On a Modular Algorithm for Computing GCDs of Polynomials Over Algebraic Number Fields*

Mark J. Encarnación†
Research Institute for Symbolic Computation
Johannes Kepler Universität, A-4040 Linz, Austria
Mark.Encarnacion@risc.uni-linz.ac.at

Abstract

Modular methods for computing the gcd of two univariate polynomials over an algebraic number field require *a priori* knowledge about the denominators of the rational numbers in the representation of the gcd. We derive a multiplicative bound for these denominators without assuming that the number generating the field is an algebraic integer. Consequently, the gcd algorithm of Langemyr and McCallum [*J. Symbolic Computation*, 8:429–448, 1989] can now be applied directly to polynomials that are not necessarily represented in terms of an algebraic integer. Worst-case analyses and experiments with an implementation show that by avoiding a conversion of representation the reduction in the computing time can be significant. We also suggest the use of an algorithm for recovering a rational number from its modular residue so that the denominator bound need not be computed explicitly. Experiments and analyses indicate that this is a good practical alternative.

1 Introduction

The modular algorithm presented by Langemyr and McCallum [11] (also Langemyr [9]) for computing the gcd of two univariate polynomials over an algebraic number field is a successful application to number fields of methods developed by Brown and Collins for the rationals. Langemyr [10] shows that a probabilistic version of the algorithm—one using fast arithmetic and dynamic evaluation—is asymptotically close to optimal. However, in each of these papers the authors required the representations of the polynomials to be in terms of an algebraic integer. We will show that this requirement is not necessary. This is an important observation since the representations in terms of an algebraic integer will be larger than the original representations and this will have a detrimental effect on the computing time.

More generally, we will derive a multiplicative bound on the denominators of the rational numbers in the representation of any monic divisor of a given univariate polynomial over the field without assuming that the number generating the field is an algebraic integer. This extends a well-known result, due to Weinberger and Rothschild [25], which is applicable when the number generating the field is an algebraic integer.

Having the new bound, we also propose a modification of the Langemyr-McCallum algorithm that avoids the explicit computation of the bound by using an algorithm for recovering a rational number from its modular residue. This modification works much better in practice, as will be demonstrated by experiments.

Our interest in this problem was sparked by experiences with an implementation of cylindrical algebraic decomposition-based quantifier elimination [3, 4, 8], in which certain gcd computations over algebraic extensions of the rationals were taking an exorbitant amount of time. Without the suggestions provided by several examples, we would not have conjectured the new result.

The organization of the paper is as follows. In Section 2 we introduce the notation that we will be using and also state some basic results from algebraic number theory. Section 3 is devoted to proving the main result of this paper. We describe how the new bound can be used in connection with polynomial gcd computations in Section 4, which also presents a modification of the Langemyr-McCallum algorithm as well as complexity analyses and experimental computing times.

2 Preliminaries

In this section we introduce the notation that will be used and give some basic definitions and results that will be needed. We refer the reader interested in more details to the books by Cohen [2], Hecke [6], and Marcus [15].

Lowercase Greek letters, with the exception of ϕ, will denote algebraic numbers. For each α there is a unique $M \in \mathbb{Z}[t]$ such that M is a primitive, irreducible polynomial with positive leading coefficient having α as a root. M is called the *minimal polynomial* of α. Throughout the paper we will fix a given minimal polynomial M and talk about the field $K = \mathbb{Q}(\alpha)$ obtained by extending the rationals by any root α of M. The degree of M, assumed to be at least 2, will be denoted by $n = \deg(M)$ and its leading coefficient by $\ell = \mathrm{lc}(M)$.

A number is an *algebraic integer* just in case it is a root of some monic polynomial with coefficients in $\mathbb{Z}$, the rational integers. The ring of all algebraic integers will be denoted

*The subject matter of this paper is within the scope of POSSO.

†Supported by the Austrian Förderungsfonds der wissenschaftlichen Forschung under grant P8572-PHY.

To appear in *ISSAC '94*, Proceedings of the International Symposium on Symbolic and Algebraic Computation, St. Catherine's College, Oxford, UK, July 20–22, 1994.

by $\mathfrak{o}$ and the subring consisting of the algebraic integers in K will be denoted by $\mathfrak{o}_K$.

Lowercase italic letters will denote non-zero rational integers, unless indicated otherwise. Define

$$\mathbb{Z}[\alpha] = \{a_0 + a_1\alpha + \cdots + a_{n-1}\alpha^{n-1} : a_i \in \mathbb{Z}\},$$

and $(1/a)\mathbb{Z}[\alpha] = \{\, (1/a)\delta : \delta \in \mathbb{Z}[\alpha]\,\}$. Note that if α is not an algebraic integer, $\mathbb{Z}[\alpha]$ will not be closed under multiplication since we require the elements of $\mathbb{Z}[\alpha]$ to be in "reduced form" and reduction modulo M may introduce fractions. For instance, if $\alpha = \sqrt{1/2}$ then $\alpha(\alpha+1) = \alpha + 1/2 \notin \mathbb{Z}[\alpha]$.

In this section, β denotes an arbitrary element of K. Evidently, β will be in $(1/b)\mathbb{Z}[\alpha]$ for some b, with b as small as possible; β is then said to have *denominator* b. The denominator is not intrinsically related to β, i.e., it also depends on α. If $F \in (1/r)\mathbb{Z}[\alpha][x]$, we say r is a *multiplicative bound* for the denominators of F. We will often omit the qualifier and speak simply of a bound when the intention is patent.

The n (not necessarily distinct) conjugates of β in K will be distinguished by superscripts, viz. $\beta^{(1)}, \ldots, \beta^{(n)}$, with $\beta = \beta^{(1)}$, say.

Write $\mathrm{res}(A, B)$ for the resultant of $A, B \in \mathbb{Z}[t]$. We will sometimes refer to $\mathrm{res}(M, \beta)$, which we define by $\mathrm{res}(M, \beta) = \mathrm{res}(M, B)$, where $\beta = B(\alpha)$ with $B \in \mathbb{Z}[t]$ and $\deg(B) < n$.

The *discriminant* of M is

$$\mathrm{disc}(M) = (-1)^{n(n-1)/2}\ell^{-1}\mathrm{res}(M, M'),$$

M' being the derivative of M. We will use D to denote $\mathrm{disc}(M)$, and d to denote the largest integer whose square divides D. The *discriminant* of α, denoted by $\Delta^2(\alpha)$, is the square of the determinant $\Delta(\alpha) = |\mathbf{V}|$ of the *Vandermonde matrix*:

$$\mathbf{V} = \begin{pmatrix} 1 & \alpha^{(1)} & {\alpha^{(1)}}^2 & \cdots & {\alpha^{(1)}}^{n-1} \\ 1 & \alpha^{(2)} & {\alpha^{(2)}}^2 & \cdots & {\alpha^{(2)}}^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{(n)} & {\alpha^{(n)}}^2 & \cdots & {\alpha^{(n)}}^{n-1} \end{pmatrix}.$$

We have the relationship $D = \ell^{2(n-1)}\Delta^2(\alpha)$.

By *the* gcd of $F, G \in K[x]$, we mean the monic gcd which we denote by $\gcd(F, G)$.

Proofs of the following two lemmas may be found in [6], which also gives an introduction to ideals and fractional ideals in number fields.

Lemma 1 *If $P(x) = \delta(x - \rho_1)(x - \rho_2)\cdots(x - \rho_m)$ has coefficients in $\mathfrak{o}$, then $\delta\rho_1\rho_2\cdots\rho_k \in \mathfrak{o}$, for $k = 1, \ldots, m$.*

In particular, $\ell\alpha^{(1)}\alpha^{(2)}\cdots\alpha^{(k)} \in \mathfrak{o}$, for $k = 1, \ldots, n$.

Lemma 2 *Let $F, G, H \in \mathfrak{o}_K[x]$ be such that $F = GH$. Denote by $\mathfrak{f}$, $\mathfrak{g}$, and $\mathfrak{h}$ the ideals in $\mathfrak{o}_K$ generated by the coefficients of F, G, and H, respectively. Then $\mathfrak{f} = \mathfrak{g}\mathfrak{h}$.*

Although it may not be immediately apparent, Lemma 2 is yet another rendition of the ubiquitous Gauss's lemma.

3 Determining Denominators

This section answers the question: Given $F \in \mathbb{Z}[\alpha][x]$, what is a bound for the monic divisors of F over K? The situation is fairly well understood when $\alpha \in \mathfrak{o}$ so let us first assume this is the case. Then $\tilde{F}$, the monic associate of F, will be in $(1/f)\mathbb{Z}[\alpha][x]$, where $f = \mathrm{res}(M, \mathrm{lc}(F))$. The other monic divisors of F are handled by the following theorem due to Weinberger and Rothschild [25].

Theorem 1 *Let α be an algebraic integer and suppose $\tilde{F} \in (1/f)\mathbb{Z}[\alpha][x]$ is monic. If G is a monic divisor of $\tilde{F}$ over K then $G \in (1/fd)\mathbb{Z}[\alpha][x]$.*

The purpose of this section is to show that we may drop the hypothesis that α be an algebraic integer. Rather than give the proof in one fell swoop, we will state intermediate results that we feel are of independent interest as separate lemmas. The idea of the proof is to look at the fractional ideals generated by the coefficients of the polynomials involved, and then to derive a bound for the elements of these ideals.

Our first problem is to determine the denominators of $\tilde{F}$. Let $\beta = \mathrm{lc}(F)$, so that $\tilde{F} = \beta^{-1}F$. Write $\beta = B(\alpha)$, where $B \in \mathbb{Z}[t]$ and $q = \deg(B) < n$. If $q = 0$, the rational integer β will be a bound for $\tilde{F}$. Otherwise, $q > 0$, and since B is relatively prime to M, we can find $\bar{M}, \bar{B} \in \mathbb{Z}[t]$, with $\deg(\bar{M}) < q$ and $\deg(\bar{B}) < n$, such that

$$\bar{M}M + \bar{B}B = f,$$

where $f = \mathrm{res}(M, B) \neq 0$. If we define $\bar{\beta} = \bar{B}(\alpha)$ then $\beta^{-1} = \bar{\beta}/f$. We will then have $f\tilde{F} = \bar{\beta}F$ and $\mathrm{lc}(f\tilde{F}) = f$. Now consider any coefficient δ of F. If α is an algebraic integer then $\mathbb{Z}[\alpha]$ is closed under multiplication and the product $\bar{\beta}\delta$ will be in $\mathbb{Z}[\alpha]$. In this case, f will be a bound for $\tilde{F}$. When α is *not* an algebraic integer, $\ell > 1$, and one would think that reduction of $\bar{\beta}\delta$ modulo M would introduce powers of ℓ into the denominator. The following lemma says that this will not happen.

Lemma 3 *If $\tilde{F}$ is the monic associate of $F \in \mathbb{Z}[\alpha][x]$ then $\tilde{F} \in (1/f)\mathbb{Z}[\alpha][x]$, where $f = \mathrm{res}(M, \mathrm{lc}(F))$.*

Proof With the notation already introduced in the preceding paragraph, we may form the equation

$$\bar{M}_k(t)M(t) + \bar{B}_k(t)B(t) = f\,t^k,$$

for $k = 0, \ldots, n+q-1$, where $\bar{M}_k, \bar{B}_k \in \mathbb{Z}[t]$ with $\deg(\bar{M}_k) < q$ and $\deg(\bar{B}_k) < n$ (see Hodge and Pedoe [7, page 148]). Dividing this equality through by $B(t)$, and substituting α for t, we get

$$\bar{B}_k(\alpha) = \bar{\beta}\alpha^k,$$

which will be in $\mathbb{Z}[\alpha]$ since $\deg(\bar{B}_k) < n$. It follows that if δ is any coefficient of F then $\bar{\beta}\delta \in \mathbb{Z}[\alpha]$, and hence that $\bar{\beta}F = f\tilde{F}$ will be in $\mathbb{Z}[\alpha][x]$. ∎

The next lemma is an easy generalization of Lemma 2.

Lemma 4 *Let $F, G, H \in K[x]$ be such that $F = GH$. Denote by $\mathfrak{f}$, $\mathfrak{g}$, and $\mathfrak{h}$ the fractional ideals in K generated by the coefficients of F, G, and H, respectively. Then $\mathfrak{f} = \mathfrak{g}\mathfrak{h}$.*

Proof Choose $\gamma, \eta \in \mathfrak{o}_K$ such that $\gamma\mathfrak{g}$ and $\eta\mathfrak{h}$ are integral ideals. Then $\gamma\eta F, \gamma G, \eta H \in \mathfrak{o}_K[x]$ and $\gamma\eta F = \gamma G\eta H$. By Lemma 2, $\gamma\eta\mathfrak{f} = \gamma\mathfrak{g}\eta\mathfrak{h}$, which implies $\mathfrak{f} = \mathfrak{g}\mathfrak{h}$. ∎

Lemma 5 *With the notation of Lemma 4, if F, G and H are monic then $\mathfrak{g} \subset \mathfrak{f}$ and $\mathfrak{h} \subset \mathfrak{f}$.*

Proof Since H is monic, $1 \in \mathfrak{h}$, which implies $\mathfrak{g} \subset \mathfrak{g}\mathfrak{h} = \mathfrak{f}$. Similarly, $\mathfrak{h} \subset \mathfrak{f}$. ∎

We would like to remark that Lemma 5 is a stronger version of Lemma 7.1 in [25] which says that if $\mathfrak{f} \subset (1/a)\mathfrak{o}_K$ then $\mathfrak{g}, \mathfrak{h} \subset (1/a)\mathfrak{o}_K$.

Our problem now is to determine the denominators of the members of a fractional ideal whose generators are given. This problem reduces to that of determining the possible denominators of products of the form $\theta\alpha^k$, where $\theta \in \mathfrak{o}_K$ and $0 \le k < n$. The next lemma provides a solution.

Lemma 6 *If $\theta \in \mathfrak{o}_K$ and $0 \le k < n$ then $\theta\alpha^k \in (1/d)\mathbb{Z}[\alpha]$.*

Proof We employ techniques used by Marcus [15, page 29]. If we write

$$\theta\alpha^k = a_1 + a_2\alpha + \cdots + a_n\alpha^{n-1}, \qquad a_i \in \mathbb{Q},$$

the conjugate equations form a linear system $\mathbf{b} = \mathbf{Va}$, where $\mathbf{b} = (\theta^{(1)}{\alpha^{(1)}}^k, \ldots, \theta^{(n)}{\alpha^{(n)}}^k)^T$ and $\mathbf{a} = (a_1, \ldots, a_n)^T$. Apply Cramer's rule to get $a_j = \delta_j/\Delta(\alpha)$, where δ_j is the determinant of the matrix obtained from $\mathbf{V}$ by replacing column j with $\mathbf{b}$. Since $D = \ell^{2(n-1)}\Delta^2(\alpha)$, we can write

$$a_j = \frac{\big(\ell^{n-1}\Delta(\alpha)\big)\big(\ell^{n-1}\delta_j\big)}{D}.$$

We will first show $Da_j \in \mathfrak{o}$, which, since Da_j is rational, will imply $Da_j \in \mathbb{Z}$. Evidently, $\ell^{n-1}\Delta(\alpha) = \sqrt{D}$ is in $\mathfrak{o}$. To show $\ell^{n-1}\delta_j \in \mathfrak{o}$, write

$$\ell^{n-1}\delta_j = \ell^{n-1}(\pm\theta^{(1)}{\alpha^{(1)}}^k\xi_{1j} \pm \cdots \pm \theta^{(n)}{\alpha^{(n)}}^k\xi_{nj}),$$

where ξ_{ij} is the determinant of the submatrix obtained from $\mathbf{V}$ by deleting the ith row and jth column. By assumption, each $\theta^{(i)}$ is in $\mathfrak{o}_K$, so it will be sufficient to show that each $\ell^{n-1}{\alpha^{(i)}}^k\xi_{ij}$ is in $\mathfrak{o}$. Now ξ_{ij} is a sum of terms, each of which is of the form $\pm{\alpha^{(1)}}^{e_1}\cdots{\alpha^{(n)}}^{e_n}$, where $0 \le e_l < n$ for $l = 1, \ldots, n$. Since ξ_{ij} will not involve any elements from the ith row of $\mathbf{V}$, we must have $e_i = 0$. We can thus express ${\alpha^{(i)}}^k{\alpha^{(1)}}^{e_1}\cdots{\alpha^{(n)}}^{e_n}$ as a product $\nu_1\cdots\nu_{n-1}$, where each ν is a product of distinct conjugates of α. Then

$$\ell^{n-1}{\alpha^{(i)}}^k{\alpha^{(1)}}^{e_1}\cdots{\alpha^{(n)}}^{e_n} = (\ell\nu_1)\cdots(\ell\nu_{n-1})$$

will be in $\mathfrak{o}$ by Lemma 1. Since $\ell^{n-1}{\alpha^{(i)}}^k\xi_{ij}$ is a sum of algebraic integers, it must itself be an algebraic integer. Therefore, $\ell^{n-1}\delta_j \in \mathfrak{o}$, which implies $Da_j \in \mathfrak{o}$. Consequently, Da_j is a rational integer which we will denote by b_j.

To show $da_j \in \mathbb{Z}$, observe that $b_j^2/D = (\ell^{n-1}\delta_j)^2$ so that b_j^2/D is an algebraic integer which is also rational, i.e., it is a rational integer. Accordingly, $a_j = b_j/D = c_j/d$, where $c_j \in \mathbb{Z}$. ∎

Setting $k = 0$ in the previous lemma yields the interesting inclusion $\mathfrak{o}_K \subset (1/d)\mathbb{Z}[\alpha]$, which, for α an algebraic integer, is familiar to students of algebraic numbers.

We are now ready to state and prove the main theorem of this paper.

Theorem 2 *Let $F \in \mathbb{Z}[\alpha][x]$. If G is a monic divisor of F over K then $G \in (1/fd)\mathbb{Z}[\alpha][x]$.*

Proof By Lemma 5, the coefficients of G will be contained in $\mathfrak{f}$, the fractional ideal generated by the coefficients of $\tilde{F}$. According to Lemma 3, the coefficients of $\tilde{F}$ will be elements of $(1/f)\mathbb{Z}[\alpha]$. Since each element of $\mathfrak{f}$ will be an $\mathfrak{o}_K$-linear combination of elements of $(1/f)\mathbb{Z}[\alpha]$, it follows from Lemma 6 that $\mathfrak{f} \subset (1/fd)\mathbb{Z}[\alpha]$. ∎

4 Computing GCDs

The results of the previous section apply to the problem of computing the gcd of two univariate polynomials over an algebraic number field using modular methods. In this section we shall discuss this application. The following notation will be fixed for the rest of the paper: F and G, elements of $\mathbb{Z}[\alpha][x]$ with $\deg(F) \ge \deg(G)$, will be the two polynomials whose gcd $H = \gcd(F, G)$ we wish to compute; the monic associate of F will be denoted by $\tilde{F}$; the resultant of the leading coefficients will be denoted by $f = \mathrm{res}\big(M, \mathrm{lc}(F)\big)$ and $g = \mathrm{res}\big(M, \mathrm{lc}(G)\big)$; we will use the bound $h = \gcd(f, g)D$ for the denominators of the gcd (see Corollary 1 below).

We begin with a brief description of the the Langemyr-McCallum gcd algorithm referring the reader to [11] for the details. Another approach to computing gcds is discussed by Smedley [22], although he concludes that an efficient implementation of [11] is superior to his method.

For a given rational prime p, define $M_p = M \bmod p$ and $\mathbf{R}_p = \mathbb{F}_p[t]/(M_p)$, where $\mathbb{F}_p$ is the Galois field with p elements. Let $\phi_p : \mathbb{Z}[\alpha] \to \mathbf{R}_p$ be the ring homomorphism defined by $\phi_p : B(\alpha) \mapsto (B(t) \bmod p) \bmod M_p$. The homomorphism from $\mathbb{Z}[\alpha][x]$ to $\mathbf{R}_p[x]$ induced by ϕ_p will also be denoted by ϕ_p, and the homomorphic images of the input polynomials will be denoted by $F_p = \phi_p(F)$ and $G_p = \phi_p(G)$.

The algorithm proceeds by computing the gcd H_p of F_p and G_p over $\mathbf{R}_p$ using the monic Euclidean polynomial remainder sequence (prs) algorithm for sufficiently many p, none of which divide h. It then reconstructs hH from its homomorphic images by Chinese remaindering; this associate of H is guaranteed to be in $\mathbb{Z}[\alpha][x]$.

For a given p, it may not be possible to compute H_p in the manner described since $\mathbf{R}_p$ will not in general be a field, and we may need to invert a zero-divisor. It may also happen that the prs computation is successful, but that h_pH_p, where $h_p = h \bmod p$, is not a homomorphic image of hH. Fortunately, for any particular input there are only finitely many of these *unlucky* primes, each of which can be detected and discarded if it happens to be used.

To prove the correctness of their algorithm, Langemyr and McCallum [11] assumed that α is an algebraic integer. Upon studying their proofs, one will see that this assumption is necessary only for finding a bound for the denominators of the gcd. Provided we stipulate that p not divide ℓ, the map ϕ_p will still be a homomorphism, this time from the subring of K generated by $\mathbb{Z}[\alpha]$ to $\mathbf{R}_p$. (If $\ell > 1$ then $\mathbb{Z}[\alpha]$ will not be closed under multiplication and will not itself be a ring.) With this additional condition on the primes we use, the following immediate corollary of Theorem 2 allows us to drop the assumption that α be an algebraic integer.

Corollary 1 *Let $F, G \in \mathbb{Z}[\alpha][x]$. If $H = \gcd(F, G)$ then $H \in (1/b)\mathbb{Z}[\alpha][x]$, where $b = \gcd(f, g)d$.*

Of course, the requirement that α be an algebraic integer is not a real restriction since we can always find an algebraic integer θ generating K and express the original inputs in terms of θ. The easiest way to obtain such an integer is to take $\theta = \ell\alpha$. The problem with this conversion is that the representations in terms of θ will be larger than the original representations. We will see in Sections 4.1 and 4.2 just how disadvantageous this conversion is.

In case α is an algebraic integer, Corollary 1 can be strengthened by replacing d with the *defect* of α, which is known to be a divisor of d [25]. However, computing either d or the defect is as hard as computing an integral basis for $\mathfrak{o}_K$ [19], so in practice D is used instead. (Hence our definition of h.) Unfortunately, D tends to be a loose bound. Given the difficulty of computing a tight bound for the gcd, we recommend a modification of the algorithm which we will now describe.

The general scheme of things is the same, but rather than reconstruct an associate of the gcd that will be in $\mathbb{Z}[\alpha][x]$, we will reconstruct the monic gcd which, however, can be expected to have fractions in the representation. To deal with this problem, we use an algorithm for recovering a rational number from its modular residue [23, 24]. (An efficient implementation of this algorithm will be described in [5].)

We compute $\gcd(F_p, G_p)$ over $\mathbf{R}_p$ by the Euclidean prs algorithm for sufficiently many p not dividing ℓD, and for which $\deg(F_p) = \deg(F)$ and $\deg(G_p) = \deg(G)$. We then reconstruct H using Chinese remaindering and the rational recovery algorithm.

We do not check whether p divides $\gcd(f, g)$. If p divided $\gcd(f, g)$, it would divide g, which would imply that the leading coefficient of G would be a zero-divisor in $\mathbf{R}_p$, and that the monic prs could not be computed. Therefore, the check is implicit in the algorithm, and this has the advantage that we avoid the computation of $\gcd(f, g)$, which may require an appreciable amount of time.

The algorithm for reconstructing rationals requires that no p divide the denominator of any rational we are trying to recover. This will be the case here since no p divides h, which is a multiple of every denominator. Thus, the algorithm will recover all of the rationals once enough primes have been processed.

We have not yet said what is meant by "sufficiently many p" in the descriptions of both the original algorithm and our modification of it. There are (at least) two ways to decide how many primes to use, either of which will return the correct result: *i*) use as many as required to guarantee that the result of the Chinese remaindering is correct; *ii*) when two successive Chinese remainderings (and rational recovery, in the case of the modified algorithm) yield the same result, say $\hat{H}$, attempt a trial division of both F and G by $\hat{H}$, using more primes if the division is not exact. For want of better names, these will be called the "play-safe" and "trial-division" methods, respectively.

In the following sections, we will be comparing three versions of the algorithm both theoretically and experimentally. Each of the three versions will have two variants corresponding to the two methods for determining how many primes to use. The first version—"Version A"—performs a preprocessing step which converts the inputs to representations in terms of the algebraic integer $\ell\alpha$, and then applies the algorithm in [11] to the converted inputs. The second—"Version B"—applies this same algorithm to the original inputs without converting representations. Both of these versions compute the bound for the gcd and perform Chinese remaindering to recover an element of $\mathbb{Z}[\alpha][x]$. The third—"Version C"—is our modification of the algorithm which does not compute the bound, but instead uses the algorithm for reconstructing rationals.

We concur with Langemyr and McCallum [11] that the trial-division method should be used in implementations, since unlucky primes are rarely encountered in practice. For this reason, we implemented trial-division variants of Versions A, B, and C, which we will compare experimentally. Theoretical computing time estimates will also be given for these variants under the assumption that no unlucky primes are encountered. Though these will not be rigorous worst-case bounds, they will model the observed timings more accurately than analyses based on less optimistic assumptions.

We will also obtain worst-case complexity bounds, but for this purpose we will be analyzing play-safe variants of the three versions. To avoid confusion, these will be referred to as Versions A$^+$, B$^+$, and C$^+$, respectively. The reason for rigorously analyzing these variants, rather than the trial-division variants that were implemented, is that in the worst case it is conceivable that the trial-division variants would be attempting unsuccessful divisions after each prime were processed. This causes the trial-division variants to have worst-case complexities worse than those of the play-safe variants.

4.1 Complexity Analysis

Following the lead of Langemyr and McCallum [11], we will count the number of word operations needed by the algorithms, and we will assume that arithmetic operations in $\mathbb{F}_p$ can be performed in constant time.

We need to define some norms. For polynomials $E(x) = \sum_{i=0}^{q} \epsilon_i x^i \in \mathbb{Q}(\alpha)[x]$, where $\epsilon_i = \sum_{j=0}^{n-1} e_{ij}\alpha^j$ and $e_{ij} \in \mathbb{Q}$, define

$$|E|_\infty = \max_{ij} \left\{ |e_{ij}| \right\}.$$

This definition also applies to elements of $\mathbb{Q}(\alpha)$ and of $\mathbb{Q}[x]$. For a matrix $\mathbf{M} = (\mu_{ij})$, of order $u \times v$, define

$$|\mathbf{M}|_\infty = \max_{1\le j\le v} \left\{ \sum_{i=1}^{u} |\mu_{ij}| \right\}.$$

For polynomials E as above, define

$$\|E\| = \max_{i} \left\{ \Big(\sum_{j=0}^{q} |\epsilon_j^{(i)}|^2 \Big)^{1/2} \right\},$$

the maximum being taken over all conjugates. Again, this definition also applies to elements of $\mathbb{Q}(\alpha)$ and of $\mathbb{Q}[x]$.

We will express the various bounds we will be deriving in terms of four parameters: the degrees $m = \deg(F) \ge \deg(G)$ and $n = \deg(M)$; and the norms $\mathcal{C} = \max(|F|_\infty, |G|_\infty)$ and $|M|_\infty$.

The first thing we will do is bound the size of the rational integers appearing in the monic divisors of F. Like Lenstra [13], we will follow the approach outlined in Section 8 of [25]; unlike Lenstra, we will allow M to be non-monic.

Lemma 7 *If F_0 is a monic divisor of F over K then*

$$|fDF_0|_\infty < 2^{m+n}\ell^{-n}(m+1)^{1/2}(n+1)^{5n}|F|_\infty^n |M|_\infty^{4n}.$$

Proof By Theorem 4 in [17], if η is any coefficient of F_0 then $\|\eta\| \le 2^m\|\tilde{F}\|$. Write $\eta = \sum_{j=0}^{n-1} h_j\alpha^j$, with $h_j \in \mathbb{Q}$, and set $\mathbf{H} = (\eta^{(1)}, \ldots, \eta^{(n)})$ and $\mathbf{h} = (h_0, \ldots, h_{n-1})$. Then $\mathbf{H} = \mathbf{h}\mathbf{V}^T$, where $\mathbf{V}^T$ is the transpose of the Vandermonde matrix. Since $\mathbf{V}^T$ is invertible, we have $\mathbf{h} = \mathbf{H}\mathbf{U}$, where $\mathbf{U}$ is the inverse of $\mathbf{V}^T$. From $|\eta|_\infty = |\mathbf{h}|_\infty \le |\mathbf{H}|_\infty |\mathbf{U}|_\infty$ and $|\mathbf{H}|_\infty = \|\eta\| \le 2^m\|\tilde{F}\|$, we infer $|\eta|_\infty \le 2^m\|\tilde{F}\|\cdot|\mathbf{U}|_\infty$ and

$$|fDF_0|_\infty \le 2^m|fD|\cdot\|\tilde{F}\|\cdot|\mathbf{U}|_\infty. \tag{1}$$

We need to bound $|\mathbf{U}|_\infty$ and $\|\tilde{F}\|$. Each entry of $\mathbf{U}$ is a subdeterminant of $\mathbf{V}^T$ of order $n-1$, divided by $\det(\mathbf{V}^T)$.

Lenstra [13] observes that each such subdeterminant is bounded by

$$(n-1)^{(n-1)/2}\Bigg(\prod_{|\alpha^{(j)}|>1}|\alpha^{(j)}|\Bigg)^{n-1}.$$

By Theorem 1 in [16], we know $\prod_{|\alpha^{(j)}|\geq 1}|\alpha^{(j)}|\leq \ell^{-1}\|M\|$. Recalling the relation $\det(\mathbf{V})=\sqrt{D}/\ell^{n-1}$, we see that

$$|\mathbf{U}|_\infty \leq |D|^{-1/2}n^n\|M\|^n. \tag{2}$$

We will now turn to $\|\tilde{F}\|$. Lenstra [13] notes that

$$\|\tilde{F}\|\leq (m+1)^{1/2}|\tilde{F}|_\infty\Bigg(\sum_{j=0}^{n-1}\|\alpha\|^{2j}\Bigg)^{1/2},$$

and Theorem 2 in [17] implies $\|\alpha\|\leq 2\ell^{-1}|M|_\infty$. These two inequalities, after a little fiddling, give

$$\|\tilde{F}\|\leq \ell^{-n}(m+1)^{1/2}|\tilde{F}|_\infty|M|_\infty^n. \tag{3}$$

Combining (1), (2), and (3) with $\|M\|\leq (n+1)|M|_\infty$, we find

$$\begin{aligned}|fDF_0|_\infty &\leq 2^m\ell^{-n}|D|^{1/2}(m+1)^{1/2}(n+1)^{5n/2}\\ &\quad\cdot|f\tilde{F}|_\infty|M|_\infty^{2n}.\end{aligned} \tag{4}$$

We want bounds for $|f\tilde{F}|_\infty$ and $|D|$ in terms of n, $|M|_\infty$, and $|F|_\infty$. Hadamard's bound gives

$$|D|\leq 2^n n^{2n}|M|_\infty^{2n}. \tag{5}$$

To bound $|f\tilde{F}|_\infty$, recall that we may form the equation

$$\bar{M}_k(t)M(t)+\bar{B}_k(t)B(t)=f\,t^k,$$

for $k=0,\ldots,n+q-1$, where $\bar{M}_k,\bar{B}_k\in\mathbb{Z}[t]$ with $\deg(\bar{M}_k)<q$ and $\deg(\bar{B}_k)<n$. Let $M=\sum_{i=0}^n c_it^i$ and $B=\sum_{j=0}^q b_jt^j$. The coefficients of $\bar{B}_k$ are determinants of certain submatrices of the *Sylvester matrix*:

$$\begin{pmatrix}
c_0 & & & b_0 & & & \\
c_1 & c_0 & & b_1 & b_0 & & \\
\vdots & \ddots & & \vdots & \ddots & & \\
 & & c_0 & & & & \\
 & & c_1 & b_q & & & \\
c_n & & & & b_q & & b_0\\
 & c_n & & & & & b_1\\
 & \ddots & \vdots & & & \ddots & \vdots\\
 & & c_n & & & & b_q
\end{pmatrix}$$

in which there are q columns of c_is and n columns of b_js. Each of the submatrices is of order $n+q-1$ and has $n-1$ columns of b_js. By Hadamard's bound and the inequality $q\leq n-1$, each of the subdeterminants, and thus also $|\bar{B}_k(\alpha)|_\infty = |f\beta^{-1}\alpha^k|_\infty$, is bounded by $\big((n+1)|M|_\infty|F|_\infty\big)^{n-1}$. Multiplying this bound by $n|F|_\infty$ gives us a bound for $|f\beta^{-1}\delta|_\infty$, where δ is any coefficient of F. Hence

$$|f\tilde{F}|_\infty\leq \big((n+1)|M|_\infty|F|_\infty\big)^n. \tag{6}$$

Finally, (4), (5), and (6) imply the asserted inequality. ∎

Given Lemma 7, one sees that the gcd H satisfies

$$|hH|_\infty < 2^{m+n}\ell^{-n}(m+1)^{1/2}(n+1)^{5n}\mathcal{C}^n|M|_\infty^{4n} \tag{7}$$

and

$$\log|hH|_\infty = O\big(m+n\log(n\mathcal{C}|M|_\infty)\big).$$

For α an algebraic integer, Langemyr and McCallum [11] use Lenstra's results [12] to get a bound of

$$O\big(m+\log\mathcal{C}+n\log(n|M|_\infty)\big),$$

which is better than the bound we just gave. However, they assumed the leading coefficients of F and G were rational integers, rather than arbitrary elements of $\mathbb{Z}[\alpha]$. Langemyr [9, 10] allows general input and comes to the same bound we get here. Lemma 7 establishes that this bound does not depend on α being an algebraic integer.

To get worst-case complexities, we need an upper bound for the number of unlucky primes. Let $k=\deg(H)$, and denote the kth *principal subresultant coefficient* of F and G by $\mathrm{psc}_k(F,G)$, where F and G are treated as univariate polynomials in x over the ring $\mathbb{Z}[t]$. (Consult Loos [14] or Mishra [18] for a definition of $\mathrm{psc}_k(F,G)$.) Define

$$r_k=\mathrm{res}\big(M,\mathrm{psc}_k(F,G)\big),$$

which will be a rational integer. In the notation used above, let $H_p=\gcd(F_p,G_p)$ where p does not divide h. From [11], we know that $\deg(H_p)\geq\deg(H)$, with strict inequality if and only if p divides r_k. We know—again from [11]—that if $\deg(H_p)=\deg(H)$ then $\phi_p(H)=H_p$. In other words, if we successfully compute the prs over $\mathbf{R}_p$, and p does not divide r_k, then p is not unlucky: H_p is a legitimate homomorphic image of H. The following lemma provides us with a bound that we will need to determine the number of unlucky primes we might encounter.

Lemma 8 *Considering F and G as univariate polynomials in x over the ring $\mathbb{Z}[t]$,*

$$|r_k|<2^{n/2}\big((m+1)(n+1)^4\mathcal{C}^2|M|_\infty^2\big)^{mn}. \tag{8}$$

Proof Expressing $\mathrm{psc}_k(F,G)$ as a determinant and applying Proposition 2.19 in [9], we find

$$|\mathrm{psc}_k(F,G)|_\infty^2\leq\big((m+1)n^2\mathcal{C}^2\big)^{2m}.$$

Since $\deg\big(\mathrm{psc}_k(F,G)\big)\leq 2m(n-1)$, by Hadamard's bound we have

$$\begin{aligned}|\mathrm{res}\big(M,\mathrm{psc}_k(F,G)\big)|_\infty^2 &\leq \big((n+1)|M|_\infty^2\big)^{2mn}\\ &\quad\cdot\big(2mn|\mathrm{psc}_k(F,G)|_\infty^2\big)^n.\end{aligned}$$

The desired result follows. ∎

Let $\wp$ be a common bound for the right-hand sides of (7) and (8). If the number of primes used by the algorithm is such that the product of the primes is at least $2\wp$ then the result of the Chinese remaindering will be the true gcd [11]. Therefore, the number of primes we will need to use is

$$O(\log\wp)=O\big(mn\log(mn\mathcal{C}|M|_\infty)\big) \tag{9}$$

in the worst case. Notice that since Lemma 8 provides a stronger inequality than Proposition 4.2 in [11], this bound on the number of primes needed is better than the bound given in [11], namely $O\big(mn\log(m\mathcal{C}|M|_\infty^n)\big)$.

We are ready to establish worst-case complexities for the play-safe variants. We will start with Version B^+; complexity bounds for Versions A^+ and C^+ will then be derived from that for B^+.

Theorem 3 *Version* B^+ *of the algorithm computes the gcd of* F *and* G *in*

$$O\big(m^4n^3\log^2(mn\mathcal{C}|M|_\infty)\big)$$

word operations.

Proof The proof is very similar to that of Theorem 5.1 in [11]. The difference is that we use the improved bound for the number of unlucky primes given by (9). ∎

Theorem 3 improves the worst-case bound

$$O\big(m^4n^3\log^2(m|F|_\infty|M|_\infty^n)\big)$$

given in [11], but this improvement is due solely to the tighter bound furnished by Lemma 8, and is independent of the results of Section 3.

Theorem 4 *Version* A^+ *of the algorithm computes the gcd of* F *and* G *in*

$$O\big(m^4n^3\log^2(mn\mathcal{C}|M|_\infty^n)\big)$$

word operations.

Proof Apply Theorem 3 to the converted inputs—call them M^*, F^*, and G^*. In the worst case, $|M^*|_\infty = O(|M|_\infty^n)$ and $\mathcal{C}^* = O(\mathcal{C}|M|_\infty^{n-1})$, where $\mathcal{C}^* = \max\{|F^*|_\infty, |G^*|_\infty\}$. ∎

Keeping the parameters other than n fixed, we see that the algorithm, which ran in time $O(n^3\log^2 n)$, now has a worst-case complexity of $O(n^5)$ due to the conversion of representations. The results of Section 3 allow us to avoid this conversion.

Theorem 5 *Version* C^+ *of the algorithm computes the gcd of* F *and* G *in*

$$O\big(m^4n^3\log^2(mn\mathcal{C}|M|_\infty)\big)$$

word operations.

Proof The only difference between Versions B^+ and C^+ that might affect their asymptotic behavior is the application by Version C^+ of the algorithm for rational recovery, which, given inputs of length $O(l)$, requires $O(l^2)$ time . This algorithm is applied once to each of $O(mn)$ sets of inputs of length $O\big(\log(mn\mathcal{C}|M|_\infty)\big)$. Hence, the worst-case complexities of Versions B^+ and C^+ are the same. ∎

If no unlucky primes are used, and the first trial division is successful, then we will say that *we are lucky*. We will now give running time estimates for Versions A, B, and C that will be derived based on the assumption that we are lucky. (Such assumptions were made in [11], also for the purpose of deriving estimates that more closely model the observed behavior of the implementations.) These estimates will involve two additional parameters: $\mathcal{G}$, the maximum of the absolute values of the numerators and denominators appearing in the gcd and in the cofactors F/H and G/H; and $\mathcal{D}$, the absolute value of $\gcd(f,g)D$. An upper bound for $\mathcal{G}$ is given by the right-hand side of (7), while one for $\mathcal{D}$ is $(2n^3|F|_\infty|M|_\infty^3)^n$, which may be derived using Hadamard's bound.

Theorem 6 *If we are lucky, Version* B *of the algorithm computes the gcd of* F *and* G *in*

$$O\big(m^2n^2(n\log^2|M|_\infty + \log^2(m\mathcal{G}) + \log(\mathcal{D}+\mathcal{G})) \\ + mn\log^2(\mathcal{D}+\mathcal{G})\big)$$

word operations.

Proof We compute $O\big(\log(\mathcal{D}+\mathcal{G})\big)$ modular gcds, each of which costs $O(m^2n^2)$. The cost of applying the Chinese remainder algorithm is $O\big(mn\log^2(\mathcal{D}+\mathcal{G})\big)$. The trial divisions—assumed to be successful—will cost the same as a multiplication of H by each of the cofactors F/H and G/H. It follows from [20, Theorem 3.3.4] that the trial divisions cost $O\big(m^2n^2(n\log^2|M|_\infty+\log^2(m\mathcal{G}))\big)$. Summing up these costs proves the theorem. ∎

Theorem 7 *If we are lucky, Version* A *of the algorithm computes the gcd of* F *and* G *in*

$$O\big(m^2n^2(n\log^2|M|_\infty + \log^2(m\mathcal{G}) + \log(\mathcal{D}+\mathcal{G})) \\ + mn\log^2(\mathcal{D}+\mathcal{G}) + m^2n^4\log\ell + mn^5\log^2\ell\big)$$

word operations.

Proof Let $\mathcal{G}^*$ and $\mathcal{D}^*$ be parameters for the converted inputs corresponding to $\mathcal{G}$ and $\mathcal{D}$. Then $\ell^{(n-1)(n-2)}\mathcal{D} \le \mathcal{D}^* \le \ell^{(n-1)(2n-3)}\mathcal{D}$ and $\mathcal{G} \le \mathcal{G}^* \le \ell^{n-1}\mathcal{G}$. The result follows from an application of Theorem 6 with these parameters. ∎

Again fixing the parameters other than n, we see that an $O(n^3)$ algorithm becomes $O(n^5)$ when representations are first converted.

Theorem 8 *If we are lucky, Version* C *of the algorithm computes the gcd of* F *and* G *in*

$$O\big(m^2n^2(n\log^2|M|_\infty + \log^2(m\mathcal{G})) + mn\log^3\mathcal{G}\big)$$

word operations.

Proof We compute $O(\log\mathcal{G})$ modular gcds for a total cost of $O(m^2n^2\log\mathcal{G})$. The algorithm for recovering a rational from its modular residue is applied $O(\log\mathcal{G})$ times to $O(mn)$ sets of inputs of length $O(\log\mathcal{G})$; the total cost is $O(mn\log^3\mathcal{G})$. Adding the cost for the trial divisions gives the result. ∎

Comparing Theorems 6 and 8, we can expect Version C to be superior to Version B, *provided* $\mathcal{G}$ *is small relative to* $\mathcal{D}$. This is usually the case in practice. Bradford [1] reports that about two-thirds of a certain set of randomly generated algebraic integers he inspected had defect equal to 1. The bound $h = \gcd(f,g)D$ can be replaced with $\gcd(f,g)$ in these cases.

4.2 Empirical Comparison

This section presents experimental evidence to support the claim that Version C is practically superior to Version B, and also to further illustrate the undesirability of having to convert representations.

We implemented Versions A, B, and C of the algorithm in SACLIB, a C-language library of computer algebra algorithms [21], and applied each to several test examples which were generated as follows. For $n = 5, 10, 15, 20, 25$, we generated a random (non-monic) minimal polynomial M of

n,k	A	B	C
5,1	0.36	0.26	0.26
5,5	0.80	0.68	0.73
10,1	9.07	3.95	2.03
10,5	22.38	10.09	5.72
15,1	89.60	29.81	8.93
15,5	217.37	70.58	23.90
20,1	471.06	134.64	27.70
20,5	1392.89	364.88	77.67
25,1	†	470.46	70.74
25,5	†	1387.62	199.32

Table 1: Average computing times (in seconds).

n,k	A	B	C
5,1	18.8	13.0	10.6
5,5	16.2	11.8	10.4
10,1	141.4	72.2	27.0
10,5	149.4	77.0	27.0
15,1	505.8	226.8	51.6
15,5	490.2	222.6	50.4
20,1	1181.2	509.0	81.8
20,5	1222.4	528.4	82.0
25,1	†	969.4	119.6
25,5	†	1022.0	120.2

Table 2: Average number of primes used.

degree n with coefficients at most n bits long. We then generated three polynomials $\bar{F}, \bar{G}, H \in \mathbb{Z}[\alpha][x]$ where $\bar{F}$ and $\bar{G}$ were each of degree 5, H had degree $k = 1, 5$, and all the integer coefficients (thinking of $\bar{F}$, $\bar{G}$, and H as bivariate polynomials over $\mathbb{Z}$) were at most 10 bits. The products $F = \bar{F}H$ and $G = \bar{G}H$, after clearing denominators, were then used as input to each of the three versions.

In the implementation of Versions A and B, a partial squarefree factorization of the discriminant was obtained by trial divisions by each of the primes less than 1,000. This was easy enough to do, and in some cases it reduced the denominator bound used.

Our timings are summarized in Table 1. The first column gives the degrees of M and H. The second, third, and fourth columns give the timings for Versions A, B, and C. Each entry is the mean for five different sets of polynomials of each input size. The times, measured in seconds, are exclusive of garbage collection and were obtained on a DECstation 5000/240. For the same set of examples, Table 2 gives the average number of primes that were used.

A list of 1,547 precomputed primes ranging from 30011 to 46337, inclusive, was made available to the algorithms.[1] Dagger (†) entries indicate that this list was exhausted and, hence, that the implementation was unable to compute the gcd. For $n = 25$, we see that converting the representations to ones in terms of an algebraic integer has placed otherwise tractable problems beyond the capabilities of our implementation. We also see from the tables that Version C outperforms Version B by a good margin.

[1] A 32-bit machine was used for the implementations and this range of primes was chosen so that the modular arithmetic could be performed without the costly function calls that would be necessary if larger primes were used.

5 Conclusions

We have derived a multiplicative bound for the denominators of the rational numbers appearing in the monic divisors of a given univariate polynomial over an algebraic number field without assuming that the number generating the field is an algebraic integer. This permits us to apply the Langemyr-McCallum algorithm [11] for computing the gcd of two univariate polynomials over the field to inputs that are not necessarily expressed in terms of an algebraic integer.

While the requirement that the representations be in terms of an algebraic integer is no loss of generality, expressing the inputs in such a form makes the representations larger. Analyses and experiments have shown that avoiding this conversion results in a significant reduction of the computing time, and an increase in the range of problems that can be solved by our implementation.

Knowing a bound for the denominators in the gcd, we also described how an algorithm for recovering a rational number from its modular residue can be used to further enhance the practical applicability of the gcd algorithm. The improved performance is due to the looseness of the *a priori* bound; by recovering rationals we avoid the explicit computation of this bound.

Algorithms for factoring polynomials over algebraic number fields, e.g., [12, 25], might benefit from similar improvements, but I have not investigated the matter.

Acknowledgements

I thank George Collins for providing the stimulus for this research, for the support he has given me, and for keeping me on my toes while writing this paper. Thanks to Ziming Li; he contributed invaluably to the proof of Lemma 3 and carefully read a draft of this paper. Thanks also go to Jeremy Johnson and Heinrich Rolletschek for the discussions I had with them.

References

[1] R. J. Bradford. *On the Computation of Integral Bases and Defects of Integrity.* PhD thesis, University of Bath, 1988.

[2] H. Cohen. *A Course in Computational Number Theory.* Graduate Texts in Mathematics. Springer-Verlag, 1993.

[3] G. E. Collins. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. In *Automata Theory and Formal Languages, 2nd GI Conference*, volume 33 of *Lecture Notes in Computer Science*, pages 134–183, Berlin, 1975. Springer-Verlag.

[4] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *J. Symbolic Computation*, 12:299–328, 1991.

[5] M. J. Encarnación. A Lehmer-type algorithm for recovering a rational number from its modular residue. In preparation, 1994.

[6] E. Hecke. *Lectures on the Theory of Algebraic Numbers.* Graduate Texts in Mathematics. Springer-Verlag, 1981.

[7] W. V. D. Hodge and D. Pedoe. *Methods of Algebraic Geometry, Book I.* Cambridge University Press, 1946.

[8] H. Hong. *Improvements in CAD-based Quantifier Elimination.* PhD thesis, Ohio State University, Columbus, Ohio, USA, 1991.

[9] L. Langemyr. *Computing the GCD of two Polynomials Over an Algebraic Number Field.* PhD thesis, NADA, Royal Institute of Technology, Stockholm, Sweden, 1988.

[10] L. Langemyr. An asymptotically fast probabilistic algorithm for computing polynomial gcd's over an algebraic number field. In *AAECC-8*, volume 508 of *Lecture Notes in Computer Science*, pages 222–233. Springer-Verlag, 1990.

[11] L. Langemyr and S. McCallum. The computation of polynomial greatest common divisors over an algebraic number field. *J. Symbolic Computation*, 8:429–448, 1989.

[12] A. K. Lenstra. Factoring polynomials over algebraic number fields. In *Proc. EUROCAL '83*, volume 162 of *Lecture Notes in Computer Science*, pages 245–254. Springer-Verlag, 1983.

[13] A. K. Lenstra. *Polynomial-time Algorithms for the Factorization of Polynomials.* PhD thesis, Universiteit van Amsterdam, 1984.

[14] R. Loos. Generalized polynomial remainder sequences. In B. Buchberger, G. E. Collins, and R. Loos, editors, *Computer Algebra—Symbolic and Algebraic Computation*, pages 115–138. Springer-Verlag, 2nd edition, 1982.

[15] D. A. Marcus. *Number Fields.* Springer-Verlag, 1977.

[16] M. Mignotte. An inequality about factors of polynomials. *Mathematics of Computation*, 28(128):1153–1157, 1974.

[17] M. Mignotte. Some useful bounds. In B. Buchberger, G. E. Collins, and R. Loos, editors, *Computer Algebra—Symbolic and Algebraic Computation*, pages 259–263. Springer-Verlag, 2nd edition, 1982.

[18] B. Mishra. *Algorithmic Algebra.* Texts and Monographs in Computer Science. Springer-Verlag, 1993.

[19] M. E. Pohst. *Computational Algebraic Number Theory.* DMV Seminar Band 21. Springer-Verlag, 1993.

[20] C. M. Rubald. *Algorithms for Polynomials Over a Real Algebraic Number Field.* PhD thesis, University of Wisconsin, Madison, 1973.

[21] Saclib 1.1 User's Guide. Technical Report 93-19, RISC-Linz, Johannes Kepler University, A-4040 Linz, Austria, March 1993.

[22] T. J. Smedley. A new modular algorithm for computation of algebraic number polynomial gcds. In *Proc. ISSAC '89*, pages 91–94. ACM Press, 1989.

[23] P. S. Wang. A p-adic algorithm for univariate partial fractions. In *Proc. SYMSAC '81*, pages 212–217. ACM Press, 1981.

[24] P. S. Wang, M. J. T. Guy, and J. H. Davenport. p-adic reconstruction of rational numbers. *SIGSAM Bulletin*, 16:2–3, 1982.

[25] P. J. Weinberger and L. P. Rothschild. Factoring polynomials over algebraic number fields. *ACM Trans. Math. Softw.*, 2(4):335–350, 1976.