Submitted exclusively to the London Mathematical Society
doi:10.1112/0000/000000

An algorithm for the principal ideal problem in indefinite quaternion algebras

A. Page

Abstract

Deciding whether an ideal of a number field is principal and finding a generator is a fundamental problem with many applications in computational number theory. For indefinite quaternion algebras, the decision problem reduces to that in the underlying number field. Finding a generator is hard, and we present a heuristically subexponential algorithm.

Keywords: quaternion algebra, principal ideal algorithm, factor base, Bruhat-Tits tree.

1. *Introduction*

Automorphic forms and their Hecke eigenvalues are of tremendous importance in number theory. These eigenvalues carry a lot of interesting arithmetic information, such as the number of points on elliptic curves or traces of Frobenius in Galois representations. One of the most successful methods for computing automorphic forms for GL_2 over number fields uses the Jacquet-Langlands correspondence. This result transfers the problem to a quaternion algebra, in which it is often easier to solve. This approach has its roots in the theory of Brandt matrices and has been successfully used by Dembélé-Donnelly and Greenberg-Voight [**5**] to compute Hecke eigenvalues of Hilbert modular forms. In both methods, a crucial step is to test whether an ideal is principal and to produce a generator in this case: this is the *principal ideal problem* that we are considering in this paper.

The principal ideal problem naturally splits into two cases: definite and indefinite algebras. In the definite case, Dembélé and Donnelly described an algorithm and Kirschmer and Voight proved that this algorithm runs in polynomial time when the base field is *fixed*, so we focus on the remaining indefinite case. In that case, testing whether an ideal is principal reduces to the same problem over the base field by Eichler's theorem (Theorem 2.3), but finding a generator is difficult. Kirschmer and Voight [**12**] provide an algorithm that improves on naive enumeration[†], without analysing its complexity.

In this paper, we present a probabilistic algorithm using a factor base and an auxiliary data structure to solve the principal ideal problem. Our algorithm is inspired by Buchmann's algorithm [**4**] for computing the class group of a number field. However, it is not easy to adapt this technique to quaternion algebras. Indeed, the set of right ideals of an order does not form a group under multiplication. In fact, for most pairs of ideals, multiplication is not well-defined. We are able to salvage the factor base technique in the case of indefinite quaternion algebras by algorithmically realizing the strong approximation property (Theorem 2.1). The main point is that if every ideal were two-sided, Buchmann's method would work unchanged. Our algorithm is divided in two parts. Because the algebra is indefinite, every ideal is equivalent to an "almost

---

2000 *Mathematics Subject Classification* 11Y40 (primary), 11R52, 11R65 (secondary).

This research was partially funded by ERC Starting Grant ANTICS 278537.

[†]Trying every linear combination of a basis until we find a generator.

two-sided" ideal: a local algorithm (Algorithm 3.9) makes this equivalence effective. The global algorithm (Algorithm 3.12) uses a factor base: by linear algebra it cancels out the valuations of the *norm* of the ideal and then corrects the ideal locally at every prime to make it two-sided. We implemented our algorithm as in Magma. It performs well in practice, compared to the built-in Magma function implementing Kirschmer and Voight's algorithm.

The paper is organized as follows. We first recall basic properties of quaternion algebras, Eichler's theorems and Bruhat-Tits trees in Section 2. We then proceed to algorithms in Section 3. In Section 3.1, we define local and global reduction structures and Algorithm 3.14, solving the principal ideal problem. In Section 3.2, Algorithm 3.19 constructs the needed local and global reduction structures: the first one uses units constructed from commutative suborders, and the second one is inspired by Buchmann's algorithm. In Section 3.3, we introduce a compact representation for quaternions to prevent coefficient explosion in the previous algorithms. Section 4 provides a complexity analysis of our algorithms: assuming suitable heuristics, we prove a subexponential running time. Section 5 presents examples.

2. *Background on quaternion algebras and Bruhat-Tits trees*

When G is a group and $S \subset G$ is a subset, we write $\langle S\rangle$ for the subgroup generated by S. When the group G acts on a set X, we say that S acts transitively on X if $\langle S\rangle$ does.

2.1. *Quaternion algebras*

Good references for this section and the next one are [**12**], [**17**] and [**19**]. Let K be a number field with ring of integers $\mathbb{Z}_K$ and discriminant d_K. We write $N : K \to \mathbb{Q}$ for the norm. Let $\mathfrak{p}$ be a prime of $\mathbb{Z}_K$. We write $K_\mathfrak{p}$ for the $\mathfrak{p}$-adic completion of K, we let $v_\mathfrak{p}$ be the $\mathfrak{p}$-adic valuation and we write $\kappa_\mathfrak{p}$ for the residue field $\mathbb{Z}_K/\mathfrak{p}$. When S is a set of primes of $\mathbb{Z}_K$, we write $\mathbb{Z}_{K,S}$ the ring of S-integers in K.

Let A be a quaternion algebra over K with reduced norm nrd. Let v be a place of K. The place v is *split* or *ramified* according to whether $A \otimes_K K_v \cong \mathcal{M}_2(K_v)$ or not. The *reduced discriminant* δ_A of A is the product of the ramified primes and its *absolute discriminant* is the integer $\Delta_A = d_K^4 N(\delta_A)^2$. Let $\mathcal{O}$ be a maximal order in A. We write $\mathcal{O}^1$ for the group $\{x \in \mathcal{O} \mid \mathrm{nrd}(x) = 1\}$. A *lattice* $I \subset A$ is a finitely generated $\mathbb{Z}_K$-submodule such that $KI = A$. The *right order* $\mathcal{O}_r(I)$ of I is the set $\{x \in A \mid Ix \subset I\}$ and the *left order* $\mathcal{O}_l(I)$ is defined analogously. A *right $\mathcal{O}$-ideal* is a lattice I such that $\mathcal{O}_r(I) = \mathcal{O}$. The ideal I is *integral* if $I \subset \mathcal{O}$ and I is *two-sided* if $\mathcal{O}_l(I) = \mathcal{O}_r(I)$. The inverse I^{-1} of I is $\{x \in A \mid xI \subset \mathcal{O}\}$. If I, J are lattices such that $\mathcal{O}_r(I) = \mathcal{O}_l(J)$, we define their product IJ to be the lattice generated by the set $\{xy \ : \ x \in I,\ y \in J\}$. If I is an $\mathcal{O}$-ideal we have $II^{-1} = \mathcal{O}_l(I)$ and $I^{-1}I = \mathcal{O}_r(I)$. The *reduced norm* $\mathrm{nrd}(I)$ of an $\mathcal{O}$-ideal I is the $\mathbb{Z}_K$-module generated by the reduced norms of elements in I. The reduced norm of ideals is multiplicative. For a right $\mathcal{O}$-ideal I we define $\mathcal{N}(I) = N(\mathrm{nrd}(I))$ and for an element $x \in A^\times$ we set $\mathcal{N}(x) = \mathcal{N}(x\mathcal{O})$. Let $\mathfrak{p}$ be a prime of $\mathbb{Z}_K$. There exists a unique two-sided $\mathcal{O}$-ideal $\mathfrak{P}$ such that every two-sided $\mathcal{O}$-ideal having reduced norm a power of $\mathfrak{p}$ is a power of $\mathfrak{P}$. We have $\mathfrak{P} = \mathfrak{p}\mathcal{O}$ if $\mathfrak{p}$ splits in A and $\mathfrak{P}^2 = \mathfrak{p}\mathcal{O}$ if $\mathfrak{p}$ ramifies in A: such an ideal $\mathfrak{P}$ is called a prime of $\mathcal{O}$, and every two-sided $\mathcal{O}$-ideal is a product of primes of $\mathcal{O}$. The set of right $\mathcal{O}$-ideals is equipped with an action of the group of two-sided $\mathcal{O}$-ideals by multiplication on the right and an action of the group $A^\times$ by multiplication on the left. Two right $\mathcal{O}$-ideals I, J are *equivalent* if there exists $x \in A^\times$ such that $xI = J$, that is if they lie in the same orbit modulo $A^\times$. The set of equivalence classes of right $\mathcal{O}$-ideals is written $\mathrm{Cl}(\mathcal{O})$. An ideal is *principal* if it is equivalent to the unit ideal $\mathcal{O}$. If S is a set of primes of $\mathbb{Z}_K$, the *S-order* associated with $\mathcal{O}$ is the ring $\mathcal{O}_S = \mathbb{Z}_{K,S}\mathcal{O}$ and the group of *S-units* (relative to $\mathcal{O}$) in A is $\mathcal{O}_S^\times$.

2.2. *Eichler's condition and theorems*

A quaternion algebra A satisfies the *Eichler condition* or is *indefinite* if there exists an infinite place of the base field K at which A is split. Indefinite algebras satisfy the following properties.

THEOREM 2.1 (Consequence of strong approximation). *Let $\mathcal{O}$ be a maximal order in a quaternion algebra A over a number field K, satisfying the Eichler condition. Let $\mathfrak{p}$ be a prime of $\mathbb{Z}_K$ that splits in A and k a positive integer. Then the map*

$$\mathcal{O}^1 \longrightarrow \mathrm{SL}_2(\mathbb{Z}_K/\mathfrak{p}^k)$$

is surjective.

THEOREM 2.2 (Integral version of Eichler's norm theorem). *Let $\mathcal{O}$ be a maximal order in a quaternion algebra A over a number field K satisfying the Eichler condition. Let S be a finite set of primes of $\mathbb{Z}_K$. Let $\mathbb{Z}_{K,S,A}^\times$ be the set of S-units that are positive at every real place of K that ramifies in A. Then the reduced norm*

$$\mathrm{nrd} : \mathcal{O}_S^\times \longrightarrow \mathbb{Z}_{K,S,A}^\times$$

is surjective.

THEOREM 2.3 (Eichler). *Let $\mathcal{O}$ be a maximal order in a quaternion algebra A over a number field K satisfying the Eichler condition. Let $\mathrm{Cl}_A(K)$ be the ray class group with modulus the product of the real places of K that ramify in A. Then the reduced norm induces a bijection*

$$\mathrm{Cl}(\mathcal{O}) \xrightarrow{\sim} \mathrm{Cl}_A(K).$$

In other words, two right $\mathcal{O}$-ideals are equivalent if and only if the classes of their norm in $\mathrm{Cl}_A(K)$ are equal. Note that since $\mathrm{Cl}(\mathcal{O})$ is not a group, this map is only a bijection of *sets*.

2.3. *The Bruhat-Tits tree*

The standard reference for this section is [**16**]. Let K be a field with a discrete valuation v. Let R be its valuation ring, π a uniformizer and $\kappa = R/\pi R$ the residue field. An *R-lattice in K^2* is an R-submodule of rank 2 in K^2. We define the *Bruhat-Tits tree* $\mathcal{T}$, which we write $\mathcal{T}_\mathfrak{p}$ when K is the $\mathfrak{p}$-adic completion of a number field. The set of vertices of $\mathcal{T}$ is the set of homothety classes of R-lattices in K^2. Let L, L' be two such R-lattices. There exists an ordered R-basis (e_1, e_2) of L and integers a, b such that $(\pi^a e_1, \pi^b e_2)$ is an R-basis of L'. The integer $|a-b|$ depends only on the homothety classes of L, L' and is called their *distance*. By definition, there is an edge in the tree $\mathcal{T}$ between every pair of vertices at distance 1. The graph $\mathcal{T}$ is an infinite tree. If P, Q are two vertices, the unique path of minimum length between P and Q is called the *segment PQ* and the distance $d(P,Q)$ equals the length of the segment PQ. The set of vertices at distance 1 from a given vertex is in natural bijection with $\mathbb{P}^1(\kappa)$. The group $\mathrm{GL}_2(K)$ acts on the tree and preserves the distance, and this action factors through $\mathrm{PGL}_2(K)$ and is transitive on the set of vertices. The stabilizer of the vertex P_0 corresponding to the R-lattice R^2 is $K^\times \mathrm{GL}_2(R)$ and the stabilizer of any vertex is a conjugate of this group. The group $\mathrm{SL}_2(R)$ acts transitively on the set of vertices at a fixed distance from P_0. For every $g \in \mathcal{M}_2(R) \setminus \pi\mathcal{M}_2(R)$, the Smith normal form shows that $d(g \cdot P_0, P_0) = v(\det(g))$. The tree is illustrated in Figure 1 where we label some vertices P with a matrix g such that $P = g \cdot P_0$.

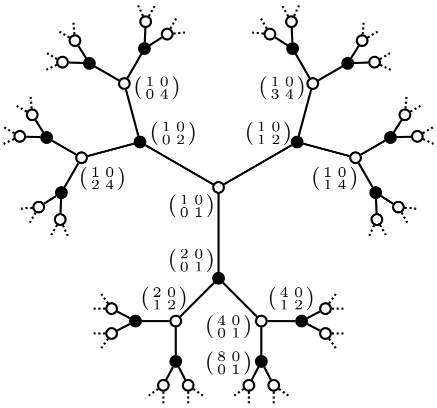

FIGURE 1. *The Bruhat-Tits tree for $K = \mathbb{Q}_2$*

THEOREM 2.4. *Let P, Q be two vertices of the tree $\mathcal{T}$ with $d(P,Q) = 1$. Then the action of the group $G = \mathrm{SL}_2(K)$ on the vertices of $\mathcal{T}$ has exactly two orbits $G \cdot P$ and $G \cdot Q$.*

The connection between the Bruhat-Tits tree and ideals is the following: a right $\mathcal{M}_2(R)$-ideal is always principal, generated by an element of $\mathrm{GL}_2(K)$. Such an ideal is two-sided if and only if it is generated by an element of $K^\times \mathrm{GL}_2(R)$. So there is a $\mathrm{GL}_2(K)$-equivariant bijection between set of the vertices of the Bruhat-Tits tree and the quotient of the set of right $\mathcal{M}_2(R)$-ideals modulo the action of the group of two-sided $\mathcal{M}_2(R)$-ideals.

3. *Algorithms*

We want to adapt the classical subexponential algorithms for computing the class group of a number field due to Hafner and McCurley [**7**] in the quadratic case and Buchmann [**4**] in the general case to indefinite quaternion algebras by using a factor base: a fixed finite set of primes of $\mathbb{Z}_K$. To simplify the notations, we set $\Delta = \Delta_A$.

DEFINITION 3.1. The *factor base* for A is a finite set $\mathcal{B}$ of primes of $\mathbb{Z}_K$ that generates the group $\mathrm{Cl}_A(K)$.

We say that a fractional ideal $\mathfrak{a}$ of K is *$\mathcal{B}$-smooth* or simply *smooth* if it is a product of the primes in $\mathcal{B}$. Let I be a right $\mathcal{O}$-ideal I. When I is integral, we say that I is *smooth* if its reduced norm is. When I is arbitrary, it is *smooth* if it can be written $I = J\mathfrak{a}$ with $\mathfrak{a}$ a smooth fractional ideal of K and J an integral smooth right $\mathcal{O}$-ideal. Equivalently, the ideal I is smooth if and only if $I_\mathfrak{p} = \mathcal{O}_\mathfrak{p}$ for all $\mathfrak{p} \notin \mathcal{B}$. An element $x \in A^\times$ is *smooth* if the ideal $x\mathcal{O}$ is smooth, or equivalently if $x \in \mathcal{O}_S^\times$ with $S = \mathcal{B}$.

We equip $\mathcal{M}_2(\mathbb{R})$ and $\mathcal{M}_2(\mathbb{C})$ with the usual positive definite quadratic form Q given by the sum of the squares of the absolute values of the coefficients, and we equip the Hamiltonian quaternion algebra $\mathbb{H}$ with the positive definite quadratic form $Q = \mathrm{nrd}$. For each infinite place of K represented by a complex embedding σ, we fix an isomorphism $\sigma' : A \otimes_K K_\sigma \cong M$ extending σ, where M is one of $\mathcal{M}_2(\mathbb{R})$, $\mathcal{M}_2(\mathbb{C})$ or $\mathbb{H}$. This defines a positive definite quadratic form $T_2 : A \otimes_\mathbb{Q} \mathbb{R} \to \mathbb{R}$ by setting $T_2(x) = \sum_\sigma [K_\sigma : \mathbb{R}] \cdot Q(\sigma'(x))$ for all $x \in A \otimes_\mathbb{Q} \mathbb{R}$, giving covolume $\Delta^{1/2}$ to the lattice $\mathcal{O}$. We represent a lattice in A by a $\mathbb{Z}_K$-pseudobasis (see [**12**]). When L is a lattice in A, we can enumerate its elements by increasing value of T_2 with the Kannan–Fincke–Pohst algorithm [**9**, **10**]. We represent this enumeration with a routine `NextElement` that outputs a new element of L every time we call `NextElement`(L), ordering them by increasing value of T_2.

3.1. *The reduction algorithms*

In this section, we describe the reduction structures and the corresponding reduction algorithms. We start with the local reduction, which is an effective version of the fact that every integral right $\mathcal{O}$-ideal of norm $\mathfrak{p}^2$ is equivalent to the two-sided ideal $\mathfrak{p}\mathcal{O}$ (Theorem 2.3). We perform this reduction by making algorithmic the reduction theory of $\mathrm{SL}_2(K_\mathfrak{p})$ on the Bruhat-Tits tree $\mathcal{T}_\mathfrak{p}$ (Section 2.3). The point is that this reduction needs only a small number of units: this leads to the definition of the $\mathfrak{p}$-reduction structure.

Definition 3.2. Let $\mathfrak{p}$ be a prime that splits in A, let $\mathcal{O}_0 = \mathcal{O}$ and let P_0 be the fixed point of $\mathcal{O}_0^\times$ in the Bruhat-Tits tree $\mathcal{T}_\mathfrak{p}$. A $\mathfrak{p}$-*reduction structure* is given by the following data:
(i) the left order $\mathcal{O}_1$ of an integral right $\mathcal{O}$-ideal of norm $\mathfrak{p}$, and the fixed point P_1 of $\mathcal{O}_1^\times$ in $\mathcal{T}_\mathfrak{p}$;
(ii) for each $b \in \{0,1\}$ and for each $P \in \mathcal{T}_\mathfrak{p}$ at distance 1 from P_b, an element $g \in \mathcal{O}_b^\times$ such that $g \cdot P = P_{1-b}$.

Such a structure exists by strong approximation (Theorem 2.1). Note that if I is an integral right $\mathcal{O}$-ideal of norm $\mathfrak{p}$ such that $\mathcal{O}_1 = \mathcal{O}_l(I)$, we have $I_\mathfrak{p} = x\mathcal{O}_\mathfrak{p}$ for some $x \in A^\times$, and $P_1 = x \cdot P_0$. We represent the points at distance 1 from P_b by elements of $\mathbb{P}^1(\kappa_\mathfrak{p})$ and we compute the action on these points via explicit splitting maps $\iota_b : \mathcal{O}_b/\mathfrak{p}\mathcal{O}_b \to \mathcal{M}_2(\kappa_\mathfrak{p})$.

This structure provides everything we need to perform reduction in the Bruhat-Tits tree. The following algorithm corresponds to the standard reduction procedure (Theorem 2.4), which is illustrated in Figure 2. The idea is to use successive "rotations" (elements in $\mathrm{SL}_2(K_\mathfrak{p})$ having a fixed point in the tree) around the adjacent vertices P_0 and P_1 to send an arbitrary vertex to one of the vertices P_b: every rotation around a vertex decreases the distance to the other one.

To realize this procedure, we need to perform the following subtask: given a right $\mathcal{O}$-ideal I, find $x \in I$ such that $I_\mathfrak{p} = x\mathcal{O}_\mathfrak{p}$. A simple idea is to let $e = v_\mathfrak{p}(\mathrm{nrd}(J))$ and to draw elements $x \in J/\mathfrak{p}\mathcal{O}$ uniformly at random until $v_\mathfrak{p}(\mathrm{nrd}(x)) = e$. To obtain a deterministic algorithm, we can adapt Euclid's algorithm in the matrix ring $\mathcal{M}_2(\mathcal{R})$ with $\mathcal{R} = \mathbb{Z}_K/\mathfrak{p}^{e+1}$. This is done in [**15**], except that the base ring $\mathcal{R}$ is assumed to be a domain. We adapt the argument to our case. First note that we have a well-defined $\mathfrak{p}$-adic valuation $v = v_\mathfrak{p}$ in the ring $\mathcal{R}$. Let $a, b \in \mathcal{R}$. We have $v(ab) \le v(a) + v(b)$ whenever $ab \neq 0$, and $a \mid b$ if and only if $v(b) \ge v(a)$. If $a \neq 0$, there is a Euclidean division taking the following simple form: if $a \mid b$ then $b = a \cdot (b/a) + 0$, and otherwise $b = a \cdot 0 + b$. In every case we have written $b = aq + r$ with $r = 0$ or $v(r) < v(a)$. Adapting this in the matrix ring leads to the following Euclidean division algorithm, where for convenience we write $w = v \circ \det$. The idea is to work with A in Smith normal form, and if A is a diagonal matrix, dividing by A is almost the same as dividing by the diagonal coefficients. The difference is that we have to ensure that $\det(R) \neq 0$ unless $R = 0$.

Subalgorithm 3.3 `DivideMatrix`.
Input: two matrices $A, B \in \mathcal{M}_2(\mathcal{R})$ with $\det A \neq 0$, where $\mathcal{R} = \mathbb{Z}_F/\mathfrak{p}^i$.
Output: two matrices $Q, R \in \mathcal{M}_2(\mathcal{R})$ such that $B = AQ + R$, and ($R = 0$ or $w(R) < w(A)$).
1: let $A' = UAV$ be the Smith form of A with $U, V \in \mathrm{SL}_2(\mathcal{R})$ and $A = \left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right)$
2: $B' \leftarrow UB$
3: let $B'' = B'W$ be the Hermite form of B' with $W \in \mathrm{SL}_2(\mathcal{R})$ and $B'' = \left(\begin{smallmatrix} c & 0 \\ e & f \end{smallmatrix}\right)$
4: **if** $a \mid c$ **then**

5: **if** $b \mid f$ **then**
6: **if** $b \mid e$ **then**
7: $Q \leftarrow \begin{pmatrix} c/a & 0 \\ e/b & f/b \end{pmatrix}$, $R \leftarrow \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$
8: **else**
9: $Q \leftarrow \begin{pmatrix} c/a & 1 \\ 0 & f/b \end{pmatrix}$, $R \leftarrow \begin{pmatrix} 0 & -a \\ e & 0 \end{pmatrix}$
10: **end if**
11: **else**
12: $Q \leftarrow \begin{pmatrix} c/a-1 & 0 \\ 0 & 0 \end{pmatrix}$, $R \leftarrow \begin{pmatrix} a & 0 \\ e & f \end{pmatrix}$
13: **end if**
14: **else**
15: **if** $b \mid f$ **then**
16: $Q \leftarrow \begin{pmatrix} 0 & 0 \\ 0 & f/b-1 \end{pmatrix}$, $R \leftarrow \begin{pmatrix} c & 0 \\ e & b \end{pmatrix}$
17: **else**
18: $Q \leftarrow \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$, $R \leftarrow \begin{pmatrix} c & 0 \\ e & b \end{pmatrix}$
19: **end if**
20: **end if**
21: **return** VQW^{-1}, $U^{-1}RW^{-1}$

PROPOSITION 3.4. *Subalgorithm 3.3 is correct.*

Proof. By case-by-case analysis, we have $B'' = A'Q + R$, and either $R = 0$ (Step 7) or $\det(R) \neq 0$ and $w(R) < w(A')$. Let $Q' = VQW^{-1}$ and $R' = U^{-1}RW^{-1}$ be the matrices returned by the algorithm. We have $B = U^{-1}B' = U^{-1}B''W^{-1} = U^{-1}(A'Q + R)W^{-1} = U^{-1}A'V^{-1}Q' + R' = AQ' + R'$. Since U, V and W have determinant 1, we have $R = 0$ if and only if $R' = 0$, and $w(R') = w(R) < w(A') = w(A)$, proving the correctness of the algorithm. □

SUBALGORITHM 3.5 `GCDMatrix`.
Input: two matrices $A, B \in \mathcal{M}_2(\mathcal{R})$ with $\det A \neq 0$, where $\mathcal{R} = \mathbb{Z}_F/\mathfrak{p}^i$.
Output: a matrix D such that $A\mathcal{M}_2(\mathcal{R}) + B\mathcal{M}_2(\mathcal{R}) = D\mathcal{M}_2(\mathcal{R})$.
1: $Q, R \leftarrow$ `DivideMatrix`(A, B)
2: **if** $R = 0$ **then**
3: **return** A
4: **else**
5: **return** `GCDMatrix`(R, A)
6: **end if**

PROPOSITION 3.6. *Subalgorithm 3.5 is correct.*

Proof. In Step 5 we have $\det(R) \neq 0$ by the properties of Subalgorithm 3.3, so the recursive call to `GCDMatrix` is valid. The rest of the proof is the same as with the usual Euclidean algorithm. □

SUBALGORITHM 3.7 `LocalGenerator`.
Input: an integral right $\mathcal{O}$-ideal I and a prime $\mathfrak{p}$, for some maximal order $\mathcal{O}$.
Output: an element $x \in I$ such that $I_\mathfrak{p} = x\mathcal{O}_\mathfrak{p}$.
1: $b_1, \ldots, b_n \leftarrow$ an *LLL*-reduced $\mathbb{Z}$-basis of I
2: $e \leftarrow v_\mathfrak{p}(\mathrm{nrd}(b_1))$

3: $\mathcal{R} \leftarrow \mathbb{Z}_K/\mathfrak{p}^{e+1}$
4: $B_1, \ldots, B_n \leftarrow$ images of $b_1, \ldots, b_n$ in $\mathcal{M}_2(\mathcal{R})$
5: $D \leftarrow B_1$
6: **for** $i = 1$ to n **do**
7: $\quad D \leftarrow$ `GCDMatrix`(D, B_i)
8: **end for**
9: let $\mu_1, \ldots, \mu_n \in \mathbb{Z}$ be such that $\sum \mu_i B_i = D$
10: **return** $\sum \mu_i b_i$

Proposition 3.8. *Subalgorithm 3.7 is correct.*

Proof. By definition of the norm of an ideal, we have $v_\mathfrak{p}(\mathrm{nrd}(I)) \leq v_\mathfrak{p}(\mathrm{nrd}(b_1)) = e$. Because of the choice of $\mathcal{R}$, at Step 5 we have $\det(D) \neq 0$, so the calls to `GCDMatrix` are valid. By the properties of Subalgorithm 3.5, at Step 9 we have $D\mathcal{M}_2(\mathcal{R}) = B_1\mathcal{M}_2(\mathcal{R}) + \cdots + B_1\mathcal{M}_2(\mathcal{R})$ so the integers $\mu_1, \ldots, \mu_n$ exist. Now let $x = \sum \mu_i b_i \in I$ be the output of the algorithm, so that $v_\mathfrak{p}(\mathrm{nrd}(x)) \leq e$, hence $\mathfrak{p}^{e+1}\mathcal{O}_\mathfrak{p} \subset x\mathcal{O}_\mathfrak{p}$. Let $y \in I_\mathfrak{p}$. By reduction modulo $\mathfrak{p}^{e+1}$ there exists $a \in \mathcal{O}_\mathfrak{p}$ and $b \in \mathfrak{p}^{e+1}\mathcal{O}$ such that $y = xa + b \in x\mathcal{O}_\mathfrak{p}$, proving the result. $\square$

Now we can present the local reduction algorithm.

Subalgorithm 3.9 `PReduce`.
Input: an integral right $\mathcal{O}$-ideal I, a prime $\mathfrak{p}$ and a $\mathfrak{p}$-reduction structure.
Output: an integer r, an element $c \in A^\times$ and an integral $\mathcal{O}$-ideal J such that $cI = J\mathfrak{p}^r$ and $v_\mathfrak{p}(\mathrm{nrd}(J)) \in \{0, 1\}$.
1: $r \leftarrow$ largest integer such that $I \subset \mathcal{O}\mathfrak{p}^r$, $J \leftarrow I\mathfrak{p}^{-r}$
2: $k \leftarrow v_\mathfrak{p}(\mathrm{nrd}(J))$
3: $c \leftarrow 1$, $b \leftarrow 0$
4: $x \leftarrow$ `LocalGenerator`$(I, \mathfrak{p})$
5: $Q \leftarrow x \cdot P_0$
6: **repeat**
7: $\quad P \leftarrow$ point at distance 1 from P_b in the segment P_bQ
8: $\quad (c, J, Q) \leftarrow g \cdot (c, J, Q)$ where $g \in \mathcal{O}_b^\times$ is such that $g \cdot P = P_{1-b}$
9: $\quad$ **if** $b = 1$ **then** $J \leftarrow J\mathfrak{p}^{-1}$, $r \leftarrow r + 1$, $k \leftarrow k - 2$ **end if**
10: $\quad b \leftarrow 1 - b$
11: **until** $k < 2$
12: **return** J, c, r

In Step 1, we have $2r = v_\mathfrak{p}(\mathrm{nrd}(\mathfrak{I}))$ where $\mathfrak{I}$ is the two-sided $\mathcal{O}$-ideal generated by I. We can compute $\mathfrak{I}$ as follows: if $w_1, \ldots, w_n$ is a $\mathbb{Z}$-basis of $\mathcal{O}$ and $b_1, \ldots, b_n$ is a $\mathbb{Z}$-basis of I, then $\mathfrak{I} = \sum_{i,j} \mathbb{Z}w_ib_j$.

Proposition 3.10. *Subalgorithm 3.9 is correct.*

Proof. Since k decreases by 2 every two iterations and is positive by the loop condition, the algorithm terminates. We now prove that the output is correct.

First, the distance $d(P_b, Q)$ decreases by 1 during each execution of the loop: before Step 8, we have $d(P_{1-b}, gQ) = d(gP, gQ) = d(P, Q) = d(P_b, Q) - 1$ since P is at distance 1 from P_b on the segment P_bQ. We claim that before or after a complete execution of the loop, we

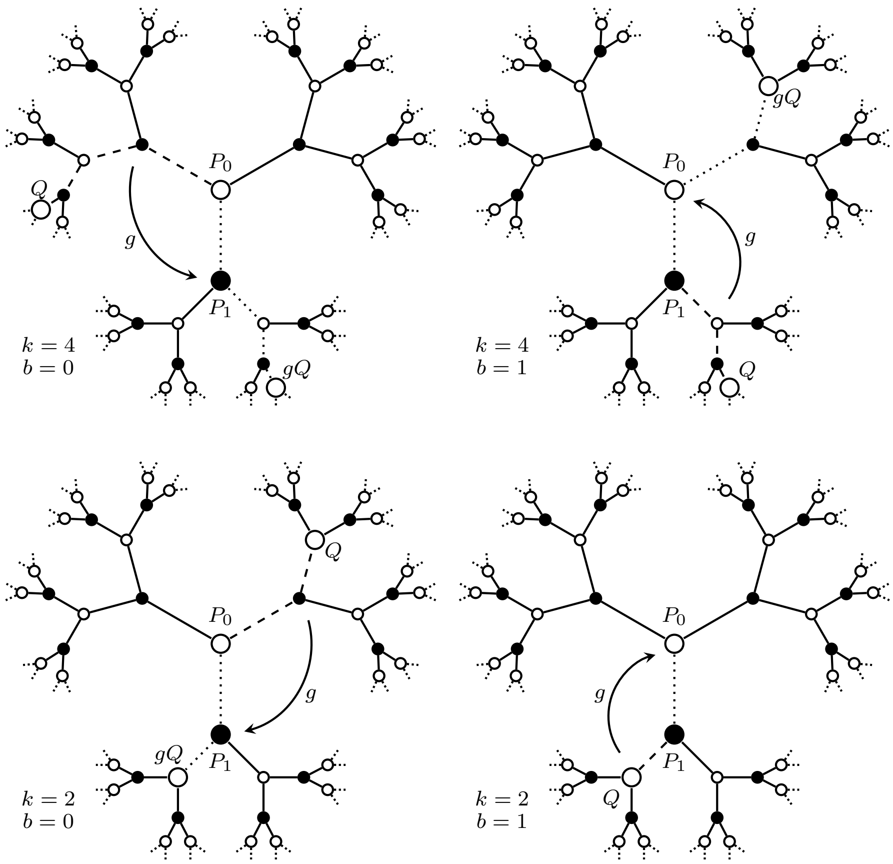

FIGURE 2. *PReduce (Subalgorithm 3.9)*

have $d(P_0, Q) = k$ (see Figure 2). The claim is true before the first iteration: we have $x \in \mathcal{O} \setminus \mathfrak{p}\mathcal{O}$ so $d(P_0, Q) = d(P_0, xP_0) = v_{\mathfrak{p}}(\mathrm{nrd}(x)) = v_{\mathfrak{p}}(\mathrm{nrd}(J)) = k$. We only need to prove that the equality $d(P_0, Q) = k$ is preserved when $b = 1$. In that case, P_1 is in the segment P_0Q because of the previous iteration, so that $d(P_0, Q) = 1 + d(P_1, Q) = k$.

We now prove that before or after a complete execution of the loop, the $\mathcal{O}$-ideal J is integral and $v_{\mathfrak{p}}(\mathrm{nrd}(J)) = k$. This property clearly holds before the first iteration. Before Step 9, after two iterations $b = 0$ and $b = 1$, J and Q have been multiplied by an element h such that $v_{\mathfrak{p}}(\mathrm{nrd}(h)) = 0$ and $d(P_0, hQ) = d(P_0, Q) - 2$, so hJ is divisible by $\mathfrak{p}$. Step 9 hence preserves integrality and updates k according to the valuation of $\mathrm{nrd}(J)$.

We now prove the proposition. The element c is a product of elements of $\mathcal{O}_0^\times$ and $\mathcal{O}_1^\times$ so c is a $\mathfrak{p}$-unit with $\mathrm{nrd}(c) \in \mathbb{Z}_K^\times$. We have just proved that J is integral, and by Step 9 the value of r is such that $cI = J\mathfrak{p}^r$. Because of the loop condition, after the algorithm terminates we have $v_{\mathfrak{p}}(\mathrm{nrd}(J)) = k \in \{0, 1\}$. □

We now explain how to perform global reduction. We use linear algebra to control the valuations of a smooth ideal and then perform local reduction at every prime to get an "almost two-sided ideal". The first step is similar to its commutative analogue: we need sufficiently many "relations" (smooth elements in $A^\times$) so that the quotient of the factor base by the norms of the relations is the ray class group $\mathrm{Cl}_A(K)$. This leads to the definition of a G-reduction structure.

Definition 3.11. A *G-reduction structure* is given by the following data:

(i) a $\mathfrak{p}$-reduction structure for each $\mathfrak{p} \in \mathcal{B}$ that splits in A;

(ii) a finite set of elements $X \subset \mathcal{O} \cap A^\times$ and a map $\phi : \mathrm{Cl}_A(K) \to \langle \mathcal{B} \rangle$ that is a lift of an isomorphism $\mathrm{Cl}_A(K) \xrightarrow{\sim} \langle \mathcal{B} \rangle / \langle \mathrm{nrd}(X) \rangle$ and such that $\phi(1) = \mathbb{Z}_K$.

The following algorithm performs global reduction. In order to avoid explosion of the size of the ideal in the local reduction, we extract the two-sided part, allowing us to reduce all exponents modulo 2. The remaining part stays small and gets $\mathfrak{p}$-reduced, while the two-sided part is only multiplied by powers of primes.

Subalgorithm 3.12 `GReduce`.

Input: a smooth integral right $\mathcal{O}$-ideal I and a G-reduction structure.

Output: an integral ideal J, an element $c \in A^\times$ and a two-sided ideal $\mathfrak{I}$ such that $cI = J\mathfrak{I}$ and I is principal if and only if $J = \mathcal{O}$ and $\mathfrak{I} = \mathcal{O}$.

1: $\mathfrak{a} \leftarrow \mathrm{nrd}(I)$
2: $\mathfrak{b} \leftarrow \phi(\mathfrak{a})$ where $\mathfrak{a}$ is seen as an element of $\mathrm{Cl}_A(K)$
3: let $e \in \mathbb{Z}^X$ be such that $\mathrm{nrd}(y)\mathfrak{a} = \mathfrak{b}$ where $y = \prod_{x \in X} x^{e_x}$.
4: $c \leftarrow \prod_{x \in X} x^{e_x \bmod 2}$, $f \leftarrow \prod_{x \in X} \mathrm{nrd}(x)^{\lfloor e_x/2 \rfloor}$ {Extract two-sided part}
5: $J \leftarrow cI$
6: $\mathfrak{I} \leftarrow$ two-sided ideal generated by J
7: $J \leftarrow J\mathfrak{I}^{-1}$ {Extract two-sided part}
8: $\mathfrak{I} \leftarrow f\mathfrak{I}$
9: **for** $\mathfrak{p} \in \mathcal{B}$ dividing $\mathrm{nrd}(J)$ and splitting in A **do**
10: $\quad J, c', r \leftarrow$ `PReduce`$(J, \mathfrak{p})$
11: $\quad c \leftarrow c'c$, $\mathfrak{I} \leftarrow \mathfrak{p}^r \mathfrak{I}$
12: **end for**
13: **return** $J, cf, \mathfrak{I}$

Proposition 3.13. *Subalgorithm 3.12 is correct.*

Proof. Since Step 7 and `PReduce` preserve integrality (Proposition 3.10), the output J is integral. The relation $cI = J\mathfrak{I}$ is clear by tracking the multiplications. If the output is such that $J = \mathcal{O}$ and $\mathfrak{I} = \mathcal{O}$, then $I = c^{-1}\mathcal{O}$ is principal. Conversely, if I is principal, then $\mathfrak{a} = \mathrm{nrd}(I)$ is trivial in the class group $\mathrm{Cl}_A(K)$ so $\mathfrak{b} = \phi(\mathrm{cl}(\mathfrak{a})) = \mathbb{Z}_F$. After Step 5, we have $\mathrm{nrd}(J) = f^{-2}\mathbb{Z}_F$. After Step 7, we have multiplied J by a two-sided ideal, so $v_\mathfrak{p}(\mathrm{nrd}(J))$ is even for all primes $\mathfrak{p}$ splitting in A. Since J is not divisible by a two-sided ideal, $\mathrm{nrd}(J)$ is not divisible by primes that ramify in A. We obtain $J = \mathcal{O}$ at the end of the loop by the properties of `PReduce` so $\mathrm{nrd}(\mathfrak{I}) = \mathbb{Z}_F$. Since $\mathfrak{I}$ is two-sided, it is entirely determined by its norm so $\mathfrak{I} = \mathcal{O}$. □

Finally, we reduce the general case to the smooth case by the noncommutative analogue of standard randomizing techniques. We generate a random smooth $\mathcal{O}$-ideal by the following procedure, to which we refer as `RandomLeftIdeal`$(\mathcal{O})$. For each $\mathfrak{p} \in \mathcal{B}$, pick a nonnegative integer k. Let $\iota : \mathcal{O} \to \mathcal{M}_2(\mathbb{Z}_F/\mathfrak{p}^k)$ be a splitting map. Let $M \in \mathcal{M}_2(\mathbb{Z}_F/\mathfrak{p}^k)$ be a random upper-triangular matrix with zero determinant and compute $R_\mathfrak{p} = \mathcal{O}\iota^{-1}(M) + \mathfrak{p}^k\mathcal{O}$. Finally, return $\bigcap_\mathfrak{p} R_\mathfrak{p}$. Choose the exponents k such that $\mathcal{N}(R) \approx \Delta$.

It not clear at the moment what the best distribution for the exponents is. A simpler idea would be to use random products $\prod_\mathfrak{p} \mathfrak{p}^{k_\mathfrak{p}}$. In our experience, this leads to poorly randomized ideals. This is clear in the case $K = \mathbb{Q}$: the randomized ideals are simply integer multiples of $\mathcal{O}$.

Algorithm 3.14 `IsPrincipal`.

Input: an integral right $\mathcal{O}$-ideal I and a G-reduction structure.

Output: an integral ideal J, an element $c \in A^\times$ and a two-sided ideal $\mathfrak{I}$ such that $cfI = J\mathfrak{I}$ and I is principal if and only if $J = \mathcal{O}$ and $\mathfrak{I} = \mathcal{O}$.

$R \leftarrow$ `RandomLeftIdeal`$(\mathcal{O})$
$x \leftarrow$ `NextElement`$(I^{-1} \cap R)$
if xI is not smooth **then** **return FAIL** **end if**
$J, c, \mathfrak{I} \leftarrow$ `GReduce`(xI)
return $J, cx, \mathfrak{I}$

By Proposition 3.13, if Algorithm 3.14 does not return FAIL, its output is correct. In practice, we repeat Algorithm 3.14 until it returns the result.

3.2. *Building the reduction structures*

Now we explain how to build the previous reduction structures. The local reduction structure needs units in $\mathcal{O}$. In general it is difficult to compute the whole unit group $\mathcal{O}^\times$: for instance in the Fuchsian case, the minimal number of generators is at least $\Delta^{3/8+o(1)}$ (this follows from the theory of signatures of Fuchsian groups [**11**, Section 4.3] and a volume formula [**14**, Theorem 11.1.1]), which makes it hopeless to find a subexponential method. However, we can find some units in $\mathcal{O}$ by considering commutative suborders and computing generators of their unit group with Buchmann's algorithm. Heuristically, these units are sufficiently random for our purpose. This strategy is implemented by the following algorithms.

Subalgorithm 3.15 `P1Search`.

Input: a maximal order $\mathcal{O}$ and a prime $\mathfrak{p}$.

Output: a set of elements $X \subset \mathcal{O}^\times$ acting transitively on $\mathbb{P}^1(\kappa_\mathfrak{p})$.

1: $X \leftarrow \emptyset$
2: **repeat**
3: $x \leftarrow$ `NextElement`$(\mathcal{O})$
4: $L \leftarrow K(x)$
5: **if** L/K has positive relative unit rank **then**
6: $\mathcal{R} \leftarrow \mathbb{Z}_L \cap \mathcal{O}$
7: $X \leftarrow X \cup$ a set of generators of $\mathcal{R}^\times$
8: **end if**
9: **until** X acts transitively on $\mathbb{P}^1(\kappa_\mathfrak{p})$
10: **return** X

In Step 7, we can compute the unit group $\mathcal{R}^\times$ with the algorithms of Klüners and Pauli [**13**]. Note that we actually do not need the full group $\mathcal{R}^\times$: a subgroup of finite index is sufficient.

Proposition 3.16. *Subalgorithm 3.15 is correct.*

Proof. By strong approximation (Theorem 2.1), the group $\mathcal{O}^\times$ acts transitively on $\mathbb{P}^1(\kappa_\mathfrak{p})$. This group is finitely generated, so after finitely many iterations we will have enumerated a set of generators and the algorithm will terminate. By the loop condition, the output is correct. □

Subalgorithm 3.17 `PBuild`.

Input: a maximal order $\mathcal{O}$ and a prime $\mathfrak{p}$.

Output: a $\mathfrak{p}$-reduction structure.

1: $I \leftarrow$ an integral right $\mathcal{O}$-ideal of norm $\mathfrak{p}$
2: $\mathcal{O}_1 \leftarrow \mathcal{O}_l(I)$
3: $X \leftarrow \texttt{P1Search}(\mathcal{O}, \mathfrak{p})$
4: from X, for each P at distance 1 from P_0 compute an element $g \in \mathcal{O}^\times$ such that $g \cdot P = P_1$
5: $X \leftarrow \texttt{P1Search}(\mathcal{O}_1, \mathfrak{p})$
6: from X, for each P at distance 1 from P_1 compute an element $g \in \mathcal{O}_1^\times$ such that $g \cdot P = P_0$
7: **return** the $\mathfrak{p}$-reduction structure

Remark 3.18. Let $g, g' \in \mathcal{O}^\times$ be such that $g \cdot P_1 = g' \cdot P_1$ and let $h = g^{-1}g'$. Then $h \cdot P_1 = P_1$ so $h \in \mathcal{O}^\times \cap \mathcal{O}_1^\times$. This allows us to construct many elements in $\mathcal{O}_1^\times$ before Step 5. If we have sufficiently many such units, which often happens in practice, they will act transitively on the points $P \neq P_0$ at distance 1 from P_1. In this case, in Step 5 we will only need to find one element $g \in \mathcal{O}_1^\times$ such that $g \cdot P_0 \neq P_0$.

We build the global reduction structure in a way similar to the commutative case: we look for small relations in smooth ideals. In addition, we get a good starting point thanks to the inclusion $\mathbb{Z}_K \subset \mathcal{O}$: the units $\mathbb{Z}_{K,\mathcal{B}}^\times$ provide all the relations up to a 2-elementary Abelian group.

Algorithm 3.19 `GBuild`.

Input: a maximal order $\mathcal{O}$ and a factor base $\mathcal{B}$.
Output: a G-reduction structure.

1: **for** $\mathfrak{p} \in \mathcal{B}$ that splits in A **do**
2: $\quad$ $\texttt{PBuild}(\mathcal{O}, \mathfrak{p})$
3: **end for**
4: $X \leftarrow$ integral generators of $\mathbb{Z}_{K,\mathcal{B}}^\times$
5: **for** $\mathfrak{p} \in \mathcal{B}$ **do**
6: $\quad$ $I \leftarrow$ integral $\mathcal{O}$-ideal of norm $\mathfrak{p}$
7: $\quad$ **repeat**
8: $\quad\quad$ $x \leftarrow \texttt{NextElement}(I)$
9: $\quad$ **until** x is smooth
10: $\quad$ $X \leftarrow X \cup \{x\}$
11: **end for**
12: **while** $\langle \mathcal{B} \rangle / \langle \mathrm{nrd}(X) \rangle \not\cong \mathrm{Cl}_A(K)$ **do**
13: $\quad$ $x \leftarrow \texttt{NextElement}(\mathcal{O})$
14: $\quad$ **if** x is smooth **then** $X \leftarrow X \cup \{x\}$ **end if**
15: **end while**
16: **return** the G-reduction structure

Remark 3.20. The various calls to `PBuild` in Step 2 are not completely independent: we can keep the elements in $\mathcal{O}^\times$ from one call for other ones.

Proposition 3.21. *Algorithm 3.19 is correct.*

Proof. Let I be the ideal in Step 6. There exists a smooth ideal J equivalent to I, let $x \in A^\times$ be such that $I = xJ$. Then $x\mathcal{O} = IJ^{-1}$, so $\mathrm{nrd}(J)x \in I\bar{J} \subset I$ is smooth. It will be enumerated at some point, so the loop starting at Step 7 terminates. Since $\mathcal{B}$ generates the class group $\mathrm{Cl}_A(K)$, by Eichler's theorem (Theorem 2.3) we have $\langle \mathcal{B} \rangle / \mathrm{nrd}(A^\times) \cong \mathrm{Cl}_A(K)$, so there exists a finite set of $\mathcal{B}$-smooth elements $X \subset \mathcal{O}$ such that $\langle \mathcal{B} \rangle / \langle \mathrm{nrd}(X) \rangle \cong \mathrm{Cl}_A(K)$. We will enumerate this set at some point, so the loop starting at Step 12 terminates. So Algorithm 3.19 terminates, and by Proposition 3.16 and Step 12 it returns a correct G-reduction structure. □

REMARK 3.22. We have restricted to maximal orders to simplify the exposition, but this restriction can be weakened as follows. Let $\mathcal{O}$ be an arbitrary order, and let S be the set of primes $\mathfrak{p}$ of $\mathbb{Z}_K$ such that $\mathcal{O}$ is not $\mathfrak{p}$-maximal. The set S is finite, so we can choose a factor basis disjoint from S. Then our algorithms work unchanged for right $\mathcal{O}$-ideals except one point: Theorem 2.3 characterizing principal ideals might no longer hold. If we restrict to Eichler orders, that is intersections of two maximal orders, Theorem 2.3 still holds. Otherwise we need to find the suitable class group and change Definition 3.11 (ii) accordingly.

3.3. *Compact representations*

In the previous algorithms, the cost of elementary operations is important. Representing units as linear combinations of a basis of $\mathcal{O}$ could be catastrophic: the classical example of real quadratic fields suggests that fundamental units in commutative orders, such as those produced by Subalgorithm 3.15, can have exponential size. This problem is classically circumvented by representing units in *compact form*: a product of small S-units with possibly large exponents. The problem is reduced to computing efficiently with those compact representations. A natural notion of compact representation in $\mathcal{O}$ would be to take ordered products of S-units in $\mathcal{O}$ but we do not know how to compute efficiently with such a general representation. Instead we use a more restricted notion: we group the units belonging to a common commutative suborder, in which we can compute efficiently. This leads to the following definition.

DEFINITION 3.23. A *compact representation* in $\mathcal{O}$ is:

(i) an element $x \in \mathcal{O}$, or

(ii) a product $y = \prod_{i=1}^{r} y_i^{e_i}$ where the exponents are signed integers, the elements y_i all lie in a single ring $\mathcal{R} \subset \mathcal{O}$ containing $\mathbb{Z}_K$ and such that $y \in \mathcal{R}^\times$, together with a $\mathbb{Z}$-basis of the integral closure of $\mathcal{R}$ and a factorization of its conductor, or

(iii) an ordered product of compact representations.

A product y as in (ii) will be called a *representation of type* (ii).

We describe the algorithms for representations of type (ii), and they naturally extend by multiplicativity to arbitrary compact representations. We first explain the algorithm for local evaluation of compact representations. Since the product represents a unit, we can replace it with a product of local units, avoiding loss of precision despite the large exponents.

SUBALGORITHM 3.24 `EvalCR`.

Input: a representation $y = \prod_{i=1}^{r} y_i^{e_i} \in \mathcal{R}$ of type (ii), an ideal $\mathfrak{a} \subset \mathbb{Z}_K$.
Output: an element $z \in \mathcal{O}$ such that $z = y \pmod{\mathfrak{a}\mathcal{O}}$.

$L \leftarrow$ field of fractions of $\mathcal{R}$
$\mathfrak{f} \leftarrow$ conductor of $\mathcal{R}$ inside $\mathbb{Z}_L$
$\prod_{j=1}^{k} \mathfrak{P}_j^{w_j} \leftarrow$ factorization of $\mathfrak{a}\mathfrak{f}\mathbb{Z}_L$
for $j = 1$ to k **do**
    $\phi \leftarrow$ reduction map onto $\mathbb{Z}_L/\mathfrak{P}_j^{w_j}$
    $\pi \leftarrow$ uniformizer in $\mathfrak{P}_j$, $v \leftarrow v_{\mathfrak{P}_j}$
    $z_j \leftarrow \prod_{i=1}^{r} \phi(y_i \pi^{-v(y_i)})^{e_i}$
end for
$z \leftarrow$ element in $\mathbb{Z}_L$ such that $z = z_j \pmod{\mathfrak{P}_j^{w_j}}$

return z

PROPOSITION 3.25. *Subalgorithm 3.24 is correct.*

Proof. First, we claim that for all $j \leq k$, we have $z_j = y \pmod{\mathfrak{P}_j^{w_j}}$. Since $\mathrm{nrd}(y) \in \mathbb{Z}_K^\times$, we have $\sum_{i=1}^r v(y_i)e_i = 0$ so $\prod_{i=1}^r (y_i \pi^{-v(y_i)})^{e_i} = \prod_{i=1}^r \pi^{-v(y_i)e_i} \prod_{i=1}^r y_i^{e_i} = y$. Since $y_i\pi^{-v(y_i)}$ is integral at $\mathfrak{P}_j$, we can apply ϕ to it and the claim follows. This implies that the output z of the algorithm satisfies $z = y \pmod{\mathfrak{a}\mathfrak{f}\mathbb{Z}_L}$. In particular, we have $z = y \pmod{\mathfrak{f}\mathbb{Z}_L}$ so $z - y \in \mathcal{R}$. Since $y \in \mathcal{R}$ we get $z \in \mathcal{R} \subset \mathcal{O}$. The relation $\mathfrak{a}\mathfrak{f}\mathbb{Z}_L \subset \mathfrak{a}\mathcal{R} \subset \mathfrak{a}\mathcal{O}$ shows that $z = y \pmod{\mathfrak{a}\mathcal{O}}$. □

We can now explain how to multiply an ideal by a compact representation. To know an ideal, it suffices to know it up to large enough precision: we reduce the problem to local evaluation.

SUBALGORITHM 3.26 `MulCR`.
Input: a representation $y = \prod_{i=1}^r y_i^{e_i} \in \mathcal{R}$ of type (ii), an integral right $\mathcal{O}$-ideal I.
Output: the ideal yI.

$\mathfrak{a} \leftarrow \mathrm{nrd}(I)$
$z \leftarrow \texttt{EvalCR}(y, \mathfrak{a})$
return $zI + \mathfrak{a}\mathcal{O}$

PROPOSITION 3.27. *Subalgorithm 3.26 is correct.*

Proof. By Proposition 3.25, we have $z = y \pmod{\mathfrak{a}\mathcal{O}}$ so $(z-y)I \subset (z-y)\mathcal{O} \subset \mathfrak{a}\mathcal{O}$, which gives $zI + \mathfrak{a}\mathcal{O} = yI + \mathfrak{a}\mathcal{O}$. Since $y \in \mathcal{O}^\times$, we have $\mathrm{nrd}(yI) = \mathrm{nrd}(I) = \mathfrak{a}$, so $\mathfrak{a}\mathcal{O} \subset yI$ and finally $yI + \mathfrak{a}\mathcal{O} = yI$. Therefore the output of the algorithm is correct. □

4. *Complexity analysis*

We perform a complete complexity analysis of our algorithms, assuming suitable heuristics. To simplify the notations, we set $L(x) = \exp(\sqrt{\log x \log\log x})$. In every complexity estimate, the degree of the base field K is fixed. When we mention a complexity of the form $L(\Delta)^{O(1)}$, we always implicitly mean $L(\Delta)^{O(1)}$ times a polynomial in the size of the input. We fix a parameter $\alpha > 0$. We will analyse our algorithm using the general paradigm that with a factor base of subexponential size, elements have a subexponential probability of being smooth. However, recall from Definition 3.1 that the factor base $\mathcal{B}$ is assumed to generate the ray class group $\mathrm{Cl}_A(K)$, so we need the following heuristic.

HEURISTIC 4.1. There exists a constant $c = c_\alpha$ such that for every quaternion algebra A with absolute discriminant Δ, the set of primes having norm less than $c \cdot L(\Delta)^\alpha$ generate the class group $\mathrm{Cl}_A(K)$.

This heuristic is a theorem under the Generalized Riemann Hypothesis [**1**]. By Minkowski's bound, Heuristic 4.1 is also true unconditionnally with the restriction that $\log\Delta \gg_\alpha (\log|d_K|)^2$. From now on, we assume Heuristic 4.1 and we assume that the factor base $\mathcal{B}$ is the set of primes having norm less than $c \cdot L(\Delta)^\alpha$. Note that this implies the bound $\#\mathcal{B} \leq L(\Delta)^{\alpha+o(1)}$.

We start by analysing the complexity of elementary operations: Subalgorithm 3.7 and the algorithms of Section 3.3.

LEMMA 4.2. *Subalgorithm 3.7 terminates in time polynomial in the size of the input.*

Proof. Subalgorithm 3.3 (`DivideMatrix`) is made of a constant number of elementary operations, so it is polynomial. In Subalgorithm 3.5 (`GCDMatrix`) with $\mathcal{R} = \mathbb{Z}_K/\mathfrak{p}^i$, since the valuation of the determinant decreases at every recursive call, there are at most i such calls. When Subalgorithm 3.7 (`LocalGenerator`) calls Subalgorithm 3.5, we have $i = e + 1 = v_{\mathfrak{p}}(\mathrm{nrd}(b_1)) + 1 = O(\log \mathcal{N}(I))$ by lattice reduction. So the algorithm is polynomial in the size of the input. □

LEMMA 4.3. *Given the factorization of* $\mathfrak{a}$, *Subalgorithm 3.24 terminates in time polynomial in the size of the input. Given the factorization of* $\mathrm{nrd}(I)$, *Subalgorithm 3.24 terminates in time polynomial in the size of the input.*

Proof. In Subalgorithm 3.24, the number of iterations of the loop is polynomial in the size of the input. The only operations that could possibly not be polynomial in the size of the input are the computation of $\mathbb{Z}_L$ and the factorization of $\mathfrak{a}\mathfrak{f}\mathbb{Z}_L$, but $\mathbb{Z}_L$ and the factorization of $\mathfrak{f}$ are contained in the compact representation, and the factorization of $\mathfrak{a}$ is assumed to be given. So the algorithm is polynomial in the size of its input. Since the HNF of the output can be computed in polynomial time [**2**], Subalgorithm 3.24 is also polynomial. □

The restriction on the factorization is not a problem in our application: every ideal on which we call these algorithms is smooth.

Since our algorithms use their commutative counterparts, we have to make assumptions on the algorithms used to compute commutative unit groups.

HEURISTIC 4.4. There is an explicit algorithm that, given a number field F with discriminant d_F, an order $\mathcal{R}$ in F and a bound $b = L(d_F)^{O(1)}$, computes a set U of integral generators for the S-unit group of $\mathbb{Z}_F$, where S is the set of primes of norm less than b, and generators for the unit group $\mathcal{R}^\times$ expressed as products of elements in U, in expected time $L(d_F)^{O(1)}$.

This is a strong hypothesis. However, under the Generalized Riemann Hypothesis it is a theorem for quadratic number fields [**7**, **20**] and experience has shown that it is not an unreasonable assumption†.

Since the reduction algorithms depend on the structures that are given as input, we analyse the algorithms building the reduction structures before the reduction algorithms. We start with Subalgorithm 3.15, for which we need some heuristic assumptions.

---

†The PARI developers experimented extensively with this algorithm in the past twenty years, as implemented in the PARI/GP function `bnfinit`, while building and checking tables of number fields of small degree [Ref: `http://pari.math.u-bordeaux1.fr/pub/pari/packages/nftables/`], as well as with number fields of much larger degree. E.g. the class group and unit group of $K = \mathbb{Q}[t]/(t^{90} - t^2 - 1)$, $d_K > 10^{175}$ can be computed in a few hours (Karim Belabas, personal communication).

HEURISTICS 4.5. In Subalgorithm 3.15, we assume the following.

(i) If K is totally real, a positive proportion of x satisfies the condition of Step 5 that $K(x)/K$ has positive relative unit rank.
(ii) The images in $\mathrm{PGL}_2(\kappa_\mathfrak{p})$ of the units produced at Step 7 are uniformly distributed in the image of $\mathcal{O}^\times$ in $\mathrm{PGL}_2(\kappa_\mathfrak{p})$.

PROPOSITION 4.6. *Assuming Heuristics 4.4 and 4.5, the expected running time of Subalgorithm 3.15 is at most $L(\Delta)^{O(1)}$.*

Proof. We first prove that the expected number of iterations of the loop is $O(1)$. If K is not totally real, the relative unit rank condition is always satisfied, so by Heuristic 4.5 (i) a positive proportion of the iterations of the loop produce a unit. By strong approximation, the image of $\mathcal{O}^\times$ in $\mathrm{PGL}_2(\kappa_\mathfrak{p})$ contains $\mathrm{PSL}_2(\kappa_\mathfrak{p})$, and the index is at most 2. By Heuristic 4.5 (ii), with probability at least $1/2$ the image of the unit produced at Step 7 is in $\mathrm{PSL}_2(\kappa_\mathfrak{p})$, and the corresponding images are equidistributed in $\mathrm{PSL}_2(\kappa_\mathfrak{p})$. By [**8**], the probability that two random elements of $\mathrm{PSL}_2(\kappa_\mathfrak{p})$ generate this group is bounded below by a constant. Therefore, after an expected number of iterations $O(1)$, the image of X generates $\mathrm{PSL}_2(\kappa_\mathfrak{p})$ and hence acts transitively on $\mathbb{P}^1(\kappa_\mathfrak{p})$.

Each computation of a unit group in Step 7 takes expected time $L(\Delta)^{O(1)}$ by Heuristic 4.4 since the discriminant of $\mathbb{Z}_K[x]$ is $\Delta^{O(1)}$. The units are stored in compact representation and we use Subalgorithm 3.24 (`EvalCR`) to compute the action on $\mathbb{P}^1(\kappa_\mathfrak{p})$, so by Lemma 4.3 this takes total time $L(\Delta)^{O(1)}$. □

HEURISTICS 4.7. Let $\mathbb{Z}_{K,\mathcal{B},A}^\times$ be the group of $\mathcal{B}$-units in $\mathbb{Z}_K$ that are positive at every real place ramified in A. In Algorithm 3.19, we assume the following.

(i) There exists a constant $\beta > 0$ such that the elements x produced in Steps 8 and 13 are smooth with probability at least $L(\Delta)^{-\beta+o(1)}$.
(ii) The norms of the smooth elements produced in Step 13 are equidistributed in $\mathbb{Z}_{K,\mathcal{B},A}^\times/\mathbb{Z}_{K,\mathcal{B}}^{\times 2}$.

By comparison with the case of integers [**6**, Equation (1.16) and Section 1.3], $\beta = 1/(2\alpha)$ could be a reasonable value.

THEOREM 4.8. *Assume Heuristics 4.4, 4.5, 4.1 and 4.7. Then, given a maximal order $\mathcal{O}$ in an indefinite quaternion algebra A, Algorithm 3.19 (`GBuild`) terminates in expected time $L(\Delta)^{O(1)}$.*

Proof. There are $2 \cdot \#\mathcal{B} = L(\Delta)^{O(1)}$ calls to Subalgorithm 3.15 (`P1Search`). By Proposition 4.6, these calls take total time $L(\Delta)^{O(1)}$. The computation of the group $\mathbb{Z}_{K,\mathcal{B}}^\times$ takes time $L(d_K)^{O(1)} = L(\Delta)^{O(1)}$ by Heuristic 4.4. By Heuristic 4.7 (i), the expected number of iterations in the loop starting at Step 7 is at most $L(\Delta)^{O(1)}$ for each $\mathfrak{p} \in \mathcal{B}$, so the total expected number of iterations of this loop is at most $\#\mathcal{B} \cdot L(\Delta)^{O(1)} = L(\Delta)^{O(1)}$.

We study the loop starting at Step 12. After Step 4, we have $\langle\mathcal{B}\rangle/\langle\mathrm{nrd}(X)\rangle = \langle\mathcal{B}\rangle/\mathbb{Z}_{K,\mathcal{B}}^{\times 2}$. Let C be the group $\mathbb{Z}_{K,\mathcal{B},A}^\times/\mathbb{Z}_{K,\mathcal{B}}^{\times 2}$. There is an exact sequence

$$1 \longrightarrow C \longrightarrow \langle\mathcal{B}\rangle/\mathbb{Z}_{K,\mathcal{B}}^{\times 2} \longrightarrow \mathrm{Cl}_A(K) \longrightarrow 1,$$

so that the loop terminates if and only if the image of $\mathrm{nrd}(X)$ generates the group C. We have $\#C = 2^{\#\mathcal{B}+O(1)}$, so by Heuristic 4.7 (ii) this happens after we find an expected number of $\#\mathcal{B} + O(1)$ smooth elements. By Heuristic 4.7 (i), the total expected number of iterations of this loop is at most $\#\mathcal{B} \cdot L(\Delta)^{O(1)} = L(\Delta)^{O(1)}$. Checking the loop condition $\langle\mathcal{B}\rangle/\langle\mathrm{nrd}(X)\rangle \ncong \mathrm{Cl}_A(K)$ amounts to linear algebra, so it also takes time $L(\Delta)^{O(1)}$. This proves the theorem. □

THEOREM 4.9. *Assume Heuristics 4.4, 4.5, 4.1 and 4.7. Then, given the G-reduction structure computed by Algorithm 3.19 and a smooth integral right $\mathcal{O}$-ideal I, Algorithm 3.12 (*`GReduce`*) terminates in expected time $L(\Delta)^{O(1)}$.*

Proof. First, since by Theorem 4.8, Algorithm 3.19 terminates in expected time $L(\Delta)^{O(1)}$ so in particular the expected size of its output is also at most $L(\Delta)^{O(1)}$. In Subalgorithm 3.12 (`GReduce`), the first part is linear algebra so it takes time $L(\Delta)^{O(1)}$. By Lemma 4.3, all the elementary operations can be performed in time polynomial in the size of their input.

We analyse the calls to Subalgorithm 3.9 (`PReduce`). In this subalgorithm, the variable k decreases by 2 every two iterations and the initial value of k is bounded by $v_{\mathfrak{p}}(\mathrm{nrd}(J))$, so the algorithm terminates after at most $v_{\mathfrak{p}}(\mathrm{nrd}(J))$ iterations by the loop condition. So the total number of iterations in the calls to Subalgorithm 3.9 is bounded by $\sum_{\mathfrak{p}\in\mathcal{B}} v_{\mathfrak{p}}(\mathrm{nrd}(J)) \leq \log_2 \mathcal{N}(J) \leq \log_2(N_X \cdot \mathcal{N}(I))$ by Step 5 of Subalgorithm 3.12, where $N_X = \prod_{x\in X} \mathcal{N}(x)$. But $\log \mathcal{N}(I)$ is polynomial in the size of the input and $\log N_X$ is polynomial in the size of the G-reduction structure, which is $L(\Delta)^{O(1)}$. This proves the theorem.

□

Finally, for a general integral right $\mathcal{O}$-ideal, repeated attempts with Algorithm 3.14 (`IsPrincipal`) takes total expected time $L(\Delta)^{O(1)}$ if we assume the following heuristic.

HEURISTIC 4.10. There exists a constant $\gamma > 0$ such that in Step 3 of Algorithm 3.14, the element x is smooth with probability at least $L(\Delta)^{-\gamma+o(1)}$.

Again, by comparison with the case of integers [**6**], $\gamma = 1/(\sqrt{2}\alpha)$ could be a reasonable value.

5. *Examples*

We have implemented the above algorithms in the computer algebra system Magma [**3**]. In this section, we demonstrate how our algorithms work and perform in practice. Every computation was performed on a 2.5 GHz Intel Xeon E5420 processor with Magma v2.20-5 from the PLAFRIM experimental testbed.

Example 1. Let A be the quaternion algebra over $\mathbb{Q}$ generated by two elements i, j such that $i^2 = 3$, $j^2 = -1$ and $ij = -ji$. The algebra A is ramified at 2 and 3 and unramified at every other place: A is indefinite and our method applies. Let $\mathcal{O}$ be the maximal order $\mathbb{Z} + \mathbb{Z}i + \mathbb{Z}j + \mathbb{Z}\omega$ where $\omega = (1 + i + j + ij)/2$. We construct a reduction structure with Algorithm 3.19 (`GBuild`) and factor base $\mathcal{B} = \{2, 3, 5, 7, 11, 13, 17\}$. Let $I = 19\mathcal{O} + a\mathcal{O}$ where $a = -3 - 4i + j$, so that $\mathrm{nrd}(I) = 19\mathbb{Z}$. We use Algorithm 3.14 (`IsPrincipal`) to compute a generator of I. It finds an element $x = (7 + i - 9j - 3\omega)/19 \in I^{-1}$ such that $\mathrm{nrd}(xI) = 7\mathbb{Z}$, so that xI is smooth. The linear algebra phase in Subalgorithm 3.12 (`GReduce`) computes $c = -1 - 2i - j + \omega$ having norm -7 and $f = 1/7$. We obtain $J = 49\mathcal{O} + b\mathcal{O}$ with $b = -17 - 8i + j$ and $\mathfrak{I} = 7^{-1}\mathcal{O}$ before the local reduction. We reduce the ideal J at 7. In Subalgorithm 3.9 (`PReduce`), at the first iteration we have $P = P_1$ so $c = 1$. In the second iteration we have $c = (-9 - 5i - 7j - 3\omega)/7$: the element c has norm 1 and $r = 1$. After multiplying every element, we obtain the output $c =$

$7/19 \cdot (8+4i+3j-11\omega)$, $f = 1/7$ and $x = (cf)^{-1} = 3+4i-3j-11\omega$ has norm -19: x is a generator of the ideal I.

Example 2. Let K be the complex cubic field of discriminant -23, which is generated by an element t such that $t^3 - t + 1 = 0$. Let A be the quaternion algebra over K generated by two elements i, j such that $i^2 = 2t^2 + t - 3$, $j^2 = -5$ and $ij = -ji$. The algebra A is ramified at the real place of K and the discriminant δ_A has norm 5. All the maximal orders in A are conjugate and we compute one of them with Magma. Algorithm 3.19 (`GBuild`) constructs the reduction structure in 4 seconds. We then compute the 22 primes of K coprime to δ_A and having norm less than 100. For every such prime $\mathfrak{p}$, we construct a random integral right $\mathcal{O}$-ideal I with norm $\mathfrak{p}$. Since $\mathrm{Cl}_A(K)$ is trivial, they are all principal. We apply Algorithm 3.14 (`IsPrincipal`) to compute a generator of each of these ideals. This computation takes 0.3 seconds per ideal on average with a maximum of 0.9 seconds. As a comparison, we compute generators for the same ideals with the function provided by Magma. This computation takes 4 hours per ideal on average with a maximum of 69h, and 5 of the 23 ideals take less that 0.1 seconds.

Example 3. When the base field is totally real and the algebra is ramified at every real place except one, there is an algorithm of Voight [**18**] for computing the unit group of an order. In [**5**, p. 25], Dembélé and Voight mention but do not describe an unpublished algorithm using this computation to speedup ideal principalization. This algorithm is provided in Magma[†] and improves on the algorithm of [**12**]. Let K be the real cubic field of discriminant $3132 = 2^2 \cdot 3^3 \cdot 29$, which is generated by an element t such that $t^3 - 15t + 6 = 0$. Let A be the quaternion algebra over K generated by two elements i, j such that $i^2 = -1$, $j^2 = (141t^2 + 57t - 2092)/2$ and $ij = -ji$. The algebra A is ramified at two of the three real places of K and no finite place. We compute a maximal order in A and then construct the reduction structure in 14 seconds. We produce a random integral ideal of norm $\mathfrak{p}$ for each prime $\mathfrak{p}$ having norm less than 100 and compute a generator for each of them with our algorithm. The computation takes 1.5 seconds per ideal on average with a maximum of 4.6 seconds. With Magma we compute the unit group $\mathcal{O}^\times$ in 8 minutes and then compute generators for the same ideals with the units-assisted algorithm [**5**, p. 25] provided by Magma. The computation takes 1 hour per ideal on average with a maximum of 17h, and 10 of the 23 ideals take less that 0.5 seconds. Magma tends to return smaller generators than our algorithm. Magma is fast whenever there exists a small generator and our algorithm is faster when this is not the case.

Example 4. In order to understand the practical behaviour of the algorithms, we conduct the following experiment. We draw algebras A and ideals I at random[‡]. In every random test case, we compute our reduction structure with Algorithm 3.19 (`GBuild`), and we compute a generator of the ideal I with Algorithm 3.14 (`IsPrincipal`). We also compute a generator of I with the function provided by Magma. In every case, we interrupt any algorithm that takes more than 1000 seconds to terminate. The result of 15 000 such test cases is plotted in

---

[†] `IsPrincipal(<Any> I, <GrpPSL2> Gamma) -> BoolElt, AlgQuatElt`

[‡] Let x be uniformly distributed in $[0, 70]$. This value controls the size of the discriminant. Let k be 1 or 2, each with probability 1/2. This is the number of prime factors of the discriminant of the algebra. Let t be uniformly distributed in $[0, 1]$. This value controls the part of the size of the discriminant coming from the base field or from the algebra. Let d be the smallest fundamental discriminant larger that $\exp(tx/4)$, and let $K = \mathbb{Q}(\sqrt{d})$. Let $\mathfrak{p}_1$ be the prime of $\mathbb{Z}_K$ with smallest norm larger than $\exp((1-t)x/2k)$, and if $k = 2$ let $\mathfrak{p}_2$ be the prime with smallest norm larger than $N(\mathfrak{p}_1)$. Let A be the quaternion algebra ramified exactly at $\mathfrak{p}_i$ for $i \leq k$ and at $k \bmod 2$ real places of K, and let $\mathcal{O}$ be a maximal order in A. Let $\Delta = d^4 N(\delta_A)^2$ be the absolute discriminant of A. Let y be uniformly distributed in $[0, 1]$, and let $\mathfrak{p}$ be the prime of $\mathbb{Z}_K$ of smallest norm larger than $y\Delta^{1/2}$, coprime to δ_A and such that the class of $\mathfrak{p}$ in $\mathrm{Cl}_A(K)$ is trivial. Finally, let I be a random integral right $\mathcal{O}$-ideal of norm $\mathfrak{p}$.

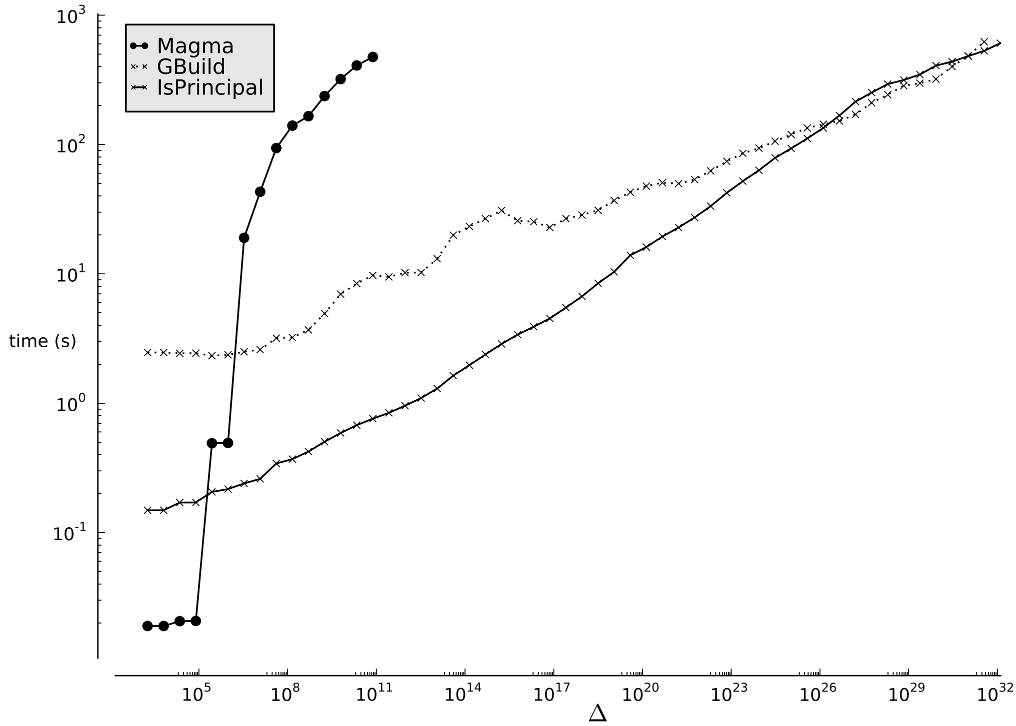

FIGURE 3. *Running time of the algorithms*

Figure 3: the discriminant Δ and the time are both in logarithmic scale, and each plot (D, T) is such that T is the average of the running time of the algorithm over the discriminants $\Delta \in [D/10, 10D]$. We do not plot the running time when more than 50% of the executions were interrupted, since the corresponding value is no longer meaningful.

Acknowledgements. I would like to thank Karim Belabas and Andreas Enge for helpful discussions and careful reading of early versions of this paper. I also want to thank an anonymous referee many comments and corrections and for suggesting a deterministic algorithm for computing a local generator and Pierre Lezowski for explaining to me Euclidean algorithms over matrix rings.

References

1. E. BACH. Explicit bounds for primality testing and related problems. *Math. Comp.* 55 (1990), no. 191, 355–380.

2. J.-F. BIASSE and C. FIEKER. A polynomial time algorithm for computing the HNF of a module over the integers of a number field. *37th International Symposium on Symbolic and Algebraic Computation (ISSAC 2012).*

3. W. BOSMA, J. CANNON and C. PLAYOUST. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

4. J. BUCHMANN. A subexponential algorithm for the determination of class groups and regulators of algebraic number fields. *Séminaire de Théorie des Nombres, Paris 1988–1989*, 27–41, Progr. Math., 91, Birkhäuser Boston, Boston, MA, 1990.

5. L. DEMBÉLÉ and J. VOIGHT. Explicit methods for Hilbert modular forms. *Elliptic curves, Hilbert modular forms and Galois deformations*, 135–198, Birkhäuser, Basel, 2013.

6. A. GRANVILLE. Smooth numbers: computational number theory and beyond. *Algorithmic number theory: lattices, number fields, curves and cryptography*, 267–323, Math. Sci. Res. Inst. Publ., 44, *Cambridge Univ. Press, Cambridge*, 2008.

7. J. L. Hafner and K. S. McCurley. A rigorous subexponential algorithm for computation of class groups. *J. Amer. Math. Soc.* 2 (1989), no. 4, 837–850.
8. W. M. Kantor and A. Lubotzky. The probability of generating a finite classical group. *Geom. Dedicata* 36 (1990), no. 1, 67–87.
9. U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Math. Comp.*, 44(170):463–471, 1985.
10. R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, STOC '83, New York, NY, USA, 1983. ACM.
11. S. Katok. Fuchsian groups. Chicago Lectures in Mathematics. *University of Chicago Press, Chicago, IL*, 1992.
12. M. Kirschmer and J. Voight. Algorithmic enumeration of ideal classes for quaternion orders. *SIAM J. Comput.*, 39(5):1714–1747, 2010.
13. J. Klüners and S. Pauli. Computing residue class rings and Picard groups of orders. *J. Algebra* 292 (2005), no. 1, 47–64.
14. C. Maclachlan and A. W. Reid. The arithmetic of hyperbolic 3-manifolds. Graduate Texts in Mathematics, 219. *Springer-Verlag, New York,* 2003.
15. I. N. Sanov. Euclid's algorithm and one-sided prime factorization for matrix rings. *Sibirsk. Mat. Ž.* **8** 1967 846–852.
16. J.-P. Serre. Trees. Translated from the French original by John Stillwell. Corrected 2nd printing of the 1980 English translation. *Springer Monographs in Mathematics.* Springer-Verlag, Berlin, 2003.
17. M.-F. Vignéras. Arithmétique des algèbres de quaternions (French). *Lecture notes in Mathematics*, Vol. 800. Springer, Berlin, 1980.
18. J. Voight. Computing fundamental domains for Fuchsian groups. *J. Théor. Nombres Bordeaux*, 21(2):469–491, 2009.
19. J. Voight. The arithmetic of quaternion algebras. *In preparation.* Accessible from the url `http://www.math.dartmouth.edu/ jvoight/research.html#books`.
20. U. Vollmer. An accelerated Buchmann algorithm for regulator computation in real quadratic fields. *Algorithmic number theory (Sydney, 2002)*, 148–162, Lecture Notes in Comput. Sci., 2369, Springer, Berlin, 2002.

Aurel Page
Univ. Bordeaux, IMB, UMR 5251,
F-33400 Talence, France.

CNRS, IMB, UMR 5251,
F-33400 Talence, France.

aurel.page@math.u-bordeaux1.fr

INRIA,
F-33400 Talence, France.