formal) square root computed in Step 6 does not correspond to any element in $K$. The probability of success of Algorithm 4.2 derives from a standard heuristic used for the computation of square roots in the Number Field Sieve algorithm [16, Sec. 8]. This argument was also used for computing units of multiquadratic fields in [3, Sec. 4.2]. Let $U := \langle \alpha_1, \ldots, \alpha_k \rangle / \langle \pm 1 \rangle$. The rank of $U/(U \cap K^2)$ is at most $s + r$ where $r$ is the rank of the unit group of $K$ and $s := |S|$. Therefore, the dual $\text{Hom}(U/(U \cap K^2), F_2)$ is an $F_2$ vector space of dimension at most $r + s$. Assuming that $\log_1 \chi_{\Omega_1}, \ldots, \log_1 \chi_{\Omega_m}$ are independent uniform random elements of this dual, they span the dual with probability at least $1 - 1/2^{m-r-s}$ by [16, Lem. 8.2]. In that case, $X(\alpha) = 0$ implies $\alpha \in U \cap K^2$. Note that when we enforce the restriction, $2^n \mid Q_i - 1$ for $i \leq m$, elements divisible by 2 will always be in the kernel of the $\log_1 \chi_{\Omega_i}$, therefore the heuristic according to which the $(\log_1 \chi_{\Omega_i})_{1 \leq m}$ span the dual only makes sense when no element in $U$ is divisible by 2, which is ensured when $S$ does not contain any ideal above 2.

**Heuristic 4.3.** Let $K$ be a multiquadratic subfield of $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$, and let $S$ be a set of prime ideals of $K$ that does not contain any ideal above 2. Let $\alpha_1, \ldots, \alpha_k$ be elements generating $U_{K,S}^2$ and let $U := \langle \alpha_1, \ldots, \alpha_k \rangle / \langle \pm 1 \rangle$ Then morphisms of the form $\log_1 \chi_{\Omega}$, where $2^n \mid N(\Omega_i) - 1$ are uniformly distributed in $\text{Hom}(U/(U \cap K^2), F_2)$.

**Proposition 4.4.** Let $K$ be a multiquadratic subfield of $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$, and let $S$ be a set of prime ideals of $K$ that does not contain any ideal above 2. Let $\alpha_1, \ldots, \alpha_k$ be elements generating $U_{K,S}^2$. Let $r$ be the rank of the unit group of $K$ and let $s := |S|$. Then the run time of Algorithm 4.2 is in $\text{Poly}(r, m, N, C, \log Q) \cdot e^{O(\sqrt{\log d})}$ where $m$ is the number of characters, $N = 2^n$, $Q = \max_{i \leq m} Q_i$, $C$ is an upper bound on the bit size of the coefficients of the vectors defining the $\alpha_i$ and $d = \max_{i \leq n} d_i$. Algorithm 4.2 returns a generating set of $U_{K,S}$ with probability at least $1 - 1/2^{m-r-s}$ under Heuristic 4.3. Moreover, the size of the coefficients defining the $\beta_i$ is bounded by $kC$.

**Remark 4.5.** The only subroutine that we have not formally analyzed is the creation of the $\chi_1, \ldots, \chi_m$. For that, we directly rely on the algorithm \text{GoodPrime} of [3]. It returns each prime in time $O(N)$. Thus, the calculation of $\chi_1, \ldots, \chi_m$ is in $O(mN)$.

### 4.4 Overall procedure.
We now have all the ingredients to specify the details of our recursive method to compute the $S$-unit group of $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ for a set of prime ideals $S$ invariant under the action of the Galois group of $L$.

**Theorem 4.6.** Let $L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ be a real multiquadratic field of degree $N$ and $S$ be a set of prime ideals of $L$ stable under $\text{Gal}(L/\mathbb{Q})$ that does not contain any ideal above 2. Then under Heuristic 4.3, the elements $\beta_1, \ldots, \beta_{r+s}$ returned by Algorithm 4.3 generate the torsion free part of $U_S$ with probability $1 - \frac{1}{N}$. The asymptotic complexity of Algorithm 4.3 is in $\text{Poly}(\text{Size}(S), \log(\Delta)) \cdot e^{O(\sqrt{\log d})}$ where $\text{Size}(S) = s \cdot \max_{p \in S} \log(N(p))$, $\Delta = \text{disc}(L)$, and $d := \max_{i \leq n} d_i$.

**Proof.** Algorithm 4.3 is called $3^n \in \mathbb{N}$ times. The run time of Algorithm 4.3 is essentially ruled by that of Algorithm 4.2 and by the cost of Steps 12 and 14.
Algorithm 4.3: MQSunits for $S$ stable under $\text{Gal}(L/\mathbb{Q})$

Input: Real multiquadratic field $L$, ring of integers $\mathcal{O}_L$ of $L$, and set of prime ideals $S$ of $\mathcal{O}_L$ stable under $\text{Gal}(L/\mathbb{Q})$.

Result: A basis of the relations $\mathcal{R}el_S(L)$.

1. $S_0 \leftarrow \{p_1, \ldots, p_s\}$ where $\forall i \leq s$, $\exists p \in S$, $p_i \mid p$.
2. if $[L : \mathbb{Q}] = 2$ then
   3. $\Lambda \leftarrow \text{basis of } \mathcal{R}el_S(L)$ using [27, Alg. I.1.2].
   4. return $\Lambda$
5. $\sigma, \tau \leftarrow$ distinct non-identity automorphisms of $L$
6. for $\ell \in \{\sigma, \tau, \sigma \tau\}$ do
7. $K_\ell \leftarrow$ fixed field of $\ell$
8. $S \leftarrow \{p \subseteq K_\ell \mid \exists e \in \mathbb{Z}, p \in S_0, N(p) = p^e\}$
9. $\Lambda_\ell \leftarrow \text{MQSunits}(K_\ell, S)$
10. $\Lambda_U \leftarrow \Lambda_\sigma \cup \Lambda_\tau \cup \sigma(\Lambda_{\sigma \tau})$
11. $\Lambda := \{(\alpha_1, \vec{e}_1), \ldots, (\alpha_k, \vec{e}_k)\} \leftarrow \text{SunitGivenSubgroup}(L, \Lambda_U)$ (Alg. 4.2)
12. $A \leftarrow (\vec{e}_i)_{i \leq k}$. Compute $U \in \text{GL}_k(\mathbb{Z})$ such that $UA = \left(\frac{U}{M}\right)$ is the HNF of $A$
13. for $i = 1, \ldots, s$: $\beta_i \leftarrow \prod_{j \leq k} \alpha_{i,j}^{U_{i,j}}$
14. Compute a basis $\vec{w}_1, \ldots, \vec{w}_r$ of the left kernel of $A$
15. for $i = 1, \ldots, r$: $\beta_{s+i} \leftarrow \prod_{j \leq k} \alpha_{i,j}^{w_{i,j}}$
16. return $(\beta_1, \vec{H}_1), \ldots, (\beta_s, \vec{H}_s), (\beta_{s+1}, \vec{0}), \ldots, (\beta_{s+r}, \vec{0})$

Moreover, the cost of the ideal arithmetic involved in the lifting of the relations is in $\text{Poly}(\text{Size}(S), \log(\Delta))$. The probability of success of the overall algorithm is at least $(1 - \frac{1}{e})^N \sim 1 - \frac{N}{e}$ where $r$ is the rank of the unit group of $L$. Therefore, a choice of $m \in \text{Poly}(N, s)$ can ensure that the probability of success is at least $1 - \frac{1}{e}$. With such a choice of $m$, we can also ensure that $Q \in \text{Poly}(N, s)$. Finally, the bit size $C$ of the coefficients of the representation of the elements in the relations only increase by a polynomial factor at every stage of the algorithm. In Algorithm 4.2, it gets multiplied by $k \leq 3(s + r)$, while in Steps 12 and 14, the coefficients of $U$ and of the $\vec{w}_i$ are in $\text{Poly}(s, \log(\Delta)) \cdot e^{O(\sqrt{\log s})}$. Moreover, the runtime of Steps 12 and 14 is also in $\text{Poly}(s, \log(\Delta)) \cdot e^{O(\sqrt{\log s})}$, which proves the statement. \qed

The result of Algorithm 4.3 can be certified in polynomial time under the Generalized Riemann Hypothesis if the prime ideals in $S$ generate the ideal class group of $L$. This is the case in all the applications that are considered in Section 5, including the computation of arbitrary $S$-unit groups. The only way Algorithm 4.3 can fail is if Algorithm 4.2 identifies non-squares as squares. If this is the case, then the set of relations returned by Algorithm 4.3 contains elements that are not in $\mathcal{R}el_S(L)$. Let $h_0 := \det(H)$ and $R_0$ be the volume of the lattice generated by $\text{Log}(\beta_i)$ for $i = s + 1, \ldots, s + r$. If the result is correct, then $h_0 = h$ the class number of $\mathcal{O}_L$ while $R_0 = R$ the regulator of $L$. If not, then $h_0 R_0 \leq \frac{h}{R}$ (i.e. $\mathcal{R}el_S(L)$ is a finite index subgroup of the output of Algorithm 4.3). An estimate for $h R$ can be found in polynomial time under the GRH by using the methods of [2].
12

JEAN-FRANÇOIS BIASSE AND CHRISTINE VAN VREENDAAL

Proposition 4.7. Under the GRH, the result of Algorithm 4.3 can be certified in polynomial time if \( S \) includes a generating set of the ideal class group of \( \mathcal{O}_L \).

5. APPLICATIONS OF THE \( S \)-UNIT COMPUTATION ALGORITHM

The \( S \)-unit group computation of Section 4 can be used to compute ideal class groups, \( S \)-class groups, and (arbitrary) \( S \)-unit groups.

5.1. Ideal class group computation. As explained in Section 2.5, the computation of \( \text{Cl}(\mathcal{O}_L) \) can be done by searching for a basis of the relations between a generating set \( \mathcal{O}_L \). Once such a generating set is found, then the strategy is the same as in [21], which was sketched in Section 3.

Algorithm 5.1: Computation of \( \text{Cl}(\mathcal{O}_L) \)

**Input:** Ring of integers \( \mathcal{O}_L \) of a real multiquadratic field \( L \) of degree \( N \) and discriminant \( \Delta \).

**Result:** Class group of \( \mathcal{O}_L \).

1. Compute \( S := \{ p \mid \mathcal{N}(p) \leq 3 \ln^2 (\Delta^2 \cdot 2^N), \ p \nmid 2 \} \).
2. \( (\alpha_1, \overline{H}_1), \ldots, (\alpha_s, \overline{H}_s), (\alpha_{s+1}, 0), \ldots, (\alpha_{s+r}, 0) \leftarrow \text{output of Algorithm 4.3} \)
3. \( \text{diag}(d_1, \ldots, d_s) \leftarrow \text{SNF}(H) \)
4. return \( \mathbb{Z}/d_1 \mathbb{Z} \times \ldots \times \mathbb{Z}/d_s \mathbb{Z} \)

Proposition 5.1. Let \( L = \mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n}) \) be a real multiquadratic field of degree \( N \) and discriminant \( \Delta \). Under the GRH, Algorithm 5.1 successfully returns the ideal class group of \( \mathcal{O}_L \) with probability \( 1 - \frac{1}{2^N} \) in time \( \text{Poly}(\log(\Delta)) \cdot e^{\tilde{O}(\sqrt{\log \Delta})} \) where \( d = \max_{i \leq n} d_i \). The result of Algorithm 5.1 can be certified in polynomial time in \( \log(\Delta) \).

Corollary 5.2. When \( d = \max_{i \leq n} d_i \) satisfies \( \log(d) < \log(\log(\Delta))^c \) for some constant \( c < 2 \), then Algorithm 5.1 returns the ideal class group of \( \mathcal{O}_L \) with probability \( 1 - \frac{1}{2^N} \) in polynomial time in \( \log(\Delta) \).

We showcase the effect of our algorithm on classes of multiquadratic fields with small \( d_i \) by the computation of the class group of the degree 128 multiquadratic field \( L = \mathbb{Q}(\sqrt{5}, \sqrt{13}, \sqrt{17}, \sqrt{29}, \sqrt{37}, \sqrt{41}, \sqrt{53}) \) and its subfields. We implemented Algorithm 5.1 and ran experiments on a single core of a HP ZBook 15 Mobile Workstation (Core i7 4800MQ - 16 GB RAM) and a single core of an AMD FX-8350 Vishera 4.0GHz CPU (32 GB RAM), both running version 7.5.1 of Sage [20]. For the low level multiquadratic arithmetic, we used the methods of [3]. For the Sage experiments the class_group(proof = False) method was used. Note that Sage’s class group routine directly calls that of Pari/GP [29]. We also ran the class group routine of Magma V.2.23 on the same fields on an Intel Core i7-2600 CPU 3.40GHz with 8 GB of RAM (PC3). Magma [14] works at a higher level of rigour by only returning results that are at least certified under GRH (we ran the command ClassGroup(K:Proof:="GRH")). Therefore the comparison with Sage is not entirely relevant. In degree 64, the computation with Magma had to be terminated after 24h since it had exhausted the machine’s memory. In the final version of the paper, all run times will be reported on the same architecture.

---

1We use [1, Th. 4] with \( f = (2) \mathcal{O}_L \) to avoid primes dividing 2
Although slower for small degrees, our method is the only implementation that is able to compute the class group of multiquadratic fields of degree more than 32. We can see on Table 5.1 that the run time (in CPU seconds) of Algorithm 5.1 is consistent with a polynomial run time in log(Δ). Our algorithm is parallelizable on several levels: subtrees of the recursion tree are independent, as well as computations modulo the (Q_i)_{i≤m}. Therefore, we anticipate that a parallel version of our algorithm could reach degrees 256 and 512.

<table>
<thead>
<tr>
<th>[L : Q]</th>
<th>Alg. 5.1 (Vishera)</th>
<th>Magma (PC3)</th>
<th>Sage (Vishera)</th>
<th>Cl(Δ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>99.9</td>
<td>1.4</td>
<td>0.25</td>
<td>trivial</td>
</tr>
<tr>
<td>16</td>
<td>648</td>
<td>12</td>
<td>0.91</td>
<td>C_4 \times C_4</td>
</tr>
<tr>
<td>32</td>
<td>5027</td>
<td>3615</td>
<td>77.7</td>
<td>C_2 \times C_4 \times C_8</td>
</tr>
<tr>
<td>64</td>
<td>4.0 \times 10^4</td>
<td>-</td>
<td>&gt; 8.5 \times 10^5</td>
<td>C_2^4 \times C_4 \times C_8 \times C_{16} \times C_{48} \times C_{240}</td>
</tr>
<tr>
<td>128</td>
<td>5.42 \times 10^5</td>
<td>-</td>
<td>-</td>
<td>C_2^{10} \times C_4^{16} \times C_8^{13} \times C_{16}^2 \times C_{48} \times C_{960}</td>
</tr>
</tbody>
</table>

Table 5.1. Comparison of class group routine run time

5.2. **S-class group and S-unit group computation.** Algorithm 4.3 computes the S-unit group with the restriction that S contains all conjugates of any p ∈ S under the action of Gal(L/Q). As shown in Section 3, the S-class group boils down to the search for the lattice of relations between the generators (q_j)_{i≤s_0} of Cl(Δ) which we enlarge with new relations of the form q_j ~ \prod_{i≤s_0} q_i^{x_{i,j}}. The SNF of this enlarged relation lattice gives the elementary divisors of the S-class group while its kernel reveals the S-unit group. Here, our only restriction on S is that it does not contain ideals above 2.

**Algorithm 5.2:** S-class group and S-unit group computation

**Input:** Real multiquadratic field L of degree N, ring of integers O_L of L, and a set S of prime ideals of O_L that does not contain ideals above 2.

**Result:** S-unit group and S-class group of L

1. Compute \( S_0 := \{ p \mid N(p) \leq 3 \ln^2 (\Delta \cdot 2^N) \}, \ p \not\in 2 \} \) for Δ = disc(L).
2. \( S := S \cup \{ q^\sigma \mid q \in S, \sigma \in \text{Gal}(L/Q) \} \).
3. \( (\alpha_1, \tilde{\alpha}_1), \ldots, (\alpha_{s_0}, \tilde{\alpha}_{s_0}), (\alpha_{s_0+1}, \tilde{0}), \ldots, (\alpha_{s_0+r}, \tilde{0}) \) ← output of Algorithm 4.3
4. Compute U, V such that \( U \begin{pmatrix} H \\ 0 \end{pmatrix} = \begin{pmatrix} \text{SNF}(H) \\ 0 \end{pmatrix} \) with SNF(H) = diag(d_i)_{i≤s_0}
5. For \( j ≤ s_0 \), define \( g_j := \prod_{i≤s_0} p_i^{V_{ij}} \) (here, Cl(Δ) = \( \bigoplus_{i≤k} (q_i) \))
6. \( V' ← V^{-1} \) For each \( j ≤ s \), find \( j_0 ≤ s_0 \) such that \( q_j = p_{j_0} \)
7. \( \tilde{x}_j ← (V_{1,j_0}, \ldots, V_{s_0,j_0}) \) (here \( q_j = \prod_{i≤s_0} x_{i,j_0} \))
8. Let \( M = \begin{pmatrix} H \\ 0 \end{pmatrix} \)
9. diag(d_i)_{i≤s_0} ← SNF(M). Compute a basis \( \bar{w}_1, \ldots, \bar{w}_s \) of the left kernel of M
10. For \( i ≤ s \), \( \alpha_i' ← \prod_{j≤s_0} \alpha_{ij} \).
11. For \( 1 ≤ i ≤ r \), \( \alpha_i' + s ← \alpha_{i+s} \) (the \( (\alpha_i')_s < i ≤ r + s \) generate U_L)
12. return \( (\alpha_1') \times \ldots \times (\alpha_{s_0}'), \mathbb{Z}/d_1' \mathbb{Z} \oplus \ldots \oplus \mathbb{Z}/d_{s_0}' \)
Proposition 5.3. Algorithm 5.2 is correct and returns the $S$-class group and the $S$-unit group with probability $1 - \frac{1}{2^N}$ where $N = [L; \mathbb{Q}]$ in time $\text{Poly}(\text{Size}(S), \log(\Delta)) \cdot e^{O(\sqrt{\log d})}$ where $\Delta = \text{disc}(L)$, and $d := \max_{i \leq n} d_i$.

REFERENCES


**University of South Florida, 4202 E Fowler Ave, Tampa, FL 33620, USA**

*E-mail address*: biasse@usf.edu

**Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, Netherlands**

*E-mail address*: c.v.vredendaal@tue.nl