INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Tuning and generalizing van Hoeij's algorithm

Karim Belabas — Guillaume Hanrot — Paul Zimmermann

N° ????

1st February 2001

THÈME 2

Rapport de recherche

ISSN 0249-6399

Tuning and generalizing van Hoeij's algorithm

Karim Belabas*, Guillaume Hanrot†, Paul Zimmermann‡

Thème 2 — Génie logiciel
et calcul symbolique
Projet SPACES

Rapport de recherche n° ???? — 1st February 2001 — 13 pages

Abstract: Recently, van Hoeij's published a new algorithm for factoring polynomials over the rational integers [11]. This algorithms rests on the same principle as Berlekamp-Zassenhaus [2, 13], but uses lattice basis reduction to improve drastically on the recombination phase. The efficiency of the LLL algorithm is very dependent on fine tuning; in this paper, we present such tuning to achieve better performance. Simultaneously, we describe a generalization of van Hoeij's algorithm to factor polynomials over number fields.

Key-words: polynomial factorization, lattice basis reduction, LLL algorithm, factorization over number fields

* Département de mathématiques, Université Paris XI, F-91405 ORSAY Cedex, e-mail : `Karim.BELABAS@math.u-psud.fr`

† e-mail : `Guillaume.Hanrot@loria.fr`

‡ e-mail : `Paul.Zimmermann@loria.fr`

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,
615, rue du Jardin Botanique, BP 101, 54602 Villers-Lès-Nancy (France)
Téléphone : +33 3 83 59 30 00 — Télécopie : +33 3 83 27 83 19

Implantation et généralisation de l'algorithme de van Hoeij

Résumé : Van Hoeij a proposé récemment un nouvel algorithme de factorisation de polynômes à coefficients entiers [11]. Cet algorithme repose sur le même principe que l'algorithme de Berlekamp-Zassenhaus [2, 13], mais utilise des techniques de réduction des réseaux pour améliorer drastiquement la seconde phase. L'efficacité de l'algorithme LLL dépend fortement des paramètres d'entrée et de réglages variés. Nous présentons dans cet articles diverses modifications mineures permettant d'obtenir de meilleurs performances. En outre, on décrit une généralisation de l'algorithme de van Hoeij permettant de factoriser des polynômes sur des corps de nombres.

Mots-clés : factorisation polynomiale, réduction des réseaux, algorithme LLL, factorisation sur les corps de nombres

Introduction

Until 2000, the main two algorithms known for factoring a polynomial P over $\mathbb{Z}[X]$ were Berlekamp-Zassenhaus algorithm, which starts by factoring P over $\mathbb{Q}_p[X]$ for suitable p and tries to recombine the p-adic factors, and Lenstra-Lenstra-Lovász algorithm, based on their famous LLL lattice reduction algorithm [5], which tries to determine the minimal polynomial of a p-adic root of P. While the latter is polynomial and the former exponential in the worst-case (P can have as much as $\Omega(\deg(P))$ factors over $\mathbb{Q}_p$, which can lead to an exponential time for the recombination), the former performs far better in practice. Lately, van Hoeij published an algorithm which, while following Berlekamp-Zassenhaus argument, uses lattice basis reduction to try to guess the correct recombination. The main idea is the fact that if (G_i) are the p-adic factors of P, and if $\prod_i G_i^{\varepsilon_i} = G_\varepsilon$ is a rational factor, then any polynomial in the coefficients of G_ε will be integers. Hence, they will be small compared with the order of magnitude of the same polynomial evaluated for a non-rational recombination. Van Hoeij's idea is to use the *Newton sums* $S_j(G) = \sum_{G(\alpha)=0} \alpha^j$, which are linear in the ε_i. The problem of finding a valid recombination is thus reduced to finding simultaneous small values of linear forms with integer coefficients (or alternatively solving a knapsack problem), one of the very situations where the LLL algorithm is known to be of interest.

In the first section, we recall some of the technicalities of van Hoeij's algorithm. The second section requires some tuning and experiments with this algorithm. The third section describes a generalization of van Hoeij's algorithm to number fields.

1 Van Hoeij's algorithm

Let P be a polynomial. Recall that the k-th Newton sum is defined as $S_k(P) = \sum_l \alpha_l^k$, where the α_l run through the roots of P. A trivial consequence of this definition is the fact that $S_k(PQ) = S_k(P) + S_k(Q)$. Furthermore, as a symmetric function of the roots, it can be seen that $S_k(P)$ belongs to the same field as the coefficients of P ; if P is monic with algebraic integer coefficients, then $S_k(P)$ is an algebraic integer.

Let p be a prime not dividing the discriminant of P, and let (G_i), $1 \le i \le n$, be the p-adic factors of P. We are looking for the set of the vectors ε_i such that $P_\varepsilon = \prod_i G_i^{\varepsilon_i}$ has itself integer coefficients. In practice, if the coefficients of the G_i are known to the precision p^ℓ with ℓ large enough, this will usually mean that the coefficients of $\prod_i G_i^{\varepsilon_i}$ are small with respect to p^ℓ (in any case, they should not vary with ℓ when ℓ

is large enough). As a consequence, the Newton sums $S_k(G_\varepsilon) = \sum_i \varepsilon_i S_k(G_i)$ should be close to a multiple of p^ℓ. This means that we are looking for $\{0,1\}$ vectors (ε_i) such that we have

$$\sum_{i=1}^{n} \varepsilon_i S_k(G_i) + \lambda p^\ell + \mu = 0, \tag{1}$$

for a small μ which can be estimated by using bounds on the coefficients of a factor of P (see van Hoeij's paper). Note that in practice, we are not using p-adic values $S_k(G_i)$ but rather truncated integer approximations with of course precision greater than p^ℓ; those truncated values to the precision b are defined as

$$\frac{S_k(G_i) - (S_k(G_i) \bmod p^b)}{p^b} \bmod p^{a-b},$$

where b is such that p^b is larger than the bound on the coefficients. Note that $x \bmod y$ is to be understood as: the integer congruent to x modulo y which is in the interval $]-y/2, y/2]$.

Identity (1) can actually be seen as a knapsack problem.

Van Hoeij's method is to see that the set of rational integer vectors (ε_i) solution of (1) is a sublattice of $\mathbb{Z}^n$, and solving our initial problem amounts to find a basis of this sublattice.

To achieve this, van Hoeij's applies an iterative process. Let

$$M = \begin{pmatrix} C\mathrm{Id}_n & 0 \\ S & p^\ell \mathrm{Id}_s \end{pmatrix}, \text{ where } S := \begin{pmatrix} S_{i_1}G_1 \dots & S_{i_1}G_n \\ \vdots \quad \vdots & \vdots \\ S_{i_k}G_1 \dots & S_{i_k}G_n \end{pmatrix}, \tag{2}$$

C is a suitable integer constant, and q be the quadratic form defined by the matrix M^tM. Then we are looking for vectors of the lattice $(\mathbb{Z}^n, q)$ of norm smaller than some bound B. We can now compute a LLL reduced basis of this lattice, which has the property to be close to orthogonal, and discard part of this basis by using the following classical

Lemma 1.1 *Let (b_i) be a LLL-reduced basis with respect to the positive quadratic form q. If $q(b_j) > B$ for $j \geq j_0$, then if v has $q(v) \leq B$, v is in the subspace $(b_1, \dots, b_{j_0-1})$.*

Note that the choice of M, and hence of q, depends on the parameter k and on the choice of the subset of Newton sums S_{i_k}. Repeating this process while varying the Newton sums and/or the precision usually decreases rather quickly the size of the lattice under consideration.

To check whether we are done, we put the basis of our lattice in row echelon form, and try to find vectors with $\{0, 1\}$ coordinates. If we can achieve this and furthermore have exactly one 1 in each row, we presumably have found out a factorization of P, which can be checked as in the usual Berlekamp-Zassenhaus process (multiply the factors, check for each coefficient if it is smaller than Mignotte's bound, and finally try to divide P by the putative factor).

2 Tuning van Hoeij's algorithm

Let us first describe a slightly modified version of van Hoeij's original algorithm. In this version, each of the successive lattices L_k that occur is embedded in $\mathbb{Z}^r$ where $r = \dim L_k$, instead of remaining embedded into $\mathbb{Z}^n$.

Algorithm SearchTrueFactors.
Input: a list of n_0 modular factors.
Output: the factors over the rationals.
Initialization: $s = 1$, $n = n_0$, $r = n_0$, BL the identity $n_0 \times n_0$ matrix

done $\leftarrow$ *false*; **repeat**

Step 1 [construction of the LLL input matrix]: M is an $(r + s)$ square matrix, as in (2), where S is the product of the $s \times n_0$ matrix of traces up to degree s by BL (of dimension $n_0 \times r$).
Step 2 [lattice reduction]: LLL-reduce (in place) the $(r + s)$ by $(r + s)$ matrix M (of rank $r + s$)
Step 3 [back to input factors]: replace the upper $r \times (r + s)$ submatrix L of M by $BL \times L$; M is now of dimension $(n_0 + s) \times (r + s)$, with rank $r + s$.
Step 4 [Gram-Schmidt]:perform Gram-Schmidt orthogonalization on M. Let $c_i, 1 \le i \le r + s$ be the norm of the ith orthogonal vector.
Step 5: Let r' the largest value such that all c_i of index larger than r' are of norm greater than

$$\sqrt{C^2 n_0 + s n_0^2/4}.$$

Step 6: if $r' = 1$, return "irreducible".

Step 7 [update BL]: let M' the $r \times r'$ matrix whose entries are those of the upper left part of M (after Step 2) divided by C (the division is exact). Replace BL by $BL \times M'$; BL is now of dimension $n_0 \times r'$. BL can also be obtained from the left r' columns of the matrix $BL \times L$ computed in Step 3, dividing the coefficients by C.

Step 8: put BL in Gauss-Jordan form. If, possibly after division by a constant factor, each column contains only 0 and 1, and each row contains exactly one 1, we might have a valid factorization, done $\leftarrow$ `true`. Otherwise, increase s, replace n by r and r by r'.

until done.

The main feature of the present version is that the dimension of the matrix M decreases quite rapidly (it is of dimension $r + s$ instead of remaining of dimension $n_0 + s$), which induces much faster LLL reductions.

We implemented the above variant in the NTL library version 5.0c [10], which was configured to use GNU MP 3.1.1 as long integer library [4]. Our implementation uses all-integer lattice reduction (the `LLL` routine from NTL). In addition, we used the following strategies and parameters:

- the initial value p^{a-b} was chosen to have about three times as many bits as the number n_0 of modular factors;
- the number s of traces considered at Step 1 is initially one. If the new dimension r' of the matrix BL is not smaller than r, s increases by one, and $a-b$ increases too.
- the precision used for Gram-Schmidt orthogonalization of M in Step 4 equals that of p^{a-b}.

Our NTL implementation is freely available from `http://www.loria.fr/~zimmerma/free/ZZXFactoring.c`. We obtained the following timings on a 1Ghz Athlon under Linux (`lucrezia.medicis.polytechnique.fr`): columns Deg, Dig and n_0 give respectively the degree of the corresponding polynomial, the number of digits of its largest coefficient, and the number of p-adic factors; columns Lift, Knap and LLL give respectively the setup time — small prime factorization and Hensel lifting —, the total recombination time using van Hoeij's algorithm, and the part spent in LLL reductions.

Pol.	Deg	Dig	n_0	Lift	Knap	LLL
P_4	462	756	42	13.2	2.3	0.8
P_5	64	40	32	0.13	0.16	0.09
P_7	384	57	76	4.14	1.98	1.56
P_8	972	213	54	23.5	0.9	0.5
$M_{12,5}$	792	2813	72	44.2	42.0	40.7
$M_{12,6}$	924	3937	84	73.5	96.5	76.1
S_7	128	87	64	0.43	1.93	1.36
S_8	256	188	128	1.89	31.8	25.8
S_9	512	402	256	7.6	621.	546.
S_{10}	1024	854	512	40.9	14440	13232
$S_{6,7}$	192	127	96	1.17	9.76	7.38
$S_{7,9}$	640	490	320	15.4	1696	1511
$S_{8,9}$	768	590	384	20.7	3579	3193

The polynomials P_4 to P_8 come from [14]; P_4 has two factors of degree 66 and 396, P_5 to P_8 are irreducible. $M_{12,5}$ and $M_{12,6}$ are the 5th and 6th resolvents of the polynomial f with Galois group M_{12} from Section 3.2 of van Hoeij's paper [11]; $M_{12,5}$ is irreducible whereas $M_{12,6}$ has two factors of degree 132 and 792; these polynomials are useful to check a particular implementation since they are non-monic. S_n is the Swinnerton-Dyer polynomial of degree 2^n corresponding to the n first primes, and is irreducible; $S_{n,k}$ is the product of the Swinnerton-Dyer polynomials of order n and k.

The dimension decrease seems quite effective: for S_8, the first matrix to reduce is of dimension 129, then it goes down to dimension 125, 113, 85, 47, 19, 7 and finally 3.

The polynomial P_8 was impossible to factor two years ago. At the last ISSAC conference, Abbott, Shoup and Zimmermann managed to prove its irreducibility using large tables to speed up Berlekamp-Zassenhaus searching phase [1]; it took however of the order of one hour of cpu time. Now, thanks to van Hoeij's algorithm, the searching phase takes less than one second for that polynomial, and the most time consuming phase is now Hensel lifting. The polynomial S_{10} is the first example of a proof of irreducibility using the Berlekamp-Zassenhaus strategy for a polynomial with at least 512 modular factors (for all p).

Comparing the Knap and LLL columns reveals that most of the recombination time is now spent in lattice reduction. It would thus be of interest to combine our implementation with the new algorithm from Koy and Schnorr [12], to see how it speeds up the LLL part.

3 A variant over number fields

In this section, we assume that a finite extension $\mathbb{K}$ of $\mathbb{Q}$ (a number field) is given¸ and denote by $\mathcal{O}_{\mathbb{K}}$ its ring of integers. By the primitive element theorem, we can write $\mathbb{K} = \mathbb{Q}(\alpha)$, where $h(\alpha) = 0$ with $h \in \mathbb{Z}[Y]$, of degree d. We can choose $\alpha \in \mathcal{O}_{\mathbb{K}}$, i.e. h monic. This implies that $\mathbb{Z}[\alpha] = \mathbb{Z}[Y]/(h)$ is a submodule of $\mathcal{O}_{\mathbb{K}}$ of finite index. In the sequel, we shall represent an element z of $\mathcal{O}_{\mathbb{K}}$ as $z = \sum_{0 \le i < d} z_i \alpha^i$, where $f z_i \in \mathbb{Z}$. We do not insist that f be minimal $(= [\mathcal{O}_{\mathbb{K}} : \mathbb{Z}[\alpha]])$, in order to avoid computing an integral basis for $\mathbb{K}$: while well understood (it amounts to factoring h over $\mathbb{Q}_p$ for all primes p whose square divides $\mathrm{disc}(h)$), this process remains quite time-consuming and involves factoring the discriminant of h. In contrast, a multiple f of the index can be easily computed, for instance we can take for f any multiplicative bound for the largest integer f_0 such that f_0^2 divides $\mathrm{disc}(h)$, .e.g $\mathrm{disc}(h)$ itself after weeding out some small prime divisors. It is worthwhile to try to obtain a somewhat minimal f, hence to choose suitably the polynomial h. For how to perform this task and all the other basic algorithms for number fields (e.g. decomposition of primes, "polynomial reduction" algorithms), the reader is referred to [3, 7, 8].

Let also P be a polynomial of degree n with coefficients in $\mathbb{K}$. For simplicity, we will assume that P is monic and has coefficients in $\mathbb{Z}[\alpha]$, which can always be achieved by a change of variables. We write $P = X^n + \sum_{i=0}^{n-1} p_i X^i$, with $p_i = \sum_{j=0}^{d-1} p_{ij} Y^j \in \mathbb{Z}[Y]/(h)$ in "reduced form". We shall further assume that P has no square factors, which can be achieved by square free factorization.

As any other factoring algorithm over $\mathbb{Q}$, van Hoeij's method can be used to factor P over $\mathbb{K}$, by factoring the norm $\mathcal{P}_\lambda := \mathrm{res}_Y(P(X - \lambda Y), h(Y)) \in \mathbb{Z}[X]$, where the p_i are lifted to $\mathbb{Z}[Y]$ and λ is a small rational integer chosen so that $\mathcal{P}_\lambda$ is squarefree. Each $\mathbb{Q}$-factor G of $\mathcal{P}_\lambda$ is divisible by a unique $\mathbb{K}$-factor of P, which is extracted as $\gcd(G, P)$ over $K[X]$. Assuming efficient modular implementations for the initial resultants and final gcds, the main bottleneck with this reduction to the absolute case is the recombination phase during the factorization of $\mathcal{P}_\lambda$. Van Hoeij's algorithm is very suitable for this kind of polynomials, and Galois resolvents in general, since it is not too sensitive to the size of coefficients (besides the Hensel lifting phase), and is able to cope with the huge number of modular factors which are often intrinsic to the problem.

We now sketch an application of van Hoeij's ideas to the direct factorization in $\mathbb{K}[X]$, which is superior to the norm approach since the number of modular factors over $\mathbb{K}$ will usually be much lower than over $\mathbb{Q}$ (possibly by a factor of d), without increasing the difficulty of the other steps. We follow the approach of Roblot [9], but for the recombination, which we replace by van Hoeij's knapsack. Also contrary to Roblot, we don't insist that computations should be done in $\mathcal{O}_{\mathbb{K}}$, and are content with $\frac{1}{f}\mathbb{Z}[\alpha]$.

We first recall classical upper bounds on the coefficients of a monic factor $G = X^k + \sum_{i<k} g_i X^i$ of P in $\mathbb{K}[X]$. We measure the size of an element x of K in terms of the quadratic form $T_2(x) := \sum_\sigma |x^\sigma|^2$, where σ runs through the n embeddings of $\mathbb{K}$ into $\mathbb{C}$ and $x^\sigma := \sigma(x)$. We can derive a uniform bound on the coefficients of G by

Lemma 3.1 *Let $G = \sum_{i\leq k} g_i X^i$ be a monic divisor of P as above. Then all the g_i are integral and we have*

$$T_2(g_i) \leq T_2(P)\binom{k-1}{i}\left[\binom{k-1}{i}+2\binom{k-1}{i-1}\right] + d\binom{k-1}{i-1}^2$$

where $T_2(P) := \sum_{i\leq n} T_2(p_i)$.

Proof. Let $P = GH$ in $K[X]$, then

$$\mathrm{cont}(P) = \mathrm{cont}(G)\mathrm{cont}(H),$$

where $\mathrm{cont}(A)$ denotes the fractional ideal generated by the coefficients of $A \in \mathbb{K}[X]$. Since $P \in \mathcal{O}_{\mathbb{K}}[X]$ is monic, we have $\mathrm{cont}(P) = \mathcal{O}_{\mathbb{K}}$. Since G is monic, so is H, and their contents both contain $\mathcal{O}_{\mathbb{K}}$. It follows that $\mathrm{cont}(G) = \mathrm{cont}(H) = \mathcal{O}_{\mathbb{K}}$, hence all g_i belong to $\mathcal{O}_{\mathbb{K}}$.

For each embedding $\sigma : \mathbb{K} \to \mathbb{C}$, the polynomial G^σ divides P^σ in $\mathbb{C}[X]$. Summing (the square of) Mignotte's bound for g_i^σ over the σ yields the T_2 bound. □

Let now p be a prime number not dividing the discriminant of h, hence unramified, and assume that p splits as $\prod_i \mathfrak{p}_i$, with $N\mathfrak{p}_i = p^{f_{\mathfrak{p}_i}}$. For matters of efficiency it is worth trying several primes p until one finds a $\mathfrak{p} \mid p$ such that $P \mod \mathfrak{p}$ is squarefree with few factors in $\mathcal{O}_{\mathbb{K}}/\mathfrak{p}$, and $\mathfrak{p}$ is of residual degree $f_{\mathfrak{p}}$ as small as possible. A good rule of thumb is to pick $\mathfrak{p}$ which minimizes the product of $f_{\mathfrak{p}}$ by the number of modular factors. Note that when $\mathbb{K}/\mathbb{Q}$ is Galois, it is not difficult to find totally split primes (probability $1/d$ by Chebotarëv density theorem); this is not the case for a "general" number field (where the probability is only expected to be larger than $1/d!$).

The following step is to lift the factorization over $\mathbb{K}_\mathfrak{p}$ (the $\mathfrak{p}$-adic completion of $\mathbb{K}$); write $P = \prod_{k=0}^{n_0} G_k$. The factors G_i should be computed modulo $\mathfrak{p}^\ell$, where ℓ is large enough to enable reconstructing a genuine factor from its modular approximation, given bounds on the size of its coefficients. More precisely

Lemma 3.2 (Lenstra [6]) *Let $\gamma \in \mathbb{Z}[\alpha]$, and assume that ℓ is so large that*

$$\ell\left(\frac{2}{d} f_\mathfrak{p} \log(p)\right) > \log(C/d) + \left(\frac{d(d-1)}{4} + 1\right) \log 4$$

then there is at most one $\beta \in \mathbb{Z}[\alpha]$ such that $\beta \equiv \gamma \pmod{\mathfrak{p}^\ell}$ and $T_2(\beta) \le C$.

Let $M := M_{\mathfrak{p}^\ell}$ be the matrix (on the fixed basis $1, \dots, \alpha^{d-1}$) of an LLL-reduced basis of the lattice $(\mathfrak{p}^\ell \cap \mathbb{Z}[\alpha], T_2)$. If it exists, β equals $\gamma - M\lceil M^{-1}\gamma\rfloor$, where we still denote γ the vector giving γ on the $\mathbb{Z}[\alpha]$-basis. As usual, $\lceil x \rfloor$ is the operator rounding to nearest integer ($:= \lfloor x + 1/2 \rfloor$), and is to be applied coordinatewise.

Proof. Roblot states this result with $\mathbb{Z}[\alpha]$ replaced by $\mathcal{O}_\mathbb{K}$ but Lenstra's argument [6] applies mutatis mutandis. □

From a practical viewpoint, $\mathfrak{p}^\ell \cap \mathbb{Z}[\alpha]$ is generated by p^ℓ and $U(\alpha)^\ell$ over $\mathbb{Z}[\alpha]$, where U is the symmetric lift in $\mathbb{Z}[X]$ of the factor of h modulo p corresponding to $\mathfrak{p}$. It is easy to compute a $\mathbb{Z}$-basis of this lattice, via an HNF reduction modulo p^ℓ of the matrix giving the multiplication by $U(\alpha)^\ell$ in $\mathbb{Z}[\alpha]$. Since p is unramified, it is a uniformizer in $\mathbb{K}_\mathfrak{p} \simeq \mathbb{Q}_p[Y]/(\tilde{U})$, where $\tilde{U}$ is the $\mathbb{Q}_p$-factor of h reducing to U modulo p. The polynomial $\tilde{U}$ is monic and has degree $f_\mathfrak{p}$; any integral element of $\mathbb{K}_\mathfrak{p}$ (in $\mathfrak{p}$-adic precision ℓ) is written in reduced form as

$$\sum_{0 \le i < f_\mathfrak{p}} \lambda_i Y^i, \tag{3}$$

where λ_i belongs to $]-p^\ell/2, p^\ell/2]$.

It only remains to introduce the adapted knapsack problem. The Newton sums are computed from the Newton formulae as in the absolute case, and are easily bounded

Lemma 3.3 *Assume that the monic $\mathfrak{p}$-adic factor G of P is in fact in $\mathcal{O}_\mathbb{K}[X]$, then for all integer k, the Newton sum $S_k(G)$ is in $\mathcal{O}_\mathbb{K}$ and for any embedding σ, we have*

$$|S_k(G)^\sigma| \le n B_\sigma^k$$

where B_σ is any bound for the modulus of the complex roots of P^σ.

Proof. The bound on $S_k(G)^\sigma$ is obvious. Since G is monic, the Newton sums are integral combinations of the coefficients of G, hence integral. □

Let V be the Vandermonde matrix associated to the complex roots (α^σ) of h, and $\|\cdot\|$ denote the matrix norm given by

$$\|(m_{ij})\| := \max_j \sum_i |m_{ij}|$$

We can now bound the coordinates of the Newton sums on the (α^i)

Lemma 3.4 *The coordinates of $fS_k(G)$ on (α^i) are bounded by*

$$C_k := fn\|V^{-1}\|(\max_\sigma B_\sigma)^k$$

Proof. Let $x = \sum_{i<d} x_i\alpha^i$, $x_i \in \mathbb{Q}$; writing the d different embeddings of this equation in $\mathbb{C}$, we obtain

$$(x^\sigma) = (x_i)V, \text{ hence } (x_i) = (x^\sigma)V^{-1}$$

and

$$\max_i |x_i| \leq \max_\sigma |x^\sigma| \cdot \|V^{-1}\|.$$

Now we apply the previous lemma. Note that V^{-1} is easily evaluated from Lagrange interpolation formula since its i-th column gives the coefficients of $E_i \in \mathbb{C}[X]$ such that $E_i(\alpha^{\sigma_j}) = \delta_{ij}$ for all j. □

The truncated traces associated to a $\mathfrak{p}$-adic factor G can be defined by

$$S_k^{b,a}(G) := \frac{S_k(G) - (S_k(G) \mod \mathfrak{p}^b)}{p^b} \pmod{\mathfrak{p}^{a-b}}$$

which is easily computed in the representation (3) by applying van Hoeij's truncation, as seen in Section 1, to each of the λ_i. Assuming that $p^\ell > 2C_k$, which will almost certainly be the case if the reconstruction bounds from Lemma 3.2 hold, then we can choose $b < a \leq \ell$ such that $S_k^{b,a}(G)$ is 0 for any $\mathbb{K}$-rational factor G. The next knapsack-type lemma should be compared with van Hoeij's Lemma 2.6.

Lemma 3.5 *Assume that $\prod_{k=0}^{n_0} G_i^{e_i} \in \mathbb{K}[X]$. Then for all k, one has*

$$\varepsilon + \sum e_i S_k^{b,a}(G_i) = 0$$

where the coordinates of $f\varepsilon$ on (α^i) are integral, bounded by $f|S|/2$, and equal to 0 if $i \geq f_\mathfrak{p}$.

Proof. Follows the same lines as van Hoeij's, left to the reader. □

Now the matrix defining the quadratic form for the LLL reduction is

$$M = \begin{pmatrix} \mathrm{Id}_n & 0 \\ S & \mathfrak{p}_k^* \end{pmatrix}$$

where S is as in the first section except the $S_i(P_j)$ must be interpreted as the column vector giving $S_i^{b_i,a}(P_j)$ in terms of $(1, \ldots, Y^{f_\mathfrak{p}-1})$ and $\mathfrak{p}_k^*$ is a block-diagonal matrix, where the $f_\mathfrak{p} \times f_\mathfrak{p}$ diagonal blocks are equal to the upper left part of $M_{\mathfrak{p}^{a-b_i}}$, $i = 1, \ldots, k$, which is assumed to be in upper HNF form. Since typically b_i will be chosen so that $a - b_i$ is small, very few different $M_{\mathfrak{p}^j}$ need to be computed. The size of the matrix M is $n + kf_\mathfrak{p}$.

Once this matrix M is computed, the computation run as in Section 2.

4 Acknowledgement.

The authors are grateful to Mark van Hoeij for several discussions about his algorithm, to Victor Shoup for his help with NTL, and to Allan Steel for providing the polynomials $M_{12,5}$ and $M_{12,6}$, together with some nice timings obtained with Magma, which stimulated our NTL implementation. Section 2 timings were obtained thanks to the *Unité Mixte de Service* Médicis (`medicis.polytechnique.fr`).

References

[1] Abbott, J., Shoup, V., and Zimmermann, P. Factorization in $\mathbb{Z}[X]$: the searching phase. In *Proceedings of ISSAC'2000* (2000), C. Traverso, Ed., ACM Press, pp. 1–7. `http://www.shoup.net/papers/asz.ps.Z`.

[2] Berlekamp, E. R. Factoring polynomials over large finite fields. *Mathematics of Computation 24*, 111 (1970), 713–735.

[3] Cohen, H. *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics 138. Springer-Verlag, 1993.

[4] Granlund, T. *GNU MP: The GNU Multiple Precision Arithmetic Library*, 3.1.1 ed., 2000. Available from `http://www.swox.se/gmp/`.

[5] Lenstra, A. K., Lenstra, H. W., Lovász, L. Factoring polynomials with rational coefficients, *Math. Ann.* **261** (1982), 515–534.

[6] LENSTRA, A. K. Factoring polynomials over algebraic number fields *LN in Comp. Sci. 144* (1982), 32–39.

[7] POHST, M. *Computational Algebraic Number Theory*, vol. 21 of *DMV Seminar*. Birkhäuser, Basel, 1993.

[8] POHST, M., AND ZASSENHAUS, H. *Algorithmic Algebraic Number Theory*. Cambridge University Press, 1989.

[9] ROBLOT, X. *Algorithmes de factorisation dans les extensions relatives et applications de la conjecture de Stark à la construction des corps de classes de rayon*. PhD thesis, Bordeaux, 1997.

[10] SHOUP, V. NTL: A library for doing number theory. `http://www.shoup.net/ntl/`.

[11] VAN HOEIJ, M. Factoring polynomials and the knapsack problem. *Journal of Number Theory* (2000). To appear. Also available at `http://www.math.fsu.edu/~hoeij/papers.html`.

[12] KOY, H. AND SCHNORR, C. P. Segment LLL-Reduction of Lattice Bases. Preprint, 2001. Available at `http://www.mi.informatik.uni-frankfurt.de/research/papers.html`

[13] ZASSENHAUS, H. On Hensel factorization I. *Journal of Number Theory 1* (1969), 291–311.

[14] ZIMMERMANN, P. Polynomial factorization challenges: a collection of polynomials difficult to factor. `http://www.loria.fr/~zimmerma/mupad/`.

Unité de recherche INRIA Lorraine
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399