Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Symbolic Computation 39 (2005) 643–652
Journal of Symbolic Computation
www.elsevier.com/locate/jsc

Efficient algorithms for the gcd and cubic residuosity in the ring of Eisenstein integers☆

Ivan Bjerre Damgård, Gudmund Skovbjerg Frandsen*

BRICS[1], *Department of Computer Science, University of Aarhus, IT-parken, Aabogade 34, DK-8200 Aarhus N, Denmark*

Received 4 February 2004; accepted 2 December 2004
Available online 2 March 2005

Abstract

We present simple and efficient algorithms for computing the gcd and cubic residuosity in the ring of Eisenstein integers, $\mathbf{Z}[\zeta]$, i.e. the integers extended with ζ, a complex primitive third root of unity. The algorithms are similar and may be seen as generalisations of the binary integer gcd and derived Jacobi symbol algorithms. Our algorithms take time $O(n^2)$ for n-bit input. For the cubic residuosity problem this is an improvement from the known results based on the Euclidean algorithm, and taking time $O(n \cdot M(n))$, where $M(n)$ denotes the complexity of multiplying n-bit integers. For the gcd problem our algorithm is simpler and faster than an earlier algorithm of complexity $O(n^2)$. The new algorithms have applications in practical primality tests and the implementation of cryptographic protocols.

Keywords: gcd; Cubic residuosity

☆ A preliminary version of this paper was presented at FCT'2003; see Damgård and Frandsen [2003a. Efficient algorithms for gcd and cubic residuosity in the ring of Eisenstein integers. In: 14th International Symposium on Fundamentals of Computation Theory—FCT '03. Malmö, 2003. In: Lecture Notes in Comput. Sci., vol. 2751. Springer, Berlin, pp. 109–117].

* Corresponding author. Tel.: +45 8942 5600; fax: +45 8942 5601.
E-mail addresses: ivan@daimi.au.dk (I.B. Damgård), gudmund@daimi.au.dk (G.S. Frandsen).

[1] Basic Research in Computer Science, Centre of the Danish National Research Foundation.

0747-7171/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jsc.2004.02.006

1. Introduction

The Eisenstein integers, $\mathbf{Z}[\zeta] = \{a + b\zeta \mid a, b \in \mathbf{Z}\}$, are the ring of integers extended with a complex primitive third root of unity, i.e. ζ is root of $x^2 + x + 1$. Since the ring $\mathbf{Z}[\zeta]$ is a unique factorisation domain, a greatest common divisor (gcd) of two numbers is well defined (up to multiplication by a unit). The gcd of two numbers may be found using the classic Euclidean algorithm, since $\mathbf{Z}[\zeta]$ is a Euclidean domain, i.e. there is a norm $N(\cdot) : \mathbf{Z}[\zeta] \setminus \{0\} \mapsto \mathbf{N}$ such that for $a, b \in \mathbf{Z}[\zeta] \setminus \{0\}$ there is $q, r \in \mathbf{Z}[\zeta]$ such that $a = qb + r$ with $r = 0$ or $N(r) < N(b)$.

When a gcd algorithm is directly based on the Euclidean property, it requires a subroutine for division with remainder. For integers there is a very efficient alternative in the form of the binary gcd, that only requires addition/subtraction and division by two; see Stein (1967). A corresponding Jacobi symbol algorithm has been analysed as well by Shallit and Sorenson (1993).

It turns out that there are natural generalisations of these binary algorithms over the integers to algorithms over the Eisenstein integers for computing the gcd and the cubic residuosity symbol. The role of 2 is taken by the number $1 - \zeta$, which is a prime of norm 3 in $\mathbf{Z}[\zeta]$.

We present and analyse these new algorithms. It turns out that they both have bit complexity $O(n^2)$, which is an improvement over the earlier algorithms given by Scheidler and Williams (1995), Williams (1986) and Williams and Holte (1977). Their algorithms have complexity $O(nM(n))$, where $M(n)$ is the complexity of integer multiplication and the best upper bound on $M(n)$ is $O(n \log n \log\log n)$; see Schönhage and Strassen (1971). Considering the gcd problem alone, Kaltofen and Rolletschek (1989) already gave an algorithm of complexity $O(n^2)$. However, their algorithm is more complicated and a larger constant is hidden under the big Oh notation.

1.1. Related work

Schönhage (1971) presented the asymptotically fastest algorithm for integer gcd. It takes time $O(n \log n \log\log n)$. There is a derived algorithm for the Jacobi symbol of complexity $O(n(\log n)^2 \log\log n)$. For practical input sizes the most efficient algorithms seem to be variants of the binary gcd and derived Jacobi symbol algorithms; see Shallit and Sorenson (1993) and Meyer Eikenberry and Sorenson (1998).

If ω_n is a complex primitive nth root of unity, say $\omega_n = \mathrm{e}^{2\pi/n}$, then the ring $\mathbf{Z}[\omega_n]$ is known to be norm-Euclidean for only finitely many n and the smallest unresolved case is $n = 17$; see Lenstra Jr. (1979–1980), Lemmermeyer (1995).

Weilert (2000a,b) have generalised both the "binary" and the asymptotically fast gcd algorithms to $\mathbf{Z}[\omega_4] = \mathbf{Z}[i]$, the ring of Gaussian integers. For the latter case Weilert (2002) has also described a derived algorithm for computing the quartic residue symbol, and in all cases the complexity is identical to the complexity of the corresponding algorithm over $\mathbf{Z}$. Recently, Collins (2002) described an alternative algorithm of complexity $O(n^2)$ for computing the gcd over the ring of Gaussian integers.

Williams (1986) and Williams and Holte (1977) both describe algorithms for computing gcd and cubic residue symbols in $\mathbf{Z}[\omega_3]$, the Eisenstein integers. Scheidler and Williams (1995) describe algorithms for computing gcd and nth power residue symbol in $\mathbf{Z}[\omega_n]$

for $n = 3, 5, 7$. Their algorithms all have complexity $O(nM(n))$ for $M(n)$, being the complexity of integer multiplication. Kaltofen and Rolletschek (1989) gave an algorithm of complexity $O(n^2)$ for computing the gcd in any fixed quadratic number ring; in particular their algorithm also applies to the rings of Gaussian and Eisenstein integers. However, their algorithm is more complicated than the "binary" algorithms presented by Weilert and in this paper.

Weilert (2000a) suggests that his binary (i.e. $(1+i)$-ary) gcd algorithm for the Gaussian integers may generalise to other norm-Euclidean rings of algebraic integers. Our gcd algorithm for the Eisenstein integers was obtained independently, but it may nevertheless be seen as a confirmation of this suggestion in a specific case. Recently, further results have appeared. Agarwal and Frandsen (2004) have generalised the binary approach to additional imaginary quadratic rings, one of which is non-Euclidean. Wikström (2004a) has generalised the binary approach to computing both gcd and octic residue symbols in $\mathbf{Z}[\omega_8]$.

Weilert gives an algorithm for the quartic residue symbol that is derived from the asymptotically fast gcd algorithm over $\mathbf{Z}[i]$. For practical purposes, however, it would be more interesting to have a version derived from the "binary" approach. In the last section of this paper, we sketch how one can obtain such an algorithm. Independently, Wikström (2004b) has described algorithms for cubic and quartic residuosity based on the "binary" approach.

1.2. Applications

Our algorithms may be used for the efficient computation of cubic residuosity in rings other than $\mathbf{Z}[\zeta]$ when using an appropriate homomorphism. As an example, consider the finite field $GF(p)$ for prime $p \equiv 1 \bmod 3$. A number $z \in \{1, \ldots, p-1\}$ is a cubic residue precisely when $z^{(p-1)/3} \equiv 1 \bmod p$, implying that (non)residuosity may be decided by a (slow) modular exponentiation. However, it is possible to decide cubic residuosity much faster provided we carry out some preprocessing depending only on p. The preprocessing takes time proportional to a modular exponentiation, so our algorithm would only be useful if p stays fixed in several residuosity computations. The preprocessing consists in factoring p over $\mathbf{Z}[\zeta]$, i.e. finding a prime $\pi \in \mathbf{Z}[\zeta]$ such that $p = \pi\bar{\pi}$. A suitable π may be found as $\pi = \gcd(p, r-\zeta)$, where $r \in \mathbf{Z}$ is constructed as a solution to the quadratic equation $x^2 + x + 1 = 0 \bmod p$. Following this preprocessing, cubic residuosity of any z is decided using that $z^{(p-1)/3} \equiv 1 \bmod p$ if and only if $[z/\pi] = 1$, where $[\cdot/\cdot]$ denotes the cubic residuosity symbol.

When the order of the multiplicative group in question is unknown, modular exponentiation cannot be used, but it may still be possible to identify some nonresidues by computing residue symbols. In particular, the primality test of Damgård and Frandsen (2003b) uses our algorithms for finding cubic nonresidues in a more general ring.

Computation of gcd and cubic residuosity is also used for the implementation of cryptosystems by Scheidler and Williams (1995) and Williams (1986).

2. Preliminary facts about $\mathbf{Z}[\zeta]$

$\mathbf{Z}[\zeta]$ is the ring of integers extended with a primitive third root of unity ζ (complex root of $z^2 + z + 1$). We will be using the following definitions and facts; see Ireland and Rosen (1990).

Let $\bar{\cdot}$ and $N(\cdot)$ denote complex conjugation and complex norm, $N(\alpha) = \alpha\bar{\alpha}$, respectively. Note that $\bar{\zeta} = \zeta^2 = \zeta^{-1}$ and $N(a + b\zeta) = a^2 + b^2 - ab$.

A unit in $\mathbf{Z}[\zeta]$ is an element of norm 1. There are six units in $\mathbf{Z}[\zeta]$: $\pm 1, \pm\zeta, \pm\zeta^2$. Two elements $\alpha, \beta \in \mathbf{Z}[\zeta]$ are said to be associates if there exists a unit ϵ such that $\alpha = \epsilon\beta$.

A prime π in $\mathbf{Z}[\zeta]$ is a non-unit such that for any $\alpha, \beta \in \mathbf{Z}[\zeta]$, if $\pi | \alpha\beta$, then $\pi | \alpha$ or $\pi | \beta$.

$1 - \zeta$ is a prime in $\mathbf{Z}[\zeta]$ and $N(1 - \zeta) = 3$. A primary number has the form $1 + 3\beta$ for some $\beta \in \mathbf{Z}[\zeta]$. If $\alpha \in \mathbf{Z}[\zeta]$ is not divisible by $1 - \zeta$ then α is associated with a primary number, and $N(\alpha) \equiv 1 \pmod 3$. The definition of *primary* seems to vary in that some authors such as Lemmermeyer (2000) and Ireland and Rosen (1990) require the alternate forms $\pm 1 + 3\beta$ and $-1 + 3\beta$, respectively. However, our definition is more convenient in the present context.

The cubic residuosity symbol $[\cdot/\cdot] : \mathbf{Z}[\zeta] \times (\mathbf{Z}[\zeta] - (1 - \zeta)\mathbf{Z}[\zeta]) \mapsto \{0, 1, \zeta, \zeta^{-1}\}$ is defined as follows:

For prime $\pi \in \mathbf{Z}[\zeta]$ where π is not associated with $1 - \zeta$:

$$[\alpha/\pi] = (\alpha^{\frac{N(\pi)-1}{3}}) \bmod \pi,$$

where the notation "$\ldots \bmod \pi$" refers to the unique congruence value for $\alpha^{(N(\pi)-1)/3}$ among the elements $\{0, 1, \zeta, \zeta^{-1}\}$.

For number $\beta = \prod_{i=1}^{t} \pi_i^{m_i} \in \mathbf{Z}[\zeta]$ where β is not divisible by $1 - \zeta$:

$$[\alpha/\beta] = \prod_{i=1}^{t} [\alpha/\pi_i]^{m_i}.$$

Note that the definition implies

$$\begin{aligned}
[\alpha/\epsilon] &= 1 \text{ for a unit } \epsilon,\\
[\alpha/\beta] &= 0 \text{ when } \gcd(\alpha, \beta) \neq 1,\\
[\alpha/\beta] &= [\alpha'/\beta], \quad \text{when } \alpha \equiv \alpha' \pmod \beta,\\
[\alpha\alpha'/\beta] &= [\alpha/\beta] \cdot [\alpha'/\beta],\\
[-1/\beta] &= 1.
\end{aligned}$$

In addition, we will need the following laws satisfied by the cubic residuosity symbol; see Lemmermeyer (2000).

The cubic reciprocity law:

$$[\alpha/\beta] = [\beta/\alpha], \quad \text{when } \alpha \text{ and } \beta \text{ are both primary.}$$

The complementary laws (for primary $\beta = 1 + 3(m + n\zeta)$, where $m, n \in \mathbf{Z}$):

$$\begin{aligned}
[1 - \zeta/\beta] &= \zeta^m,\\
[\zeta/\beta] &= \zeta^{-(m+n)}.
\end{aligned}$$

3. Computing the gcd in $\mathbf{Z}[\zeta]$

It turns out that the well-known binary integer gcd algorithm has a natural generalisation to a gcd algorithm for the Eisenstein integers. The generalised algorithm is best understood by relating it to the binary algorithm in a nonstandard version. The authors are not aware

of any description of the latter in the literature. For the standard version see Bach and Shallit (1996).

A slightly nonstandard version of the binary gcd is the following. Every integer can be represented as $(-1)^i \cdot 2^j \cdot (4m+1)$, where $i \in \{0,1\}$, $j \geq 0$ and $m \in \mathbf{Z}$. Without loss of generality, we may therefore assume that the numbers in question are of the form $(4m+1)$. One iteration consists in replacing the numerically larger of the two numbers by their difference. If it is nonzero then the dividing 2-power (at least 2^2) may be removed without changing the gcd. If necessary the resulting odd number is multiplied with -1 to get a number of the form $4m+1$ and we are ready for the next iteration. It is fairly obvious that the product of the numeric values of the two numbers decreases by a factor at least 2 in each step until the gcd is found, and hence the gcd of two numbers a, b can be computed in time $(\log^2 |ab|)$.

To make the analogue, we recall that any element of $\mathbf{Z}[\zeta]$ that is not divisible by $1-\zeta$ is associated with a (unique) primary number, i.e. a number of the form $1+3\alpha$. This implies that any element in $\mathbf{Z}[\zeta] \setminus \{0\}$ has a (unique) representation of the form $(-\zeta)^i \cdot (1-\zeta)^j \cdot (1+3\alpha)$ where $0 \leq i < 6$, $0 \leq j$ and $\alpha \in \mathbf{Z}[\zeta]$. In addition, the difference of two primary numbers is divisible by $(1-\zeta)^2$, since $3 = -\zeta^2(1-\zeta)^2$. Now a gcd algorithm for the Eisenstein integers may be formulated as an analogue to the binary integer gcd algorithm. We may assume without loss of generality that the two input numbers are primary. Replace the (normwise) larger of the two numbers with their difference. If it is nonzero, we may divide out any powers of $(1-\zeta)$ that divide the difference (at least $(1-\zeta)^2$) and convert the remaining factor to primary form by multiplying with a unit. We have again two primary numbers and the process may be continued. In each step we are required to identify the (normwise) larger of two numbers. Unfortunately it would be too costly to compute the relevant norm, but it suffices to choose the large number based on an approximation that we can afford to compute. By a slightly nontrivial argument one may prove that the product of the norms of the two numbers decreases by a factor at least 2 in each step until the gcd is found, and hence the gcd of two numbers α, β can be computed in time $O(\log^2 N(\alpha\beta))$.

Algorithm 1 describes the details including a start-up to bring the two numbers into primary form.

Algorithm 1. Compute gcd in $\mathbf{Z}[\zeta]$

Require: $\alpha, \beta \in \mathbf{Z}[\zeta] \setminus \{0\}$

Ensure: $g = \gcd(\alpha, \beta)$

1: Let primary $\gamma, \delta \in \mathbf{Z}[\zeta]$ be defined by $\alpha = (-\zeta)^{i_1} \cdot (1-\zeta)^{j_1} \cdot \gamma$ and $\beta = (-\zeta)^{i_2} \cdot (1-\zeta)^{j_2} \cdot \delta$.
2: $g \leftarrow (1-\zeta)^{\min\{j_1, j_2\}}$
3: Replace α, β with γ, δ.
4: **while** $\alpha \neq \beta$ **do**
5: LOOP INVARIANT: α, β are primary
6: Let primary γ be defined by $\alpha - \beta = (-\zeta)^i \cdot (1-\zeta)^j \cdot \gamma$
7: Replace "approximately" larger of α, β with γ.
8: **end while**
9: $g \leftarrow g \cdot \alpha$

Theorem 1. *Algorithm 1 takes time $O(\log^2 N(\alpha\beta))$ to compute the gcd of α, β, or formulated alternatively, the algorithm has bit complexity $O(n^2)$.*

Proof. Let us assume that a number $\alpha = a + b\zeta \in \mathbf{Z}[\zeta]$ is represented by the integer pair (a, b). Observe that since $N(\alpha) = a^2+b^2-ab$, we have that $\log|a|+\log|b| \leq \log N(\alpha) \leq 2(\log|a|+\log|b|+1)$ for $|a|, |b| \geq 1$, i.e. the logarithm of the norm is proportional to the number of bits in the representation of a number.

We may do addition, subtraction on general numbers and multiplication by units in linear time. Since $(1-\zeta)^{-1} = (2+\zeta)/3$, division by (and check for divisibility by) $(1-\zeta)$ may also be done in linear time.

Clearly, the start-up part of the algorithm that brings the two numbers into primary form can be done in time $O(\log^2 N(\alpha\beta))$. Hence, we need only worry about the while loop.

We want to prove that the norm of the numbers decrease for each iteration. The challenge is to see that forming the number $\alpha - \beta$ does not increase the norm too much. In fact, by elementary properties of the complex norm, $N(\alpha-\beta) \leq 4 \cdot \max\{N(\alpha), N(\beta)\}$. Hence, for the γ computed in the loop of the algorithm, we get $N(\gamma) = 3^{-j}N(\alpha-\beta) \leq 3^{-2}4 \cdot \max\{N(\alpha), N(\beta)\}$. In each iteration, γ ideally replaces the one of α and β with the larger norm. However, we cannot afford to actually compute the norms to find out which one is the larger. Fortunately, by Lemma 2, it is possible in linear time to compute an approximate norm that may be slightly smaller than the exact norm, namely up to a factor $9/8$. When γ replaces the one of α and β with the larger approximate norm, we know that $N(\alpha\beta)$ decreases by a factor at least $9/4 \cdot 8/9 = 2$ in each iteration, i.e. the total number of iterations is $O(\log N(\alpha\beta))$.

Each loop iteration takes time $O(\log N(\alpha\beta))$ except possibly for finding the exponent of $(1-\zeta)$ that divides $\alpha-\beta$. Assume that $(1-\zeta)^{t_i}$ is the maximal power of $(1-\zeta)$ that divides $\alpha-\beta$ in the ith iteration. Then the combined time complexity of all loop iterations is $O((\sum_i t_i)\log N(\alpha\beta))$. We also know that the norm decreases by a factor of at least $3^{t_i-2}\cdot 2$ in the ith iteration, i.e. $\prod_i (3^{t_i-2}\cdot 2) \leq N(\alpha\beta)$. Since there are only $O(\log N(\alpha\beta))$ iterations it follows that $\prod_i 3^{t_i} \leq (9/2)^{O(\log N(\alpha\beta))} N(\alpha\beta)$ and hence $\sum_i t_i = O(\log N(\alpha\beta))$. □

Lemma 2. *Given $\alpha = a + b\zeta \in \mathbf{Z}[\zeta]$ it is possible to compute an approximate norm $\tilde{N}(\alpha)$ such that*

$$\frac{8}{9}N(\alpha) \leq \tilde{N}(\alpha) \leq N(\alpha)$$

in linear time, i.e. in time $O(\log N(\alpha))$.

Proof. Note that

$$N(a+b\zeta) = \frac{(a-b)^2+a^2+b^2}{2}.$$

Given $\epsilon > 0$, we let $\tilde{d}$ denote some approximation to integer d satisfying that $(1-\epsilon)|d| \leq \tilde{d} \leq |d|$. Note that

$$(1-\epsilon)^2 N(a+b\zeta) \leq \frac{\widetilde{(a-b)}^2+\tilde{a}^2+\tilde{b}^2}{2} \leq N(a+b\zeta).$$

Since we may compute $a - b$ in linear time it suffices to compute approximations and square them in linear time for some $\epsilon < 1/18$. Given d in the usual binary representation, we take $\tilde{d}$ to be $|d|$ with all but the six most significant bits replaced with zeros, in which case

$$\left(1 - \frac{1}{32}\right)|d| \leq \tilde{d} \leq |d|$$

and we can compute $\tilde{d}^2$ from d in linear time. □

4. Computing cubic residuosity in $\mathbf{Z}[\zeta]$

Just as the usual integer gcd algorithms may be used for constructing algorithms for the Jacobi symbol, so our earlier strategy for computing the gcd in $\mathbf{Z}[\zeta]$ can be used as the basis for an algorithm for computing the cubic residuosity symbol.

In each iteration we will assume the two numbers α, β to be primary with $\tilde{N}(\alpha) \geq \tilde{N}(\beta)$. We write their difference in the form $\alpha - \beta = (-\zeta)^i(1-\zeta)^j\gamma$, for primary γ. By the complementary laws of the cubic residuosity symbol, $[\alpha/\beta] = \zeta^{mj-(m+n)i}[\gamma/\beta]$ when $\beta = 1 + 3(m + n\zeta)$. If $\tilde{N}(\gamma) < \tilde{N}(\beta)$, we use the reciprocity law to swap γ and β before being ready for a new iteration. The algorithm stops when the two primary numbers are identical. If the identical value (the gcd) is not 1 then the residuosity symbol evaluates to 0.

Algorithm 2 describes the entire procedure including a start-up to ensure that the numbers are primary.

Algorithm 2. Compute cubic residuosity in $\mathbf{Z}[\zeta]$

Require: $\alpha, \beta \in \mathbf{Z}[\zeta] \setminus \{0\}$, and β is not divisible by $(1-\zeta)$
Ensure: $c = [\alpha/\beta]$

1: Let primary $\gamma, \delta \in \mathbf{Z}[\zeta]$ be defined by $\alpha = (-\zeta)^{i_1} \cdot (1-\zeta)^{j_1} \cdot \gamma$ and $\beta = (-\zeta)^{i_2} \cdot \delta$.
2: Let $m, n \in \mathbf{Z}$ be defined by $\delta = 1 + 3m + 3n\zeta$.
3: $t \leftarrow mj_1 - (m+n)i_1 \bmod 3$
4: Replace α, β by γ, δ.
5: If $\tilde{N}(\alpha) < \tilde{N}(\beta)$ then interchange α, β.
6: **while** $\alpha \neq \beta$ **do**
7: LOOP INVARIANT: α, β are primary and $\tilde{N}(\alpha) \geq \tilde{N}(\beta)$
8: Let primary γ be defined by $\alpha - \beta = (-\zeta)^i \cdot (1-\zeta)^j \cdot \gamma$
9: Let $m, n \in \mathbf{Z}$ be defined by $\beta = 1 + 3m + 3n\zeta$.
10: $t \leftarrow t + mj - (m+n)i \bmod 3$
11: Replace α with γ.
12: If $\tilde{N}(\alpha) < \tilde{N}(\beta)$ then interchange α, β.
13: **end while**
14: If $\alpha \neq 1$ then $c \leftarrow 0$ else $c \leftarrow \zeta^t$

Theorem 3. *Algorithm 2 takes time $O(\log^2 N(\alpha\beta))$ to compute $[\alpha/\beta]$, or formulated alternatively, the algorithm has bit complexity $O(n^2)$.*

Proof. The complexity analysis from the gcd algorithm carries over without essential changes. □

5. Computing gcd and quartic residuosity in the ring of Gaussian integers

We may construct fast algorithms for gcd and quartic residuosity in the ring of Gaussian integers, $\mathbf{Z}[i] = \{a + bi \mid a, b \in \mathbf{Z}\}$, in a completely analogous way to the algorithms over the Eisenstein integers. In the case of the gcd, this was essentially done by Weilert (2000a). However, the case of the quartic residue symbol may be of independent interest since such an algorithm is likely to be more efficient for practical input values than the asymptotically ultrafast algorithm by Weilert (2002).

Here is a sketch of the necessary facts; see Lemmermeyer (2000). There are four units in $\mathbf{Z}[i]$: $\pm 1, \pm i$. $1+i$ is a prime in $\mathbf{Z}[i]$ and $N(1+i) = 2$. A primary number has the form $1 + (2 + 2i)\beta$ for some $\beta \in \mathbf{Z}[i]$. If $\alpha \in \mathbf{Z}[i]$ is not divisible by $1 + i$ then α is associated with a primary number.

In particular, any element in $\mathbf{Z}[i] \setminus \{0\}$ has a (unique) representation on the form $i^j \cdot (1 + i)^k \cdot (1 + (2 + 2i)\alpha)$ where $0 \leq j < 4$, $0 \leq k$ and $\alpha \in \mathbf{Z}[i]$. In addition, the difference of two primary numbers is divisible by $(1+i)^3$, since $(2+2i) = -i(1+i)^3$. This is the basis for obtaining an algorithm for computing the gcd over the Gaussian integers analogous to Algorithm 1. This new algorithm also has bit complexity $O(n^2)$ as one may prove when using that $N((1 + i)^3) = 8$ and $N(\alpha - \beta) \leq 4 \cdot \max\{N(\alpha), N(\beta)\}$.

For computing quartic residuosity, we need the following additional facts; see Lemmermeyer (2000). If π is a prime in $\mathbf{Z}[i]$ and π is not associated with $1 + i$ then $N(\pi) \equiv 1 \pmod 4$, and the quartic residue symbol $[\cdot/\cdot] : \mathbf{Z}[i] \times (\mathbf{Z}[i] - (1 + i)\mathbf{Z}[i]) \mapsto \{0, 1, -1, i, -i\}$ is defined as follows:

For prime $\pi \in \mathbf{Z}[i]$ where π is not associated with $1 + i$:

$$[\alpha/\pi] = (\alpha^{\frac{N(\pi)-1}{4}}) \bmod \pi.$$

For number $\beta = \prod_{j=1}^{t} \pi_j^{m_j} \in \mathbf{Z}[i]$ where β is not divisible by $1 + i$:

$$[\alpha/\beta] = \prod_{j=1}^{t} [\alpha/\pi_j]^{m_j}.$$

The quartic reciprocity law and complementary laws for α and β primary with $\beta = 1 + (2 + 2i)(m + ni)$ and $m, n \in \mathbf{Z}$ are

$$\begin{aligned} [\alpha/\beta] &= [\beta/\alpha] \cdot (-1)^{\frac{N(\alpha)-1}{4} \cdot \frac{N(\beta)-1}{4}}, \\ [1 + i/\beta] &= i^{-n-(n+m)^2}, \\ [i/\beta] &= i^{n-m}. \end{aligned}$$

This is the basis for obtaining an algorithm for computing quartic residuosity analogous to Algorithm 2. This new algorithm also has bit complexity $O(n^2)$.

Acknowledgements

We wish to thank the anonymous referees for their suggestions which helped improve the presentation. In addition, one referee has communicated to us that he/she has implemented the gcd algorithm in Maple, using ten-digit floating point approximations for the norm, and performed experiments confirming that timings improve from $O(nM(n))$ to $O(n^2)$ as expected. The second author was partially supported by the IST Programme of the EU under contract number IST-1999-14186 (ALCOM-FT).

References

Agarwal, S., Frandsen, G.S., 2004. Binary gcd like algorithms for some complex quadratic rings. In: Algorithmic Number Theory: 6th International Symposium. ANTS-VI, 13–18 June 2004, Burlington, VT, USA. In: Lecture Notes in Comput. Sci., vol. 3076. Springer, Berlin, pp. 57–71.

Bach, E., Shallit, J., 1996. Algorithmic number theory. In: Efficient Algorithms. Foundations of Computing Series. vol. 1. MIT Press, Cambridge, MA.

Collins, G.E., 2002. A fast Euclidean algorithm for Gaussian integers. J. Symbolic Comput. 33 (4), 385–392.

Damgård, I.B., Frandsen, G.S., 2003a. Efficient algorithms for gcd and cubic residuosity in the ring of Eisenstein integers. In: 14th International Symposium on Fundamentals of Computation Theory—FCT '03. Malmö, 2003. In: Lecture Notes in Comput. Sci., vol. 2751. Springer, Berlin, pp. 109–117.

Damgård, I.B., Frandsen, G.S., 2003b. An extended quadratic Frobenius primality test with average and worst case error estimates. In: 14th International Symposium on Fundamentals of Computation Theory—FCT '03. Malmö, 2003. In: Lecture Notes in Comput. Sci., vol. 2751. Springer, Berlin, pp. 118–131.

Ireland, K., Rosen, M., 1990. A Classical Introduction to Modern Number Theory, 2nd edition. In: Graduate Texts in Mathematics, vol. 84. Springer-Verlag, New York.

Kaltofen, E., Rolletschek, H., 1989. Computing greatest common divisors and factorizations in quadratic number fields. Math. Comp. 53 (188), 697–720.

Lemmermeyer, F., 1995. The Euclidean algorithm in algebraic number fields. Exposition. Math. 13 (5), 385–416.

Lemmermeyer, F., 2000. Reciprocity laws. Springer Monographs in Mathematics. Springer-Verlag, Berlin (from Euler to Eisenstein).

Lenstra Jr., H.W., 1979–1980. Euclidean number fields I. Math. Intelligencer 2 (1), 6–15.

Meyer Eikenberry, S., Sorenson, J.P., 1998. Efficient algorithms for computing the Jacobi symbol. J. Symbolic Comput. 26 (4), 509–523.

Scheidler, R., Williams, H.C., 1995. A public-key cryptosystem utilizing cyclotomic fields. Des. Codes Cryptogr. 6 (2), 117–131.

Schönhage, A., 1971. Schnelle Berechnung von Kettenbruchentwicklungen. Acta Informat. 1, 139–144.

Schönhage, A., Strassen, V., 1971. Schnelle Multiplikation grosser Zahlen. Computing (Arch. Elektron. Rechnen) 7, 281–292.

Shallit, J., Sorenson, J., 1993. A binary algorithm for the Jacobi symbol. ACM SIGSAM Bull. 27 (1), 4–11.

Stein, J., 1967. Computational problems associated with Racah algebra. J. Comput. Phys. 1, 397–405.

Weilert, A., 2000a. $(1+i)$-ary GCD computation in $\mathbf{Z}[i]$ is an analogue to the binary GCD algorithm. J. Symbolic Comput. 30 (5), 605–617.

Weilert, A., 2000b. Asymptotically fast GCD computation in $\mathbf{Z}[i]$. In: Algorithmic Number Theory. Leiden, 2000. In: Lecture Notes in Comput. Sci., vol. 1838. Springer, Berlin, pp. 595–613.

Weilert, A., 2002. Fast computation of the biquadratic residue symbol. J. Number Theory 96 (1), 133–151.

Wikström, D., 2004a. On the *l*-ary gcd-algorithm and computing residue symbols. Tech. Rep. TRITA-NA-04-39, ISSN: 0348-2952, ISRN KTH/NA/R–04/06–SE, Department of Numerical Analysis and Computer Science, Royal Institute of Technology.

Wikström, D., 2004b. On the security of mix-nets and related problems. Tech. Rep. TRITA-NA-04-06, ISSN: 0348-2952, ISRN KTH/NA/R–04/06–SE, ISBN: 91-7283-717-9, Department of Numerical Analysis and Computer Science, Royal Institute of Technology, Licentiate Thesis.

Williams, H.C., 1986. An M^3 public-key encryption scheme. In: Advances in Cryptology—CRYPTO '85. Santa Barbara, Calif., 1985. In: Lecture Notes in Comput. Sci., vol. 218. Springer, Berlin, pp. 358–368.
Williams, H.C., Holte, R., 1977. Computation of the solution of $x^3 + Dy^3 = 1$. Math. Comp. 31 (139), 778–785.