COMPUTING A SQUARE ROOT FOR THE NUMBER FIELD SIEVE

JEAN-MARC COUVEIGNES

ABSTRACT. The number field sieve is a method proposed by Lenstra, Lenstra, Manasse and Pollard for integer factorization (this volume, pp. 11–40). A heuristic analysis indicates that this method is asymptotically faster than any other existing one. It has had spectacular successes in factoring numbers of a special form. New technical difficulties arise when the method is adapted for general numbers (this volume, pp. 48–89). Among these is the need for computing the square root of a huge algebraic integer given as a product of hundreds of thousands of small ones. We present a method for computing such a square root that avoids excessively large numbers. It works only if the degree of the number field that is used is odd. The method is based on a careful use of the Chinese remainder theorem.

1. INTRODUCTION

We begin by recalling the basic scheme of the number field sieve, cf. [7]. Let n be a positive integer that is not a power of a prime number. In order to factor n, we first find many congruences modulo n involving a given set of numbers called the *basis*. This is done by means of a suitable ring of algebraic integers. To construct this ring, one chooses a positive integer d, which is the degree of the ring to be constructed; if n has between 110 and 160 decimal digits then $d = 5$ is a good choice. Next one chooses a monic polynomial $f \in \mathbf{Z}[X]$ of degree d that represents n, i. e., $f(m) = n$ for some integer m. We want both m and the coefficients of f to be as small as possible. Indeed, we can easily make them smaller than $n^{1/d}$. Now, if f is reducible, then a non-trivial factor h of f should give a non-trivial factor $h(m)$ of n. Otherwise, we consider the number field $K = \mathbf{Q}[X]/(f)$ together with the morphism φ from the order $\mathbf{Z}[\alpha]$ to $\mathbf{Z}/n\mathbf{Z}$ for which $\varphi(\alpha) = m$, where we write $\alpha = (X \bmod f)$. We then look for algebraic numbers of the form $a + b\alpha$ such that both $a + bm$ and $a + b\alpha$ are *smooth*. This means that, for a suitable positive integer y chosen at the beginning, both $a + bm$ and the norm $\mathbf{N}(a + b\alpha)$ of $a + b\alpha$ are integers divisible only by prime numbers not exceeding y. With each prime number $p \leq y$ we associate a function $\nu_p: \mathbf{Z} - \{0\} \to \mathbf{Z}/2\mathbf{Z}$, which assigns to any non-zero integer k the residue class modulo 2 of the number of factors p in k. In the same way, we associate with any prime ideal $\mathfrak{p}$ of $\mathbf{Z}[\alpha]$ of norm at most y the function $\nu_{\mathfrak{p}}: \mathbf{Z}[\alpha] - \{0\} \to \mathbf{Z}/2\mathbf{Z}$ that maps an element to the residue class modulo 2 of the number of factors $\mathfrak{p}$ appearing in that element (cf. [3, Section 5]).

Membre de l'Option Recherche du Corps des ingénieurs de l'Armement.

Ce travail a été en partie réalisé à l'occasion d'un mémoire de D. E. A. à l'Ecole Polytechnique, sous la direction de MM. Marc Chardin, Marc Giusti et Jacques Stern, en Mai 1991.

March 24, 1993.

Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX

As suggested by Adleman [1], we also use characters obtained in the following way. Choose a collection of non-zero prime ideals $\mathfrak{q}$ of $\mathbf{Z}[\alpha]$ that are different from all primes $\mathfrak{p}$ used before and that do not divide the discriminant of f. With each such $\mathfrak{q}$ we associate the function $\chi_{\mathfrak{q}}: \mathbf{Z}[\alpha] - \mathfrak{q} \to \mathbf{Z}/2\mathbf{Z}$ defined by

$$\chi_{\mathfrak{q}}(x) = \begin{cases} 0 & \text{if } x \text{ is a square modulo } \mathfrak{q}, \\ 1 & \text{otherwise.} \end{cases}$$

The first part of the algorithm consists of the search for many pairs (a, b) of relatively prime rational integers for which both $a + bm$ and $a + b\alpha$ are smooth, and $a + bm > 0$. In the second part one looks for subsets S of the set of pairs that have been found for which

$$\prod_{(a,b)\in S} (a + bm) \quad \text{is a square in } \mathbf{Z},$$

$$\prod_{(a,b)\in S} (a + b\alpha) \quad \text{is a square in } K.$$

One hopes that this is ensured by the following three conditions:

$$\sum_{(a,b)\in S} \nu_p(a + bm) = 0 \bmod 2 \qquad \text{for all prime numbers } p \le y,$$

$$\sum_{(a,b)\in S} \nu_{\mathfrak{p}}(a + b\alpha) = 0 \bmod 2 \qquad \text{for all prime ideals } \mathfrak{p} \text{ of } \mathbf{Z}[\alpha] \text{ of norm } \le y,$$

$$\sum_{(a,b)\in S} \chi_{\mathfrak{q}}(a + b\alpha) = 0 \bmod 2 \qquad \text{for all } \mathfrak{q} \text{ that have been chosen.}$$

Indeed, these conditions are necessary. If enough characters $\chi_{\mathfrak{q}}$ have been chosen then one may expect that the conditions are sufficient as well.

This leads to a large algebraic number γ that is given as a product of many small ones, and that is a square in $\mathbf{Z}[\alpha]$ (see [3]):

$$\gamma = f'(\alpha)^2 \cdot \prod_{(a,b)\in S} (a + b\alpha) = \beta^2 \qquad \text{with } \beta \in \mathbf{Z}[\alpha]. \tag{1}$$

We also know that the image of γ under φ,

$$\varphi(\gamma) = f'(m)^2 \cdot \prod_{(a,b)\in S} (a + bm) \bmod n,$$

satisfies

$$f'(m)^2 \cdot \prod_{(a,b)\in S} (a + bm) = f'(m)^2 \cdot \prod_{p \le y} p^{2e_p} = v^2,$$

where the integers e_p can be determined from the prime factorization of the numbers $a + bm$, and where

$$v = f'(m) \cdot \prod_{p \le y} p^{e_p}.$$

As for β, we know its decomposition as an *ideal* of $\mathbf{Z}[\alpha]$; but since we do not know generators for the prime ideals of norm at most y, this does not enable us to write down an explicit expression for β itself. However, we do know an expression for the *norm* of β:

$$\mathbf{N}(\beta) = \pm\mathbf{N}(f'(\alpha)) \cdot \prod_{p \leq y} p^{f_p}, \tag{2}$$

where the f_p are non-negative integers that can be determined from the prime ideal decomposition of β. Furthermore, since $\beta \in \mathbf{Z}[\alpha]$ there exists a polynomial $B \in \mathbf{Z}[X]$ of degree at most $d-1$ such that $\beta = B(\alpha)$. We shall compute this polynomial.

The method suggested in [3] is as follows. First, look for an odd prime q such that the polynomial f remains irreducible modulo q. Then, compute $\gamma \bmod q$ by performing all multiplications in the product (1) modulo q. We view $\gamma \bmod q$ as an element of the finite field $\mathbf{F}_{q^d}$, and we can easily compute the square roots of this element. Next, we choose one of the two square roots and lift it to a square root modulo q^2, q^4, q^8, ..., using Newton's method, until the modulus is larger than twice a given estimate of the size of the coefficients of B. In this manner we find B. The congruence

$$B(m)^2 = \varphi(\beta)^2 = \varphi(\beta^2) = \varphi(\gamma) = v^2 \bmod n$$

suggests that $\gcd(B(m) - v, n)$ has a good chance to be a non-trivial factor of n.

One of the difficulties with this method is the very large size of the numbers that occur in the last iterations of Newton's method. The time taken by this computation is comparable to the time taken by the entire algorithm, except if one uses fast multiplication techniques; but even if that is done one may have serious practical difficulties with the very large integers that arise. It is particularly disconcerting that the huge numbers that we compute are ultimately replaced by their remainder modulo n.

The approach that we suggest is to work with many different moduli $m_i = q_i^{k_i}$, where the q_i are distinct odd primes for which f is irreducible modulo q_i. We first compute, as above, a square root β_i of γ modulo each m_i, i.e., a polynomial $B_i \in \mathbf{Z}[X]$ of degree at most $d-1$ such that $\beta_i = B_i(\alpha)$ satisfies $\beta_i^2 \equiv \gamma \bmod m_i$; the coefficients of B_i matter only modulo m_i. If $\beta = B(\alpha)$ denotes, as above, one of the two square roots of γ in $\mathbf{Z}[\alpha]$, then we have $\beta_i = \pm\beta \bmod m_i$, where the signs are *a priori* unknown. Our first problem is to compute those signs or, equivalently, to make sure that the various β_i are congruent to the *same* square root β modulo m_i. Next, using the Chinese remainder theorem, we can compute $\beta = B(\alpha) \in \mathbf{Z}[\alpha]$ and $\varphi(\beta) = (B(m) \bmod n) \in \mathbf{Z}/n\mathbf{Z}$. However, the coefficients of B and the number $B(m)$ are so large one should avoid explicitly calculating any of them. We shall see that once the $B(m) \bmod m_i$ are known, we can compute $B(m)$ modulo n without computing B or $B(m)$ itself.

2. Description and analysis of the method

We first consider the sign problem discussed at the end of Section 1. We shall make the assumption that the degree of the extension $K/\mathbf{Q}$ is *odd*. The basic observation is that, under this hypothesis, we have $\mathbf{N}(-x) = -\mathbf{N}(x)$ for any non-zero element x of K. Hence, exactly one of the two square roots of γ has positive norm. Let that one be called β. Suppose, as above, that we know a square root $\beta_i = B_i(\alpha)$ of γ modulo $m_i = q_i^{k_i}$ for each i, and that we want to test whether $\beta_i \equiv \beta \bmod m_i$ or $\beta_i \equiv -\beta \bmod m_i$. We can decide this by looking modulo q_i. Thus, we compute the norm of $(\beta_i \bmod q_i)$, viewed as an element of the finite field of cardinality q_i^d; this norm is the $(q_i^d - 1)/(q_i - 1)$th power of $(\beta_i \bmod q_i)$, and it belongs to the prime field $\mathbf{Z}/q_i\mathbf{Z}$. We compare this norm with the residue modulo q_i of the norm of β, which is computed by means of formula (2), but with $\pm\mathbf{N}(f'(\alpha))$ replaced by its absolute value; the multiplications in (2) are performed modulo q_i. If the two norms are equal, then $\beta_i \equiv \beta \bmod m_i$, and we keep B_i. If they are opposite, then we replace B_i by $-B_i$.

Substituting m in B_i we find $B(m)$ modulo m_i for all i, and we wish to compute $B(m)$ modulo n. We discuss this problem in a more general setting.

2.1. Changing moduli. In *modular arithmetic* one represents an integer by means of its residue classes modulo each of a set of pairwise coprime integers m_i. The theoretical basis of modular arithmetic is formed by the Chinese remainder theorem. An introduction to the algorithmic aspects of modular arithmetic, and a discussion of its applications, can be found in [4, Section 4.3.2].

We consider the following algorithmic problem from modular arithmetic (cf. [9, Section 4]). One is given a collection of pairwise coprime positive integers m_i, a positive integer n, for each i an integer x_i with $0 \le x_i < m_i$, and a small positive real number ϵ, for example $\epsilon = 0.01$. In addition, one is provided with the information that there exists an integer x satisfying $x \equiv x_i \bmod m_i$ for each i, and $|x| < (\frac{1}{2} - \epsilon)\prod_i m_i$; clearly, such an integer x is unique if it exists. The question is to compute the residue class of x modulo n.

If we define the quantities

$$M = \prod_i m_i, \tag{3}$$

$$M_i = \prod_{j \ne i} m_j = M/m_i, \tag{4}$$

$$a_i = 1/M_i \bmod m_i, \qquad 0 \le a_i < m_i, \tag{5}$$

then the number $z = \sum_i a_i M_i x_i$ is congruent to x modulo M. Hence, if we round z/M to an integer: $r = [\frac{z}{M} + \frac{1}{2}]$, then we have $x = z - rM$. The point is that we can calculate r without calculating the possibly very large number z, as follows.

From $x = z - rM$ and our hypothesis $|x| < (\frac{1}{2} - \epsilon)M$ it follows that $\frac{z}{M} + \frac{1}{2}$ is not within ϵ of an integer. Hence, to calculate r it suffices to know an approximation t to z/M with $|t - z/M| < \epsilon$. Such an approximation can be obtained from

$$\frac{z}{M} = \sum_i \frac{a_i x_i}{m_i}. \tag{6}$$

All terms in the sum are between 0 and $\max_i m_i$, so they can be computed as low precision real numbers.

This results in the following algorithm. Denote by $\mathrm{rem}(a, b)$ the remainder of the Euclidean division of a by b.

1. For each i, compute $\mathrm{rem}(M_i, m_i)$ by multiplying out the product (4) modulo m_i, and compute the numbers a_i as in (5) with the extended Euclidean algorithm.
2. Compute $\mathrm{rem}(M, n)$ by multiplying out the product (3) modulo n, and compute $\mathrm{rem}(M_i, n)$ for each i; if $\gcd(m_i, n) = 1$ one can do this by dividing $\mathrm{rem}(M, n)$ by m_i modulo n;
3. Compute a number t that differs by less than ϵ from the sum in (6), and round t to an integer: $r = [t + \frac{1}{2}]$.
4. Output
$$\mathrm{rem}(x, n) = \mathrm{rem}\Big(\Big(\sum_i a_i \,\mathrm{rem}(M_i, n) x_i\Big) - r\,\mathrm{rem}(M, n), n\Big).$$
(If m_i is much larger than n one may prefer to replace a_i and x_i by $\mathrm{rem}(a_i, n)$ and $\mathrm{rem}(x_i, n)$ in this expression.)

Note that we never handle numbers substantially larger than the moduli.

2.2. Size and complexity. We derive upper bounds for the integers b_i for which $\beta = b_0 + b_1\alpha + \cdots + b_{d-1}\alpha^{d-1}$, with β as in (1). For $f = \sum_{i=0}^{d} a_i X^i$ we write $\|f\| = \big(\sum_{i=0}^{d} a_i^2\big)^{1/2}$, and we let u be an upper bound for all numbers $|a|$, $|b|$ for which $(a, b) \in S$.

Proposition. *We have*
$$|b_i| \leq d^{3/2} \cdot \|f\|^{d-i} \cdot (2u\|f\|)^{\#S/2}$$
for $0 \leq i \leq d-1$.

Proof. The field K has d embeddings into the field of complex numbers, and we denote the image of an element $\epsilon \in K$ under the kth embedding by $\epsilon^{(k)}$. We have $f = \prod_{i=1}^{d}(X - \alpha^{(i)})$, and
$$\max(1, |\alpha^{(k)}|) \leq \prod_{j=1}^{d} \max\big(1, |\alpha^{(j)}|\big) \leq \|f\| \tag{7}$$
for each k; the first inequality is trivial, and the second is due to Landau [5; 8, Chapitre IV, Section 3.3].

Let $\delta_0, \delta_1, \ldots, \delta_{d-1} \in K$ be defined by $\sum_{i=0}^{d-1} \delta_i X^i = f/(X - \alpha)$, so that $\delta_i = \sum_{j=0}^{d-1-i} a_{i+j+1}\alpha^j$. By [6, Chapter III, Proposition 2] we have
$$b_i = \mathrm{Tr}(\delta_i \beta / f'(\alpha)) = \sum_{k=1}^{d} \delta_i^{(k)} \beta^{(k)} / f'(\alpha^{(k)}), \tag{8}$$
where $\mathrm{Tr}\colon K \to \mathbf{Q}$ is the trace function. The Cauchy-Schwarz inequality and (7) imply that
$$|\delta_i^{(k)}|^2 \leq \|f\|^2 \cdot \sum_{j=0}^{d-1-i} |\alpha^{(k)}|^{2j} \leq d \cdot \|f\|^{2(d-i)} \qquad (0 \leq i \leq d-1).$$

From

$$|\beta^{(k)}/f'(\alpha^{(k)})|^2 = \prod_{(a,b)\in S} |a + b\alpha^{(k)}| \le (2u\|f\|)^{\#S}$$

we now obtain

$$|b_i| = \Big|\sum_{k=1}^{d} \delta_i^{(k)} \beta^{(k)}/f'(\alpha^{(k)})\Big| \le d^{3/2} \cdot \|f\|^{d-i} \cdot (2u\|f\|)^{\#S/2},$$

as required.

If f is chosen as in [3], then we have $\|f\| \le \sqrt{d} \cdot n^{1/d}$ and $|m| \le n^{1/d}$, and therefore

$$|B(m)| \le d^{(d+5)/2} \cdot n \cdot (2u\sqrt{d}n^{1/d})^{\#S/2}. \tag{9}$$

Note that the three factors in this upper bound are of completely different orders of magnitude. In realistic cases, the last factor has millions of decimal digits, the middle one has between one and two hundred digits, and the first one has just a few digits. We refer to [3, 9.3 and Section 11] for an estimate of u and $\#S$ as functions of n and d.

The estimate (9) is good enough to enable us to get a rough impression of the precision needed, i. e., the number and size of moduli m_i. Note that too many moduli would make us waste time, while not enough moduli would give an incorrect result. In order to get a more accurate estimate, we can explicitly compute the zeroes $\alpha^{(k)}$ of f as low precision complex numbers. We can then evaluate the complex numbers $\delta_i^{(k)}$ and $f'(\alpha^{(k)})$, and, multiplying out the product (1), the real numbers $|\beta^{(k)}|$. Then we obtain from (8) an upper bound for $|b_i|$. This leads to an upper bound for $B(m)$ that is better than (9).

We now give a rough estimate of the complexity of our method as a function of the logarithm of an upper bound like (9) for $B(m)$; let this logarithm be called s. For the sake of comparison, we mention that the square root method proposed in [3] takes time $s^{1+o(1)}$ if fast multiplication techniques are used, and $s^{2+o(1)}$ otherwise, and that the entire number field sieve is conjectured to run in time $s^{2+o(1)}$, see [3, 9.3 and Section 11]; here and below the $o(1)$ is for $n \to \infty$. The time taken by our method depends on the size and the number of the moduli, and on whether or not fast multiplication techniques are used. We consider two extreme cases.

In the first case the moduli are chosen as small as possible. We assume, heuristically, that f is irreducible modulo one out of every d primes (see the remark below). We take the m_i to be the first $ds(1+o(1))/\log(ds)$ such primes. Their product will then be $s^{1+o(1)}$, which is large enough for the algorithm of 2.1. From $d = s^{o(1)}$ one sees that both the number of moduli and the largest of them is of the order $s^{1+o(1)}$. The most time-consuming part of the method is the computation of the product (1) modulo each m_i. This requires $s^{2+o(1)}$ multiplications. Since all multiplications are done with small numbers, it is not clear how fast multiplication techniques can help at all. It is conceivable, though, that the computation can be speeded up by some divide-and-conquer technique.

The second extreme possibility is to fix the number of moduli and to choose each m_i to be $\exp(s^{1+o(1)})$ for some positive constant c. In this case we can use fast multiplication, and the time taken is $s^{1+o(1)}$. This is also achieved in [3] by means of a single modulus. With this choice of moduli one does ultimately handle very large integers.

The conclusion is that from a theoretical point of view our method does not represent an improvement over [3]. In practice, however, our method has the advantage of offering the possibility to work with much smaller numbers, and in addition it can be run in parallel. The number and the size of the moduli to be used depend strongly on the features of the available computing equipment. In [2] many small moduli are chosen, which is justified by the use of a massively parallel computer. One can imagine that in other situations it is desirable to use larger moduli so as to take advantage of sophisticated multiplication techniques. In principle it is not necessary to let the moduli be of the same approximate size: if several different computers are used, then one can adapt the sizes of the moduli to the individual machines.

Remark. It was pointed out in [3] that primes q for which f is irreducible modulo q do not necessarily exist, but that for "most" f one may expect that one out of every d primes has this property (see [10]). If the degree d, which we assumed to be odd, is a prime number, then indeed every irreducible polynomial $f \in \mathbf{Z}[X]$ of degree d remains irreducible modulo at least one out of every d primes (asymptotically). This applies in particular to $d = 3$, 5, and 7. For the proof it suffices, by the argument of [3, Proposition 9.1], to show that every transitive permutation group G of prime degree d contains at least $\#G/d$ cycles of length d. Indeed, by Cauchy's theorem G has an element of order d, and this element must be a d-cycle. Letting G act by conjugation on the set of its d-cycles one finds that the number of d-cycles is divisible by $\#G/d$, as required.

The complete algorithm may be summarized as follows.

Algorithm. Let the integers n and m, the integer $v \bmod n$, the polynomial f, and the set S of pairs (a, b) be given, as in the number field sieve. This algorithm determines the possibly trivial factorization of n that the set S gives rise to.

1. Choose the moduli $m_i = q_i^{k_i}$ according to the above remarks. This necessitates the computation of complex approximations to the $\alpha^{(i)}$ and $|\beta^{(i)}|$.
2. Compute for each i the numbers $\operatorname{rem}(M_i, m_i)$, a_i, and $\operatorname{rem}(M_i, n)$ as in 2.1.
3. For each modulus m_i, compute the product
$$\gamma_i = f'^2 \cdot \prod_{(a,b)\in S} (a + bX) \bmod (f, m_i),$$
as well as a square root β_i of γ_i.
4. Compute $N_{1,i}$, the norm of $\beta_i \bmod q_i$, and $N_{2,i}$, the product in (2) modulo q_i. If $N_{1,i} \neq N_{2,i}$ then replace β_i by $-\beta_i$. Let $B_i \in \mathbf{Z}[X]$ be a polynomial of degree $\leq d - 1$ for which $\beta_i = B_i \bmod (f, m_i)$, and compute $B_i(m)$ (modulo m_i). This step is to be performed for each i.
5. Compute $B(m) \bmod n$ from the $B_i(m)$ using the quantities calculated in step 2 (see 2.1).
6. Output $\gcd(B(m) - v, n)$.

Note that steps 3 and 4 can be parallelized.

References

1. L.M. Adleman, *Factoring numbers using singular integers*, Proc. 23rd Annual ACM Symp. on Theory of Computing (STOC) (1991), 64–71.
2. D.J. Bernstein, A.K. Lenstra, *A general number field sieve implementation*, this volume, 98–119.
3. J.P. Buhler, H.W. Lenstra, Jr., Carl Pomerance, *Factoring integers with the number field sieve*, this volume, pp. 48–89.
4. D.E. Knuth, *The art of computer programming*, volume 2, second edition, Addison-Wesley, Reading, Mass., 1981.
5. E. Landau, *Sur quelques théorèmes de M. Petrovič relatifs aux zéros des fonctions analytiques*, Bull. Soc. Math. France **33** (1905), 251–261.
6. S. Lang, *Algebraic number theory*, Addison-Wesley, Reading, Massachusetts, 1970.
7. A.K. Lenstra, H.W. Lenstra, Jr., M.S. Manasse, J.M. Pollard, *The number field sieve*, this volume, pp. 11–40. Extended abstract: Proc. 22nd Annual ACM Symp. on Theory of Computing (STOC) (1990), 564–572.
8. M. Mignotte, *Mathématiques pour le calcul formel*, Presses Universitaires de France, Paris, 1989.
9. P.L. Montgomery, R.D. Silverman, *An FFT extension to the $P-1$ factoring algorithm*, Math. Comp. **54** (1990), 839–854.
10. B.L. van der Waerden, *Algebra*, seventh edition, Springer-Verlag, Berlin, 1966.

U. M. R. d'Algorithmique Arithmétique de Bordeaux, Université de Bordeaux

Groupe de recherche en complexité et cryptographie, L. I. E. N. S., URA 1327 du CNRS, D. M. I., Ecole Normale Supérieure, 45 rue d'Ulm, 75230 Paris Cedex 05, France

E-mail address: couveign@dmi.ens.fr