FACTORING MULTIVARIATE INTEGRAL POLYNOMIALS

A.K. LENSTRA

Centre for Mathematics and Computer Science, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands

Abstract. We present an algorithm to factorize polynomials in several variables with integral coefficients that is polynomial-time in the degrees of the polynomial to be factored, for any fixed number of variables. Our algorithm generalizes the algorithm presented by Lenstra, Lenstra Jr. and Lovász (1982) to factorize integral polynomials in one variable.

1. Introduction

The problem of factoring polynomials with integral coefficients remained open for a long time, i.e., no polynomial-time algorithm was known. The best known algorithms took exponential-time in the worst case; these algorithms had to consider a possibly exponential number of combinations of p-adic factors before the true factors could be found or irreducibility could be decided. In [1] it was proven that the problem of factorization in $\mathbb{Z}[X]$ belongs to NP$\cap$co-NP, which made its membership of P quite likely [2]. That this was indeed the case, was proven in [7] where a polynomial-time algorithm for factoring in $\mathbb{Z}[X]$ was given. This algorithm is based on three observations:

(1.1) The multiples of degree $<m$ of a p-adic factor together form a lattice in $\mathbb{Z}^m$.

(1.2) If this p-adic factor is computed up to a high enough precision, then the factor we are looking for is the shortest vector in the lattice.

(1.3) An approximation of the shortest vector in such a lattice can be found by means of the so-called *basis reduction algorithm*.

In this paper we show that (1.1) and (1.2) can be generalized to polynomials in $\mathbb{Z}[X_1, X_2, \ldots, X_t]$ in an elementary way, for any $t \geq 2$. Combined with the same basis reduction algorithm as in (1.3), this leads to a polynomial-time algorithm for factoring in $\mathbb{Z}[X_1, X_2, \ldots, X_t]$. In [8] the present author shows that the above three points can be applied to various other kinds of polynomial factoring problems as well (like multivariate polynomials over finite fields or over algebraic number fields).

Another approach to multivariate integral polynomial factorization is given in [5]. There, the multivariate case is reduced in polynomial-time to the univariate case, and the univariate case is handled by the algorithm from [7].

For practical purposes we do not recommend any of these polynomial-time algorithms; their running time will be dominated by the rather slow basis reduction algorithm. For polynomials in $\mathbb{Z}[X_1, X_2, \ldots, X_t]$ the algorithm from [10] for instance is very useful, although it is exponential-time in the worst case.

We restrict ourselves in this article to integral polynomials in two variables; the multivariate case immediately follows from this. In Section 2 we present an important result from [7, Section 1] concerning the basis reduction algorithm mentioned in (1.3). The generalizations of (1.1) and (1.2) to polynomials in $\mathbb{Z}[X, Y]$ are described in Section 3, and in Section 4 we give an outline of the factoring algorithm, and we analyse its running time.

2. The basis reduction algorithm

The basis reduction algorithm from [7, Section 1] makes it possible to determine in polynomial-time a reasonable approximation of the shortest vector in a lattice. We will not give a description of the algorithm here. It will suffice to summarize those results from [7, Section 1] that we will need here.

Let $b_1, b_2, \ldots, b_n \in \mathbb{Z}^n$ be linearly independent. For our purposes we may assume that the $n \times n$ matrix having $b_1, b_2, \ldots, b_n$ as columns is upper-triangular. The i-dimensional *lattice* $L_i \subset \mathbb{Z}^i$ with basis $b_1, b_2, \ldots, b_i$ is defined as $L_i = \sum_{j=1}^{i} \mathbb{Z} b_j = \{\sum_{j=1}^{i} r_j b_j : r_j \in \mathbb{Z}\}$. We put $L = L_n$.

2.1. Proposition (cf. [7, (1.11), (1.26), (1.37)]). *Let $B \in \mathbb{Z}_{\geq 2}$ be such that $|b_j|^2 \leq B$ for $1 \leq j \leq n$, where $|\;|$ denotes the ordinary Euclidean length. The basis reduction algorithm as described in [7, (1.15)] determines a vector $\mathbf{b} \in L$ such that $\mathbf{b}$ belongs to a basis for L, and such that $|b|^2 \leq 2^{n-1}|x|^2$ for every $x \in L$, $x \neq 0$; the algorithm takes* $O(n^4 \log B)$ *elementary operations on integers having binary length* $O(n \log B)$*. Furthermore, during the first* $O(i^4 \log B)$ *operations (on integers having binary length* $O(i \log B)$*), vectors $\mathbf{b}_i \in L_i$ are determined such that $|\mathbf{b}_i|^2 \leq 2^{i-1}|x_i|^2$ for every $x_i \in L_i$, $x_i \neq 0$, for $1 \leq i \leq n$.*

So, we can find a reasonable approximation of the shortest vector in L in polynomial-time. But also we find, during this computation, approximations of the shortest vectors of the lattices L_i without any time loss.

3. Factors and lattices

We describe how to generalize (1.1) and (1.2) to polynomials in $\mathbb{Z}[X, Y]$. Let $f \in \mathbb{Z}[X, Y]$ be the polynomial to be factored; we may assume that f has no multiple factors, i.e., f is *square-free*. Furthermore we assume that f is *primitive* with respect to X, i.e., the greatest common divisor of the coefficients in $\mathbb{Z}[Y]$ of f equals one. We denote by $\delta_X f$ and $\delta_Y f$ the degrees of f in X and Y respectively, and by $\mathrm{lc}(f)$ the *leading coefficient* of f with respect to X. We put $n_X = \delta_X f$ and $n_Y = \delta_Y f$.

Suppose that we are given a prime number p, an integer s and a positive integer k. By (p^k, S_l) we denote the ideal generated by p^k and $(Y-s)^{l+1}$, for some integer

$l \geqslant 0$. In Section 4 we will see how to find a polynomial $h \in \mathbb{Z}[X, Y]$ such that:

(3.1) $\operatorname{lc}(h) = 1$,
(3.2) $(h \bmod (p^k, S_{n_Y}))$ divides $(f \bmod (p^k, S_{n_Y}))$ in $\mathbb{Z}[X, Y]/(p^k, S_{n_Y})$,
(3.3) $(h \bmod (p, S_0)) \in (\mathbb{Z}/p\mathbb{Z})[X]$ is irreducible in $(\mathbb{Z}/p\mathbb{Z})[X]$,
(3.4) $(h \bmod (p, S_0))^2$ does not divide $(f \bmod (p, S_0))$ in $(\mathbb{Z}/p\mathbb{Z})[X]$.

We put $l = \delta_X h$; so $0 < l \leqslant n_X$.

Let $h_0 \in \mathbb{Z}[X, Y]$ be the up to sign unique irreducible factor of f for which $(h \bmod (p, S_0))$ divides $(h_0 \bmod (p, S_0))$ in $(\mathbb{Z}/p\mathbb{Z})[X]$ (or equivalently $(h \bmod (p^k, S_{n_Y}))$ divides $(h_0 \bmod (p^k, S_{n_Y}))$ in $\mathbb{Z}[X, Y]/(p^k, S_{n_Y})$, cf. [7, (2.5)]).

(3.5). Let m_X and m_Y be two integers with $l \leqslant m_X < n_X$ and $0 \leqslant m_Y \leqslant \delta_Y \operatorname{lc}(f)$. We define L as the collection of polynomials $g \in \mathbb{Z}[X, Y]$ such that

(i) $\delta_X g \leqslant m_X$,
(ii) $\delta_Y g \leqslant n_Y$,
(iii) $\delta_Y \operatorname{lc}(g) \leqslant m_Y$,
(iv) $(h \bmod (p^k, S_{n_Y}))$ divides $(g \bmod (p^k, S_{n_Y}))$ in $\mathbb{Z}[X, Y]/(p^k, S_{n_Y})$.

Putting $M = m_X(n_Y + 1) + m_Y + 1$ it is not difficult to see that L is an M-dimensional lattice contained in $\mathbb{Z}^M$, where we identify polynomials in L and M-dimensional vectors in the usual way (i.e., $\sum_{i=0}^{m_X-1} \sum_{j=0}^{n_Y} a_{ij} X^i Y^j + \sum_{j=0}^{m_Y} a_{m_X j} X^{m_X} Y^j$ is identified with $(a_{00}, a_{01}, \ldots, a_{0n_Y}, a_{10}, \ldots, a_{m_X-1\, n_Y}, a_{m_X 0}, \ldots, a_{m_X m_Y})$). Because of (3.1) a basis for L is given by

$$\{p^k X^i Y^j : 0 \leqslant i < l, 0 \leqslant j \leqslant n_Y\} \cup$$
$$\cup \{((h \bmod (p^k, S_{n_Y}) X^{i-l} Y^j) \bmod (p^k, S_{n_Y})) : (l \leqslant i < m_X \text{ and } 0 \leqslant j \leqslant n_Y) \text{ or } (i = m_X \text{ and } 0 \leqslant j \leqslant m_Y)\}.$$

This generalizes (1.1) (cf. [7, (2.6)]). We now come to (1.2). The *height* $g_{\max}$ of a polynomial g is defined as the maximal absolute value of any of its integral coefficients. We prove that, if k and s are suitably chosen, then a vector of small height in L must lead to a factorization of f.

3.6. Proposition. *Suppose that $g \in L$ satisfies*

(3.7) $$|s|^{n_Y+1} > (e^{n_X+n_Y} f_{\max} \sqrt{(n_X+1)(n_Y+1)})^{m_X} (g_{\max} \sqrt{(m_X+1)(n_Y+1)})^{n_X}$$

and

(3.8) $$p^k > (e^{n_X+n_Y} f_{\max} \sqrt{(n_X+1)(n_Y+1)})^{m_X} (g_{\max} \sqrt{(m_X+1)(n_Y+1)})^{n_X} \times (1 + (1+|s|)^{n_Y+1})^{n_Y(n_X+m_X-1)}.$$

Then h_0 divides g, and in particular $\gcd(f, g) \neq 1$ *in* $\mathbb{Z}[X, Y]$.

Proof. Suppose that $\gcd(f, g) = 1$. This implies that the resultant $R \in \mathbb{Z}[Y]$ of f and

g is unequal to zero. Using the result from [4] we find that

$$(3.9)\quad |R| < \left(f_{\max}\sqrt{(n_X+1)(n_Y+1)}\right)^{m_X}\left(g_{\max}\sqrt{(m_X+1)(n_Y+1)}\right)^{n_X},$$

where $|R|$ denotes the Euclidean length of the vector identified with R. Since $(h \bmod (p^k, S_{n_Y}))$ divides both $(f \bmod (p^k, S_{n_Y}))$ and $(g \bmod (p^k, S_{n_Y}))$, the polynomials f and g have a nontrivial common divisor in $\mathbb{Z}[X, Y]/(p^k, S_{n_Y})$, so that R must be zero modulo (p^k, S_{n_Y}). The polynomial $(Y-s)^{n_Y+1}$ cannot divide R, because this would imply, with [9, Theorem 1], that $|s|^{n_Y+1} \leq |R|$, which is, combined with (3.9), a contradiction with (3.7). Therefore, $(R \bmod (Y-s)^{n_Y+1})$ has to be zero modulo p^k. Using induction on n_Y+1 we prove that

$$(R \bmod (Y-s)^{n_Y+1})_{\max} \leq R_{\max}(1+(1+|s|)^{n_Y+1})^{n_Y(n_X+m_X-1)},$$

so that, with $R_{\max} \leq |R|$ and (3.8), $(R \bmod (Y-s)^{n_Y+1})$ cannot be zero modulo p^k. We conclude that $\gcd(f, g) \neq 1$.

Suppose that h_0 does not divide g. So h_0 does not divide $r = \gcd(f, g)$, so that $(h \bmod (p^k, S_{n_Y}))$ divides $((f/r) \bmod (p^k, S_{n_Y}))$. Because f/r divides f, we find from [3] that $(f/r)_{\max} \leq e^{n_X+n_Y} f_{\max}$. This implies that the above reasoning applies to f/r and the same polynomial g in L, and we find $\gcd(f/r, g) \neq 1$. This is a contradiction with $r = \gcd(f, g)$, since f is square-free. □

3.10. Proposition. *Suppose that s and k are chosen in such a way that* (3.7) *and* (3.8) *are satisfied with $g_{\max}$ replaced by $2^{(M-1)/2}\sqrt{M}\, e^{n_X+n_Y} f_{\max}$. Let b be, as in Proposition* 2.1, *the result of an application of the basis reduction algorithm to the M-dimensional lattice L as defined in* (3.5). *Then $h_0 \in L$ if and only if* (3.7) *and* (3.8) *are satisfied with g replaced by b.*

Proof. To prove the 'if'-part, assume that (3.7) and (3.8) hold with $g_{\max}$ replaced by $b_{\max}$. According to Proposition 3.6 this implies that h_0 divides b, so that $h_0 \in L$.

To prove the 'only if'-part, assume that $h_0 \in L$. Because h_0 divides f, we find from [3] that $(h_0)_{\max} \leq e^{n_X+n_Y} f_{\max}$. So there exists a nonzero vector in L with Euclidean length bounded by $\sqrt{M}\, e^{n_X+n_Y} f_{\max}$. Application of (2.1) yields that $b_{\max} \leq |b| \leq 2^{(M-1)/2}\sqrt{M}\, e^{n_X+n_Y} f_{\max}$. Combined with the above choices of s and k, this implies that (3.7) and (3.8) hold with g replaced by b. □

4. Description of the algorithm

In this section we present the polynomial-time algorithm to factorize f. First we give an algorithm to determine the factor h_0, given p, s and h. After that, we will see how p and s have to be chosen.

(4.1). Let p, s and h be as in Section 3, such that (3.1), (3.3), (3.4) and (3.2) with k replaced by 1 are satisfied. Assume that s satisfies the condition in Proposition

3.10, with m_X and m_Y replaced by $n_X - 1$ and $\delta_Y \mathrm{lc}(f)$ respectively:

$$(4.2)\quad |s|^{n_Y+1} > \left(e^{n_X+n_Y} f_{\max}\sqrt{(n_X+1)(n_Y+1)}\right)^{n_X-1} \times \left(2^{(M-1)/2}\sqrt{M}e^{n_X+n_Y} f_{\max}\sqrt{n_X(n_Y+1)}\right)^{n_X},$$

where $M = (n_X - 1)(n_Y + 1) + \delta_Y \mathrm{lc}(f) + 1$.

We describe an algorithm that determines h_0, the up to sign unique irreducible factor of f such that (h mod(p, S_0)) divides (h_0 mod(p, S_0)) in $(\mathbb{Z}/p\mathbb{Z})[X]$.

We may assume that $l = \delta_X h < n_X$. Take k minimal such that the condition from Proposition 3.10 is satisfied, with m_X and m_Y replaced by $n_X - 1$ and $\delta_Y \mathrm{lc}(f)$ respectively:

$$(4.3)\quad p^k > \left(e^{n_X+n_Y} f_{\max}\sqrt{(n_X+1)(n_Y+1)}\right)^{n_X-1} \times \left(2^{(M-1)/2}\sqrt{M}\, e^{n_X+n_Y} f_{\max}\sqrt{n_X(n_Y+1)}\right)^{n_X}\left(1+(1+|s|)^{n_Y+1}\right)^{2n_Y(n_X-1)}.$$

Next modify h in such a way that (3.2) holds for this value of k; because of (3.4) this can be done by means of Hensel's lemma [11].

Apply Proposition 2.1 to the M-dimensional lattice L as defined in (3.5) for each of the values of $M = l(n_Y+1)+1, l(n_Y+1)+2, \ldots, l(n_Y+1)+\delta_Y \mathrm{lc}(f)+1, (l+1)(n_Y+1)+1, \ldots, (n_X-1)(n_Y+1)+\delta_Y \mathrm{lc}(f)+1$ in succession (so, for $m_X = l, l+1, \ldots, n_X - 1$ in succession and for every value of m_X the values $m_Y = 0, 1, \ldots, \delta_Y \mathrm{lc}(f)$ in succession). But stop as soon as a vector b is found satisfying (3.7) and (3.8) with g replaced by b.

If such a vector b is found for a certain value of M ($m_X = m_{X0}$ and $m_Y = m_{Y0}$), then we know from Proposition 3.10 that $h_0 \in L$. Since we try the values of M in succession, this implies that $\delta_X h_0 = m_{X0}$ and $\delta_Y \mathrm{lc}(h_0) = m_{Y0}$. By Proposition 3.6, h_0 divides b, so that $\delta_X b = m_{X0}$ and $\delta_Y \mathrm{lc}(b) = m_{Y0}$. So $b = ch_0$ for some $c \in \mathbb{Z}$, but $h_0 \in L$ and b belongs to a basis for L, so $b = \pm h_0$.

If we did not find such a vector b, then Proposition 3.10 implies that $\delta_X h_0 > n_X - 1$, so that $h_0 = f$, because f is primitive.

This finishes the description of algorithm (4.1).

4.4. Proposition. *Denote by $m_{X0} = \delta_X h_0$ the degree in X of the irreducible factor h_0 of f that is found by algorithm* (4.1). *Then the number of arithmetic operations needed by algorithm* (4.1) *is* $O(m_{X0}(n_X^5 n_Y^5 + n_X^4 n_Y^4 \log(f_{\max}) + n_X^4 n_Y^6 \log(|s|) + n_X^3 n_Y^4 \log p))$ *and the integers on which these operations have to be performed each have binary length* $O(n_X^3 n_Y^2 + n_X^2 n_Y \log(f_{\max}) + n_X^2 n_Y^3 \log(|s|) + n_X n_Y \log p)$.

Proof. Let M_1 be the largest value of M for which Proposition 2.1 is applied; so $M_1 = O(m_{X0} n_Y)$. It follows from Proposition 2.1 that the number of operations needed for the applications of the basis reduction algorithm for $l(n_Y+1)+1 \leq M \leq M_1$ is equal to the number of operations needed for $M = M_1$ only. Assuming that the coefficients of the initial basis for L are reduced modulo p^k, we find, using (4.3), that the following holds for the bound B on the length of these vectors:

$$\log B = O(n_X^2 n_Y + n_X \log(f_{max}) + n_X n_Y^2 \log(|s|) + \log p).$$

With $M_1 = O(m_{X0} n_Y)$ and Proposition 2.1 this gives the estimates in Proposition 4.4.

The verification that the same estimates are valid for the application of Hensel's lemma is straightforward [11]. □

We now describe how s and p have to be chosen. First, s must be chosen such that $(f \bmod (Y-s)) = f(X, s)$ remains square-free, and such that (4.2) holds. The resultant R of f and its derivative f' with respect to X is a nonzero polynomial in $\mathbb{Z}[Y]$ of degree $\leq n_Y(2n_X - 1)$. Therefore, we can find in $O(n_X n_Y)$ trials the minimal integer s such that s is not a zero of R, and such that (4.2) holds. It is easily verified that $\log(|s|) = O(n_X^2 + n_X \log(f_{max}))$.

Next we choose p as the smallest prime number not dividing the resultant of $f(X, s)$ and $f'(X, s)$. Since $\log(f(X, s)_{max}) = O(n_X^2 n_Y + n_X n_Y \log(f_{max}))$, it follows as in the proof of [7, (3.6)] that $p = O(n_X^3 n_Y + n_X^2 n_Y \log(f_{max}))$.

The complete factorization of $(f \bmod (p, S_0))$ can be determined by means of Berlekamp's algorithm [6, Section 4.6.2]; notice that (3.4) holds for every factor $(h \bmod (p, S_0))$ of $(f \bmod (p, S_0))$, because of the choice of p, and that this factorization can be found in polynomial-time, because of the bound on p. The algorithm to factorize f completely now follows by repeated application of algorithm (4.1). The above bounds on $\log(|s|)$ and p, combined with Proposition 4.4 and the fact that a factor g of f satisfies $\log(g_{max}) = O(n_X + n_Y + \log(f_{max}))$ [3], yields the following theorem.

4.5. Theorem. *The number of arithmetic operations needed to factorize f completely is $O(n_X^7 n_Y^6 + n_X^6 n_Y^6 \log(f_{max}))$, and the integers on which these operations have to be performed each have binary length $O(n_X^4 n_Y^3 + n_X^3 n_Y^3 \log(f_{max}))$.*

5. Conclusion

We have shown that basically the same ideas that were used for the polynomial-time algorithm for factoring in $\mathbb{Z}[X]$, lead to a polynomial-time factoring algorithm in $\mathbb{Z}[X, Y]$ (Theorem 4.5). Our method can be generalized to polynomials in $\mathbb{Z}[X_1, X_2, \ldots, X_t]$ for any fixed $t > 2$. The evaluation $Y = s$ is then replaced by $(X_2 = s_2, X_3 = s_3, \ldots, X_t = s_t)$, where the $s_i \in \mathbb{Z}$ have to satisfy conditions similar to (4.2). It will not be surprising that in this case the estimates become rather complicated [8].

References

[1] D.G. Cantor, Irreducible polynomials with integral coefficients have succinct certificates, *J. Algorithms* **2** (1981) 385–392.

[2] M.R. Garey and D.S. Johnson, *Computers and Intractability* (Freeman, San Francisco 1979).
[3] A.O. Gel'fond, *Transcendental and Algebraic Numbers* (Dover Publ., New York, 1960).
[4] A.J. Goldstein and R.L. Graham, A Hadamard-type bound on the coefficients of a determinant of polynomials, *SIAM Rev.* **16** (1974) 394–395.
[5] E. Kaltofen, On the complexity of factoring polynomials with integer coefficients, Ph.D. Thesis, Rensselaer Polytechnic Institute, 1982.
[6] D.E. Knuth, *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms* (Addison-Wesley, Reading, MA, 2nd ed., 1981).
[7] A.K. Lenstra, H.W. Lenstra, Jr. and L. Lovász, Factoring polynomials with rational coefficients, *Math. Ann.* **261** (1982) 515–534.
[8] A.K. Lenstra, Polynomial-time algorithms for the factorization of polynomials, Ph.D. Thesis, Univ. of Amsterdam, 1984.
[9] M. Mignotte, An inequality about factors of polynomials. *Math. Comput.* **28** (1974) 1153–1157.
[10] P.S. Wang, An improved multivariate polynomial factoring algorithm, *Math. Comput.* **32** (1978) 1215–1231.
[11] D.Y.Y. Yun, *The Hensel Lemma in Algebraic Manipulation* (MIT, Cambridge, MA, 1974) (reprint: Garland, New York, 1980).