MULTIVARIATE POLYNOMIAL FACTORIZATION*

By

David R. Musser

January 1973 TR-11

*Work partially supported by National Science Foundation grants GJ-239, GJ-30125X, and GJ-1069, the Mathematics Research Center, and the Wisconsin Alumni Research Foundation. This paper is a revision and extension of part of the author's Ph.D. thesis at the University of Wisconsin (1971).

Technical Report No. 11

Department of Computer Sciences

The University of Texas
Austin, Texas 78712

CONTENTS

Page

1. Introduction 1
2. Basic concepts 2
   a. Unique factorization domains 2
   b. Homomorphic mappings 2
   c. Polynomial notation 3
3. The univariate algorithm 4
   a. Reduction to a primitive polynomial 4
   b. Reduction to squarefree polynomials 4
   c. Choice of a prime 5
   d. Complete factorization modulo p 6
   e. Computation of factor bounds 7
   f. Construction of a corresponding factorization modulo p^j 7
   g. Quadratic Hensel construction 8
   h. Solution of a polynomial equation 9
   i. Construction of the complete factorization 10
4. The multivariate algorithm 12
   a. Reduction to a primitive polynomial 12
   b. Reduction to squarefree polynomials 12
   c. Choice of evaluation points 12
   d. Univariate factorization 13
   e. Choice of a prime 13
   f. Computation of factor bounds 14
   g. Construction of a corresponding factorization modulo $p^j,(v_1-a_1)^{j1},\dots,(v_n-a_n)^{jn}$ 14
   h. Generalized Hensel construction (mod p case) 15
   i. Generalized Hensel construction (evaluation case) 16
   j. Construction of the complete factorization 18
5. Alternative algorithms 19
6. Abstract algorithms 21
7. Squarefree factorization 24
8. Hensel constructions 27
   a. Solution of a polynomial equation 27
   b. Linear Hensel construction 28
   c. Quadratic Hensel construction 30
   d. Generalized Hensel construction 31
9. Corresponding factorizations and refinements of factorizations 37
   a. Construction of a corrseponding factorization 37
   b. Multisets 38
   c. Refinements of factorizations 39
10. Construction of a complete factorization 42
   a. Homomorphisms and sets of representives 42
   b. R-factorability 45
   c. A general factorization algorithm 45
   d. Another application 48

Acknowledgment

References

1. Introduction

This paper presents algorithms for factoring a given polynomial with integer coefficients into factors which are irreducible over the integers. These algorithms are based on the use of homomorphic mappings and "Hensel's Lemma constructions" as suggested by Zassenhaus [ZAS69]. Previous discussions of such algorithms [KNU69, pp. 390-393], [BER71], have considered only the univariate case, but the algorithms described herein (some of which are from the author's Ph.D. thesis [MUS71]) apply to the multivariate case and also to other problems such as factorization of multivariate polynomials with coefficients from a finite field.

The algorithm for the univariate case has been implemented in the SAC-1 system for algebraic calculation [COL71] and tested thoroughly. A detailed description of this implementation is given in [COL72]. Implementation of the multivariate algorithm is in progress.

Following a brief review of basic concepts in §2, we shall outline the univariate algorithm in §3 and the multivariate algorithm in §4. In order to present concisely the common theory behind these algorithms and algorithms for other coefficient domains, we shall define the concept of an abstract algorithm in §6. Sections 7 through 10 then discuss abstract algorithms for the main phases of the factorization process.

2. Basic concepts

a. Unique factorization domains

In a commutative ring with identity, zero-divisors are elements y and z such that $y \cdot z = 0$. A unit is a divisor of unity, and a prime is a nonunit element which cannot be expressed as a product of nonunit elements. An integral domain is a commutative ring with identity which contains no zero-divisors. A unique factorization domain (UFD) is an integral domain in which every nonzero element is a unit, or is prime, or has a unique factorization into primes (an expression as a product of a finite number of primes which is unique except for unit factors and the order of factors).

Primes are also called irreducible elements, and a unique factorization into primes is often called a complete factorization.

The integral domain Z of integers is a UFD (Fundamental Theorem of Arithmetic), in which the only units are 1 and -1. Any field F is a UFD in which every nonzero element is a unit and there are no irreducible elements. According to a theorem of Gauss [VDW49,§23], the polynomial domain $D[x_1,\ldots,x_n]$ is a UFD whenever D is. Thus, for example, $Z[x_1,\ldots,x_n]$ and $F[x_1,\ldots,x_n]$ are UFDs.

b. Homomorphic mappings

A mapping h from a ring R into a ring $\bar{R}$ is called a homomorphism if for all $a,b \in R$,

$$(1) \quad h(a+b) = h(a) + h(b),$$

$$(2) \quad h(ab) = h(a)h(b).$$

The kernel of h, written $\mathrm{Ker}(h)$, is the set of all elements $a \in R$ such that $h(a)=0$. $\mathrm{Ker}(h)$ is an ideal of R, a subring I such that $ir, ri \in I$ whenever $i \in I$ and $r \in R$. If R and $\bar{R}$ are commutative with identities, h induces a homomorphism of $R[x]$ into $\bar{R}[x]$, which will also be denoted by h, defined by $h(a_0 + a_1x + \ldots + a_nx^n) = h(a_o) + h(a_1)x + \ldots + h(a_n)x^n$.

The application of homomorphic mappings to factorization is based on the factor preserving property (2). The classical algorithm for factoring polynomials, Kronecker's algorithm [VDW49, §25], is based on the use of evaluation homomorphisms. For any fixed $a \in R$, the mapping e_a of $R[x]$ onto R, defined by $e_a(P) = P(a)$ for all $P(x) \in R[x]$, is homomorphic and is called an evaluation homomorphism. To factor $P(x) \in Z[x]$, for example, Kronecker's algorithm evaluates $P(x)$ at several integers, factors the resulting values in Z, and constructs the factors of $P(x)$ using interpolation.

Another well-known application of homomorphic mappings to polynomial factorization is the use of mod p factorizations, where p is a prime integer. Let $P(x) \in Z[x]$ and p be a prime which does not divide the leading coefficient of P. Let h_p denote the homomorphism of Z onto Z_p, the ring of integers modulo p. Z_p is actually a field, so $Z_p[x]$ is a UFD. If $h_p(P)$ turns out to be irreducible over Z_p, then P is irreducible over Z (except possibly for integer factors). If $h_p(P)$ does factor, then its factorization gives an idea what degrees the factors of P might have, and what residue classes the coefficients modulo p might belong to. These facts have long been used in the limited number of cases in which $h_p(P)$ is easy to factor, e.g. [VDW49, § 25]. More general applications of mod p homomorphisms have become possible since the invention by Berlekamp of efficient algorithms for factorization in $Z_p[x]$ ([BER68, Ch. 6], [KNU69, §4.6.2]). A second breakthrough was Zassenhaus' suggestion that a construction based on Hensel's Lemma, from the theory of p-adic fields, could be used to progress from a mod p factorization to a corresponding factorization modulo any power of p [ZAS69]. Taking p^j sufficiently large, we can determine from consideration of all mod p^j factorizations all factorizations over the integers. The resulting algorithm for factoring in Z[x] is much more efficient than Kronecker's algorithm. This algorithm, as developed in [MUS71], is outlined in the next section. Section 4 then presents a new algorithm for the multivariate case based on the use of evaluation and mod p homomorphisms and a generalized Hensel construction, which lifts a factorization in $Z[v_1,\ldots,v_n,x]$ modulo p, $v_1-a_1,\ldots,v_n-a_n$ to a corresponding factorization modulo p^j, $(v_1-a_1)^{j_1},\ldots,(v_n-a_n)^{j_n}$.

c. Polynomial notation

A polynomial $A(x) = a_n x^n + \ldots + a_1 x + a_0$ with coefficients $a_n,\ldots,a_1,a_0$ from a ring R, $a_n \neq 0$, is said to have degree n, leading coefficient a_n, and trailing coefficient (or constant term) a_0; we write

$$\deg(A) = n, \quad \mathrm{lc}(A) = a_n, \quad \mathrm{tc}(A) = a_0.$$

By convention, we define

$$\deg(0) = -\infty, \quad \mathrm{lc}(0) = 0, \quad \mathrm{tc}(0) = 0.$$

If R has an identity 1, we say A(x) is monic if $\mathrm{lc}(A) = 1$.

3. The univariate algorithm

The algorithm for the univariate case consists of several major sub-algorithms. In this section we shall only outline these algorithms, deferring details and proofs to later sections or references to the literature.

a. Reduction to a primitive polynomial. We are given a polynomial $C(x) \in Z[x]$ to be factored, i.e. we are given its coefficients $c_m, c_{m-1}, \ldots, c_0$ and we must determine the coefficients in Z of its irreducible factors. If $C(x) = c_0 \in Z$ then we merely have to factor c_0 in Z. Otherwise, we compute the greatest common divisor d of $c_m, \ldots c_0$ (called the content of C) and divide C(x) by d, thereby obtaining a primitive polynomial $C^*(x)$, i.e. one whose coefficients are relatively prime. Thus $C^*(x)$, called the primitive part of C(x) (denoted pp(C)), has no proper factors of degree zero, and this property simplifies the task of factoring $C^*(x)$. We proceed to factor d and $C^*(x)$ and combine the two lists of factors to produce the list of factors of C(x).

It should be noted that the algorithm requires factorization of only one integer, d, as opposed to the many integer factorizations which are required by Kronecker's algorithm. (In some applications it will not even be necessary to factor d, as only the factorization of primitive polynomials will be of interest.)

b. Reduction to squarefree polynomials. Given a primitive polynomial C(x) we proceed to factor it into "squarefree" polynomials. A polynomial is squarefree if it is the product of distinct irreducible factors. The method of factorization is based on the observation that if $C = P_1^{e1} \ldots P_k^{ek}$, P_i distinct and irreducible, then $B = \gcd(C,C') = P_1^{e1-1} \ldots P_k^{ek-1}$ (where C' is the derivative of C); hence $A = C/B = P_1 \ldots P_k$, the greatest squarefree factor of C. B can be factored into its squarefree factors with further gcd computations. Details are given in §7. We thus obtain factors $Q_1, \ldots Q_t$ such that $C=Q_1 Q_2^2 \ldots Q_t^t$, each Q_i

is squarefree, $\deg(Q_t)>0$, and the Q_i are pairwise relatively prime. We then factor each Q_i, putting i copies of each factor on the list of factors of C.

The reduction to squarefree factors is necessary to the application of the Hensel construction (§3f). Even if it were not, the ease with which gcd calculations can be carried out with modern modular gcd algorithms [BRO71] suggests that this phase of the factorization should still be carried out.

c. Choice of a prime. Given a primitive, squarefree polynomial C(x) to be factored, we first perform factorizations modulo several primes, in order to either prove irreducibility of C(x) or, failing that, to search for a prime p which yields relatively few irreducible factors modulo p, for use in the Hensel construction.

We choose only primes p such that $\bar{C} = h_p(C)$ has the same degree as C and is squarefree over Z_p. (A test of whether $\gcd(\bar{C},\bar{C}') = 1$ determines whether $\bar{C}$ is squarefree; $h_p(C)$ can fail to be squarefree for only finitely many primes p). The smallest ν such primes $p_1,\ldots p_\nu$ are chosen. Each polynomial $h_{p_i}(C)$ is partially factored over Z_{p_i} using a "distinct degree factorization" algorithm ([KNU69, p.389], [COL69]). Given a monic squarefree polynomial A over Z_p, this algorithm produces a list $((d_1,A_1),\ldots,(d_s,A_s))$ where the d_i are positive integers, $d_1<d_2<\ldots<d_s$, and A_i is the product of all monic irreducible factors of A which are of degree d_i. Thus $A = A_1\ldots A_s$ and this is a complete factorization just in case no two irreducible factors of A have the same degree.

From the list $((d_1,A_1),\ldots,(d_s,A_s))$ it is easy to construct a list of the degrees of the irreducible factors of A. In particular, if the list turns out to be $((\deg(A),A))$ then A is irreducible, and the input polynomial C must be irreducible, so the algorithm terminates. More generally, comparison of the lists of degrees of irreducible factors for the several primes yields important information about the possible degrees of factors of C. The set of degrees of factors of C must be contained in the set D_p of mod p factors for any prime p, and therefore must be contained in $D_{p_1} \cap D_{p_2} \cap \ldots \cap D_{p_\nu}$. D_p is just the set of all sums of combinations of the degrees of irreducible factors mod p, and is easily computed by an algorithm given in [MUS71, §1.6, §3.3]. If C is irreducible then we will often find

$$D_{p_1} \cap D_{p_2} \cap \ldots = \{0, \deg(C)\}$$

after a few primes have been tried, thus proving irreducibility of C.

d. <u>Complete factorization modulo p</u>. Having chosen among $p_1,\ldots,p_\nu$ the prime p for which $h_p(C)$ has the minimum number of irreducible factors, we now proceed to obtain the complete factorization of $h_p(C)$ over Z_p. We have at hand the partial factorization $((d_1,A_1),\ldots,(d_t,A_t))$ such that $h_p(C) = A_1 \ldots A_t$ and A_i is the product of irreducible factors of degree d_i. Those A_i for which $\deg(A_i)=d_i$ are irreducible, so we have only to factor the remaining A_i and combine the resulting lists of factors.

There are several possible algorithms for complete factorization over Z_p to choose from. Berlekamp's most recent algorithm [BER71] appears to be reasonably efficient even for very large primes, and its use might permit the choice of a prime large enough that the Hensel construction (§3f) would be unnecessary. This algorithm is quite complex, however, and it is probably simpler and more efficient to use Berlekamp's original algorithm with a small prime, followed by the Hensel construction. (The original algorithm has a computing time proportional to p, while the average time for the more recent algorithm is evidently proportional to a power of log p.)

A good discussion of Berlekamp's original algorithm is contained in [KNU69, §4.6.2]. SAC-1 implementations of this algorithm and the "distinct degree factorization algorithm" are described in [COL69,§3.8].

e. Computation of factor bounds. In order to determine how large the modulus, p^j, must be, it is necessary to have a bound on the coefficients of factors of $C(x)$. Let us define, for any polynomial $A(x) = a_n x^n + \ldots + a_1 x + a_0 \in Z[x]$,

$$|A|_\infty = \max_{0 \le i \le n} |a_i| ,$$

$$|A|_1 = \sum_{i=0}^{n} |a_i| .$$

One might expect, for any factor A of C, that $|A|_\infty \le |C|_\infty$ and $|A|_1 \le |C|_1$, but a simple counterexample to both inequalities is given by

$$A(x) = x^4 + 2x^3 + 3x^2 + 2x + 1,$$

$$C(x) = (x-1)A(x) = x^5 + x^4 + x^3 - x^2 - x - 1.$$

It is not difficult to show, however, that

$$|A|_1 \le (m+1)^{2m} |C|_1, \quad m = \deg(c). \qquad (1)$$

The proof, due to Collins [COL72a], is based on interpolation theory. Somewhat better bounds are obtainable with a little more computation with the coefficients of $C(x)$, as is discussed in [MUS71, §3.4].

Once a bound b on the coefficients of factors has been computed, it is necessary to compute the least integer j such that $p^j/2 > |lc(C)| \cdot b$. This choice of modulus, p^j, will ensure that any factorization of $C(x)$ will be determinable from the corresponding factorization modulo p^j, in the algorithm to be described in §3i.

f. Construction of a corresponding factorization modulo p^j. The inputs to this algorithm are a primitive, squarefree polynomial $C(x)$; a prime p such that $\deg(h_p(C)) = \deg(C)$ and $h_p(C)$ is squarefree over Z_p; a positive integer j; and distinct monic factors $G_1, \ldots, G_r$ ($r \ge 2$) of $h_p(C)$, such that

$$h_p(C) = lc(h_p(C)) G_1 \ldots G_r . \qquad (1)$$

The goal is to find a corresponding factorization of C modulo p^j, i.e.

$$
\left.\begin{array}{l}
F_1,\ldots,F_r \in Z[x] \text{ such that} \\
C \equiv lc(C)F_1\cdots F_r \pmod{p^j} \\
h_p(F_i)=G_i \\
deg(F_i)=deg(G_i) \\
F_i \text{ is monic}
\end{array}\right\} \quad i=1,\ldots,r. \qquad (2)
$$

If p^j is sufficiently large and the factorization (1) is complete, the corresponding factorization (2) can be used to determine the complete factorization of C over Z (§3i).

The algorithm initially sets $\bar{C} \leftarrow h_p(C)$, then, for i=1,...,t, repeats the following steps:

1. Set $\bar{A} \leftarrow G_i$, $\bar{B} \leftarrow \bar{C}/\bar{A}$.
2. Since $\bar{C}$ is squarefree, $\bar{A}$ and $\bar{B}$ are relatively prime; hence there exist $\bar{S}$, $\bar{T} \in Z_p[x]$ such thet $\bar{A}\bar{S} + \bar{B}\bar{T} = 1$. $\bar{S}$ and $\bar{T}$ can be computed via the Extended Euclidean Algorithm [KNU69,p.377,537].
3. Using the Hensel construction described in §3g, applied to p, j, C, $\bar{A}$, $\bar{B}$, $\bar{S}$, $\bar{T}$, obtain A, $B \in Z[x]$ such that $C \equiv AB \pmod{p^j}$, $h_p(A)=\bar{A}$, $h_p(B)=\bar{B}$, $deg(A)=deg(\bar{A})$ and A is monic.
4. Set $F_i \leftarrow A$, $C \leftarrow B$, $\bar{C} \leftarrow \bar{B}$.

g. Quadratic Hensel construction. Given a nonzero polynomial $C \in Z[x]$; a nonzero integer p; a positive integer j; $\bar{A}$, $\bar{B}$, $\bar{S}$, $\bar{T} \in Z_p[x]$ such that $h_p(C)=\bar{A}\bar{B}$ and $\bar{A}\bar{S}+\bar{B}\bar{T}=1$; this algorithm constructs A, B, S, $T \in Z[x]$ such that $C \equiv AB$ and $AS+BT \equiv 1 \pmod{p^j}$ $h_p(A)=\bar{A}$, $h_p(B)=\bar{B}$, $deg(A)=deg(\bar{A})$, and lc(A) is a unit modulo p^j. This "quadratic" construction, so-called because it progresses through factorizations modulo p, p^2, p^4, p^8,... in successive iterations, is based on a construction discussed by Knuth [KNU69, pp.398, 546]. This version differs somewhat from the construction originally proposed by Zassenhaus [ZAS69], although the latter is also quadratic in nature. (Hensel's original construction [VDW49, pp. 248-250] was only linear.)

1. Set $i \leftarrow 1$, $q \leftarrow p$ and choose A, B, S, T $\in$ Z[x] such that $h_p(A)=\bar{A}$, $h_p(B)=\bar{B}$, $h_p(S)=\bar{S}$, $h_p(T)=\bar{T}$, and $\deg(A)=\deg(\bar{A})$.
2. If $i \geq j$, the algorithm terminates.
3. Set $U \leftarrow (C-AB)/q$. Using the algorithm in § 3h, solve the congruence $AY+BZ \equiv U \pmod q$ for Y, Z $\in$ Z[x] such that $\deg(Z) < \deg(A)$.
4. Set $A^* \leftarrow A+qZ$, $B^* \leftarrow B+qY$.
5. Set $U_1 \leftarrow (A^*S+B^*T-1)/q$. Using the algorithm in §3h, solve the congruence $AY_1 + BZ_1 \equiv U_1 \pmod q$ for Y_1, $Z_1 \in$ Z[x] such that $\deg(Z_1) < \deg(A)$.
6. Set $S^* \leftarrow S-qY_1$, $T^* \leftarrow T-qZ_1$.
7. Replace i, q, A, B, S, T by 2i, q^2, A*, B*, S*, T* and go to 2.

A proof of the validity of this algorithm will be given in § 8.

Although we have not indicated it in the above description, it is easy to construct A so that $|A|_\infty < p^j/2$, and similarly for B, S and T. (The test for termination must be changed and the modulus used in the last iteration must be chosen appropriately. Details are contained in [COL72, §5.2]). If $\bar{A}$ is monic then A can be constructed to be monic also.

h. Solution of a polynomial equation. Let R be the ring of integers modulo q. Given A, B, S, T, U $\in$ R[x] such that lc(A) is a unit of R and AS+BT=1, we may determine Y, Z $\in$ R[x] such that AY+BZ=U and $\deg(Z) < \deg(A)$ as follows:

1. Set $V \leftarrow TU$.
2. Divide V by A, obtaining Q, Z $\in$ R[x] such that

   $$V = AQ + Z,\ \deg(Z) < \deg(A).$$

3. Set $Y \leftarrow SU+BQ$. (Thus $AY+BZ=A(SU+BQ) + B(TU-AQ) = (AS+BT)U = U$.)

The division in step 2 is possible because lc(A) is a unit of R. A division algorithm will be discussed in §6.

i. <u>Construction of the complete factorization</u>.

The inputs to this algorithm are a primitive polynomial $C(x) \in Z[x]$; an integer m such that $m/2 > |lc(C)|\, b$, where b bounds the coefficients of any factor of C of degree $\leq d^* = \lfloor \deg(C)/2 \rfloor$; monic polynomials $G_1, \ldots, G_r \in Z[x]$ which comprise a "modulo m refinement" of the complete factorization of C (see below); and a set D which contains the set $\{d: d=\deg(A),\ A \mid C,\ 0 < d \leq d^*\}$. The outputs are irreducible polynomials $F_1, \ldots, F_r \in Z[x]$ such that $C = F_1 \ldots F_r$, the complete factorization of C over Z.

For polynomials $A_1, A_2, \ldots, A_r, B_1, B_2, \ldots, B_s$ over a ring R such that

$$A_1 A_2 \ldots A_r = e B_1 B_2 \ldots B_s \text{ for some } e \in R,$$

we say that $B_1, B_2, \ldots, B_s$ are a <u>refinement</u> of $A_1, A_2, \ldots, A_r$ if there exists a partition of $\{1, \ldots, s\}$ into disjoint subsets $I_1, I_2, \ldots, I_r$ for which

$$A_j = e_j \prod_{k \in I_j} B_k \quad \text{for some } e_j \in R,\ 1 \leq j \leq r.$$

We call $G_1, \ldots, G_s \in Z[x]$ a modulo m refinement of $F_1, \ldots, F_r$ if $h_m(G_1), \ldots, h_m(G_s)$ are a refinement of $h_m(F_1), \ldots, h_m(F_s)$.

The set $D = D_{p_1} \cap D_{p_2} \cap \ldots \cap D_{p_\nu}$, constructed from the modulo p_i factorizations as described in §3c, can be used as the last input parameter to this algorithm.

The algorithm, which will be described in detail in §10, considers all products

$$A^* \equiv lc(C)\, G_{i_1} \ldots G_{i_s} \pmod{m} \tag{2}$$

$$|A^*|_\infty < m/2,$$

testing whether A* divides C*= lc(C)C. If so, A=pp(A*) is a factor of C, and the algorithm continues with B=C/A in place of C, the remaining G_i, $i \neq i_1, \ldots, i_s$, being a modulo m refinement of the complete factorization of B. The products (2) are considered in order of increasing degree of A*, so that when a factor A=pp(A*) is found, it is known to be irreducible. Only products with degree $d \leq d^*$, $d \in D$ are considered.

A "trailing coefficient test" is applied to eliminate computation of some or all of the A*. Letting tc(A) denote the trailing coefficient (constant term) of A, we compute

$$t \equiv lc(C)tc(G_{i_1}) \ldots tc(G_{i_s}) \pmod m$$
$$|t|_\infty < m/2.$$

Then t = tc(A*), and if t fails to divide tc(C*) then A* cannot divide C*, so the computation of A* can be skipped.

4. The multivariate algorithm

The multivariate algorithm uses evaluation and mod p homomorphisms to reduce the problem of factoring a given polynomial in $Z[v_1,\ldots,v_n,x]$ to factorization of a related polynomial in $Z[x]$; construction of corresponding factorizations in $Z[v_1,\ldots,v_n,x]$ modulo p^j, $(v_1-a_1)^{j_1},\ldots,(v_n-a_n)^{j_n}$, where $a_1,\ldots,a_n,j,j_1,\ldots,j_n$ are selected integers; and determination, from these factorizations, of all irreducible factors in $Z[v_1,\ldots,v_n,x]$.

a. Reduction to a primitive polynomial. Given $C(v_1,\ldots,v_n,x) \in Z[v_1,\ldots,v_n,x]$, $n \geq 0$, to be factored, we can proceed initially as in the univariate case by regarding C as a polynomial in x with coefficients $c_m, c_{m-1},\ldots,c_0$ in $Z[v_1,\ldots,v_n]$. We first compute the gcd d of $c_m,\ldots,c_0$ and divide C by d, obtaining $C^*(x)$ which is primitive over $Z[v_1,\ldots,v_n]$. We recursively factor $d(v_1,\ldots,v_n)$ and factor $C^*(v_1,\ldots,v_n,x)$, then combine the resulting lists of factors.

Except for $d(v_1,\ldots,v_n)$, we are not required to factor any polynomials in $Z[v_1,\ldots,v_n]$; unlike the multivariate version of Kronecker's algorithm, this algorithm is not directly recursive.

b. Reduction to squarefree polynomials. As in a, we can apply the same squarefree factorization algorithm as in the univariate case.

c. Choice of evaluation points. Given a squarefree polynomial $C \in Z[v_1,\ldots,v_n,x]$, $n \geq 1$ (if n = 0, this step is omitted), we choose integers $a_1,\ldots,a_n$ such that the univariate polynomial $\bar{C}(x) = C(a_1,\ldots,a_n,x)$ satisfies:

(1) $\deg(\bar{C}) = \deg_x(C)$ (degree of C in x)

(2) $\bar{C}$ is squarefree.

The following recursive algorithm can be used.

1. Set $a_n \leftarrow 0$, $c \leftarrow lc(C)$.
2. Set $A \leftarrow c(v_1,\ldots,v_{n-1},a_n)$.
3. If A = 0, set $a_n \leftarrow a_n + 1$ and go to 2.

4. Set $\hat{C}(v_1,\ldots,v_{n-1},x) \leftarrow C(v_1,\ldots,v_{n-1},a_n,x)$, $B \leftarrow \gcd(\hat{C},\hat{C}')$ where the prime denotes differentiation with respect to x. If $\deg(B) > 0$ (in which case $\hat{C}$ is not squarefree), set $a_n \leftarrow a_n + 1$ and go to 2. (If $\deg(B) = 0$, then $\hat{C}$ is squarefree; $\hat{C}$ can fail to be squarefree for only a finite number of integers a_n.)

5. If n = 1, the algorithm terminates. ($\bar{C}(x) = C(a_1,x) = \hat{C}(x)$, hence $\deg(\bar{C}) = \deg(\hat{C}) = \deg_x(C)$ and $\bar{C}$ is squarefree.)

6. Apply this algorithm recursively to $\hat{C}(v_1,\ldots,v_{n-1},x)$ to obtain $a_1,\ldots,a_{n-1} \in Z$ such that $\deg(\hat{C}(a_1,\ldots,a_{n-1},x)) = \deg_x(\hat{C})$ and $\hat{C}(a_1,\ldots,a_n,x)$ is squarefree. (Then, since $\bar{C}(x) = C(a_1,\ldots,a_n,x) = \hat{C}(a_1,\ldots,a_{n-1},x)$, we have $\deg(\bar{C}) = \deg_x(\hat{C}) = \deg_x(C)$ and $\bar{C}$ is squarefree.)

d. Univariate factorization. Having obtained $\bar{C}(x)$ from $C(v_1,\ldots,v_n,x)$ as described in c, we now factor $\tilde{C}=pp(\bar{C})$ using the algorithm described in §3, obtaining irreducible $\tilde{F}_1,\ldots,\tilde{F}_t \in Z[x]$ such that $\tilde{C}=\tilde{F}_1\ldots\tilde{F}_t$. If n=0 or t=1, we are done; otherwise, we attempt to extend this factorization to a factorization of C in the following steps.

e. Choice of a prime. We now choose a prime integer p such that $h_p(\tilde{C})$ is squarefree and has the same degree as $\tilde{C}$. It is not necessary to use the same prime as was chosen in the univariate algorithm; it is better now to choose p as large as possible while less than the bound on single precision numbers for the machine on which the algorithm is implemented. Now let G_i be the monic associate of $h_p(\tilde{F}_i)$ for i=1,...,t. Thus $G_i \in Z_p[x]$ and $h_p(\tilde{C}) = lc(h_p(\tilde{C}))G_1\ldots G_t$. Since $h_p(\tilde{C}) = h(C)$, where h is the homomorphism from $Z[v_1,\ldots,v_n]$ onto Z_p which is the composite of h_p and the evaluation homomorphism $A(v_1,\ldots,v_n) \mapsto A(a_1,\ldots,a_n)$, we have

$$h(C) = lc(h(C))G_1\ldots G_t.$$

It is easy to show (§9) that this is a refinement of $h(F_1),\ldots,h(F_r)$, where $C = F_1\ldots F_r$ is a complete factorization of C.

f. Computation of factor bounds. The bound 3e-(1) generalizes to multivariate polynomials. Define $|A|_\infty$ and $|A|_1$ for $A \in Z[x]$ as in §3e and for $A \in Z[v_1,\ldots,v_n,x]$ inductively by:

$$A(v_1,\ldots,v_n,x) = \sum_{i=0}^{m} A_i(v_1,\ldots,v_n)x^i,$$

$$|A|_\infty = \max_{0 \le i \le m} |A_i|_\infty,$$

$$|A|_1 = \sum_{i=0}^{m} |A_i|_1.$$

If A, C ε $Z[v_1,\ldots,v_n,x]$ and A divides C, then

$$|A|_1 \le \prod_{i=0}^{n} (m_i+1)^{2m_i} |C|_1$$

where $m_0 = \deg_x(C)$, $m_i = \deg_{v_i}(C)$. This theorem (in slightly weaker form) is due to Collins [COL72a].

If b is a factor bound for $C \varepsilon Z[v_1,\ldots,v_n,x]$ and b' is a factor bound for $lc(c) \varepsilon Z[v_1,\ldots,v_n]$, then we must choose the least integer j such that $p^j/2 > bb'$. We also choose $j_i = \deg_{v_i}(C) + \deg_{v_i}(lc(C))$ $1 \le i \le n$, as a bound on the degree in v_i of any factor of the polynomial $lc(C) \cdot C$, as will be required in §4j.

g. Construction of a corresponding factorization modulo $p^j, (v_1-a_1)^{j_1}, \ldots, (v_n-a_n)^{j_n}$. This construction is a generalization of the one described for the univariate case in §3f. The inputs are a primitive, squarefree polynomial $C(v_1,\ldots,v_n,x)$; a prime p and integers $a_1,\ldots,a_n$ determining a homomorphism h with kernel $(p, v_1-a_1, \ldots, v_n-a_n)$ such that $\deg(h(C)) = \deg_x(C)$ and h(C) is squarefree over Z_p; a list $G = (G_1,\ldots,G_r)$ $(r \ge 2)$ of monic factors of $h_p(C)$; and positive integers j, $j_1,\ldots,j_n$. The output is a corresponding factorization of C modulo $\mathfrak{m} = (p^j,$

$(v_1-a_1)^{j_i},\ldots,(v_n-a_n)^{j_n})$: a list $F = (F_1,\ldots,F_r)$ of $F_i \in Z[v_1,\ldots,v_n,x]$ such that

$$\left.\begin{array}{l} C \equiv lc(C)F_1\cdots F_r \pmod{\mathfrak{m}} \\ h(F_i) = G_i \\ \deg_x(F_i) = \deg(G_i) \\ F_i \text{ is monic} \\ |F_i|_\infty < p^j/2 \\ \deg_{v_k}(F_i) < j_k,\ k=1,\ldots,n \end{array}\right\} \quad i=1,\ldots,r \qquad (1)$$

When applied with $G_1,\ldots,G_r$ which are a refinement of $h(F_1),\ldots,h(F_r)$, the corresponding factorization (1) is easily shown to be a modulo $\mathfrak{m}$ refinement of $F_1,\ldots,F_r$.

The algorithm is the same as in §3f, except that in step 3 the generalized Hensel construction to be described in §4h is applied to p, $v_1-a_1,\ldots,v_n-a_n,j,j_1,\ldots,j_n,\bar{C},\bar{A},\bar{B},\bar{S},\bar{T}$ to obtain $A,B \in Z[v_1,\ldots,v_n,x]$ such that $C\equiv AB \pmod{m}$, $h(A)=\bar{A}$, $h(B)=\bar{B}$, $\deg_x(A)=\deg(\bar{A})$, A is monic, $|A|_\infty$, $|B|_\infty < p^j/2$ and $\deg_{v_k}(B) < j_k$, $k=1,\ldots,n$.

h. <u>Generalized Hensel construction (mod p case)</u>. This algorithm is a generalization of that in §3g in which the ring of coefficients Z is replaced by $Z[v_1,\ldots,v_n]$, and the kernel (p) of the homomorphism becomes $(p,v_1-a_1,\ldots,v_n-a_n)$. Instead of (p^j) we have $(p^j,(v_1-a_1)^{j_1},\ldots,(v_n-a_n)^{j_n})$. In §8, we shall give a still more general version and a proof of its validity. In that version the prime p is treated equally with $v_1-a_1,\ldots,v_n-a_n$, but for ease and efficiency of implementation it seems best to handle the extension from a mod p to a mod p^j factorization as a separate case from the extensions of $\text{mod}(v_i-a_i)$ to $\text{mod}(v_i-a_i)^{j_i}$ factorizations. The algorithm for the mod p case is as follows:

1. Let h_1 be the homomorphism from $Z[v_1,\ldots,v_n]$ onto $Z_p[v_1,\ldots,v_n]$, set $C^+ \leftarrow h_1(C) \in Z_p[v_1,\ldots,v_n,x]$, $a_1^+ \leftarrow h_1(a_1),\ldots,a_n^+ \leftarrow h_1(a_n)$, and apply the

algorithm in §4i to $a_1^+,\ldots,a_n^+$, $j_1,\ldots,j_n$, $C^+,\bar{A},\bar{B},\bar{S},\bar{T}$, obtaining $A^+,B^+,S^+,T^+ \in Z_p[v_1,\ldots,v_n,x]$ such that lc(A) is a unit modulo $\mathfrak{m}^+=((v_1-a_1^+)^{j_1},\ldots,(v_n-a_n^+)^{j_n})$, $C^+\equiv A^+B^+$ and $A^+S^+ + B^+T^+\equiv 1 \pmod{\mathfrak{m}^+}$, $\deg(A^+)=\deg(\bar{A})$, $h^+(A^+)=\bar{A}$ and $h^+(B^+)=\bar{B}$ where h^+ is the evaluation homomorphism from $Z[v_1,\ldots,v_n]$ onto Z_p defined by $P(v_1,\ldots,v_n)\mapsto P(a_1,\ldots,a_n)$.

2. Apply the Quadratic Hensel construction to p, j, C, A^+, B^+, S^+, T^+. The version to be used is identical to that of §3g except that, instead of performing operations in the ring Z, we perform them in $Z[v_1,\ldots,v_n]$ modulo $(v_1-a_1)^{j_1}$, $\ldots,(v_n-a_n)^{j_n}$. As outputs we obtain A, B, S, $T\in Z[v_1,\ldots,v_n,x]$ satisfying the conditions stated at the end of §4g and $AS + BT \equiv 1 \pmod{\mathfrak{m}}$.

i. <u>Generalized Hensel construction (evaluation case)</u>. The inputs are $C\in Z_p[v_1,\ldots,v_n,x]$, $a_1,\ldots,a_n \in Z_p$, $j_1,\ldots,j_n$, $\bar{A},\bar{B},\bar{S},\bar{T} \in Z_p[x]$ such that $h(C)=\bar{A}\bar{B}$ and $\bar{A}\bar{S}+\bar{B}\bar{T}=1$, where h is the evaluation homomorphism $P(v_1,\ldots,v_n)\mapsto P(a_1,\ldots,a_n)$. The outputs are $A,B,S,T \in Z_p[v_1,\ldots,v_n,x]$ such that lc(A) is a unit modulo $\mathfrak{m}=((v_1-a_1)^{j_1},\ldots,(v_n-a_n)^{j_n})$, $C\equiv AB$ and $AS+BT\equiv 1 \pmod{\mathfrak{m}}$, $h(A)=\bar{A}$, $h(B)=\bar{B}$ and $\deg(A)=\deg(\bar{A})$.

1. If n=1, apply the Quadratic Hensel construction to v_1-a_1, j_1, C, $\bar{A}$, $\bar{B}$, $\bar{S}$, $\bar{T}$. The version to be used is identical to that of §3g except that, instead of performing operations in the ring Z, we perform them in $Z_p[v_1]$. We obtain $A,B,S,T\in Z_p[v_1,x]$ satisfying the output conditions stated above, and terminate the algorithm.

2. Let h_1 be the evaluation homomorphism $P(v_1,\ldots,v_n)\mapsto P(v_1,\ldots,v_{n-1},a_n)$ of $Z_p[v_1,\ldots,v_n]$ onto $Z_p[v_1,\ldots,v_{n-1}]$ and h^+ be the evaluation homomorphism $P(v_1,\ldots,v_{n-1})\mapsto P(a_1,\ldots,a_{n-1})$. Set $C^+\leftarrow h_1(C)$, and apply this algorithm recursively to C^+, $a_1,\ldots,a_{n-1}$, $j_1,\ldots,j_{n-1}$, $\bar{A},\bar{B},\bar{S},\bar{T}$, obtaining $A^+,B^+,S^+,T^+ \in Z_p[v_1,\ldots,v_{n-1},x]$ such that $lc(A^+)$ is a unit modulo $\mathfrak{m}^+=((v_1-a_1)^{j_1},\ldots,(v_{n-1}-a_{n-1})^{j_{n-1}})$, $C^+\equiv A^+B^+$ and $A^+S^+ + B^+T^+ \equiv 1 \pmod{\mathfrak{m}^+}$, $h^+(A^+)=\bar{A}$, $h^+(B^+)=\bar{B}$ and $\deg(A)=\deg(\bar{A})$.

3. Apply the Quadratic Hensel construction to $v_n - a_n$, j_n, C, A^+, B^+, S^+, T^+. This time, instead of performing operations in the ring Z, we perform them in $Z_p[v_1, \ldots, v_n]$ modulo $(v_1-a_1)^{j_1}, \ldots, (v_{n-1}-a_{n-1})^{j_{n-1}}$. We obtain $A, B, S, T \in Z_p[v_1, \ldots, v_n, x]$ satisfying the output conditions stated above.

j. <u>Construction of the complete factorization</u>. The inputs are $C \in Z[v_1,\ldots,v_n, x]$ which is primitive over $Z[v_1,\ldots,v_n]$; an integer m such that $m/2 > bb'$, where b bounds the coefficients of any factor of C of degree $\leq d^* = \lfloor (\deg_x(C)/2 \rfloor$ and b' bounds the coefficients of any factor of lc(C); and polynomials $G_1,\ldots,G_r$ such that

$C \equiv lc(C)G_1G_2\ldots G_r \pmod{\mathfrak{m}}$ where $\mathfrak{m} = (m, (v_1-a_1)^{j_1},\ldots,(v_n-a_n)^{j_n})$ and $a_1,\ldots,$ $a_n \in Z$,

$G_i \in Z[v_1,\ldots,v_n,x]$, monic, $1 \leq i \leq r$,

$j_i = \deg_{v_i}(C) + \deg_{v_i}(lc(C))$, $1 \leq i \leq n$,

and $G_1,\ldots,G_r$ are a modulo $\mathfrak{m}$ refinement of the complete factorization of C over Z. The outputs are irreducible polynomials $F_1,\ldots,F_t \in Z[v_1,\ldots,v_n,x]$ such that $C = F_1\ldots F_t$, the complete factorization of C in $Z[v_1,\ldots,v_n,x]$.

The algorithm is much the same as in the univariate case. All products

$A^* \equiv lc(C)G_{i_1}\ldots G_{i_s} \pmod{\mathfrak{m}}$,

$|A^*|_\infty < m/2$, $\deg_{v_i}(A^*) \leq j_i$, $1 \leq i \leq n$,

are considered, but no degree tests or trailing coefficient tests are applied, since in fact, with $G_1,\ldots,G_r$ constructed as in §4d - 4g (but denoted by $F_1,\ldots,$ F_r in §4g), it is probable that each G_i corresponds to a factor of C.

5. Alternative algorithms.

The reader may be dismayed by the complexity of the multivariate algorithm and wonder whether simpler alternatives exist which are of comparable efficiency. Several somewhat simpler versions were considered by the author before the discovery of the generalized Hensel construction:

a. By regarding a polynomial C in $Z[v_1,\ldots,v_n,x]$ as a polynomial in $Q(v_1,\ldots,v_n)[x]$, where $Q(V_1,\ldots,v_n)$ denotes the field of rational functions of $v_1,\ldots,v_n$, and a factorization of $C(v_1,\ldots,v_{n-1},a_n,x) \varepsilon Q(v_1,\ldots,v_{n-1})[x]$ as a factorization of C modulo v_n-a_n, a quadratic Hensel algorithm can be used to lift this factorization to a corresponding factorization modulo $(v_n-a_n)^{j_n}$, which can then be tested for being an actual factorization. Since $Q(v_1,\ldots,v_{n-1})$ is a field, the theory presented in §8 shows that such a Hensel construction exists. Thus the original problem is reduced recursively to a problem in one fewer variable and ultimately to factorization in Q[x], which can be handled by a minor extension to the algorithm for Z[x]. The problem with this approach is that rational function computations are required, which are generally much more expensive than computations with polynomials, because of the gcd computations required to keep results in lowest terms.

b. Another approach, probably somewhat better than a, would be to map C into $\bar{C}$ in $Z_p(v_1,\ldots,v_n)[x]$ for an appropriately chosen prime p. A recursive algorithm can be used for factoring in $Z_p(v_1,\ldots,v_n)[x]$, similar to the one for $Q(v_1,\ldots,v_n)[x]$ described in a. Then a factorization of $\bar{C}$ can be lifted by means of a qudratic Hensel construction to a factorization of C modulo p^j, which can be tested for being an actual factorization of C. Again, however, the computations required in $Z_p(v_1,\ldots,v_n)$ and in $Q[v_1,\ldots,v_n]$ would be very costly.

c. Instead of working in $Q(v_1,\ldots,v_n)$ or $Z_p(v_1,\ldots,v_n)$, the computations can be restricted to the integral domain $\mathcal{J} = Z[v_1,\ldots,v_n]$ or $Z_p[v_1,\ldots,v_n]$, by using a "trial Hensel construction." This construction uses polynomials $\bar{S},\bar{T} \in \mathcal{J}[x]$ and $r \in \mathcal{J}$ for which $\bar{A}\bar{S} + \bar{B}\bar{T} = r$ in an attempt to find a factorization $C \equiv AB(\mathrm{mod}\ p^j)$, $A,B \in \mathcal{J}[x]$ corresponding to a factorization $C \equiv \bar{A}\bar{B}(\mathrm{mod}\ p)$. The construction may fail, but it is not difficult to arrange the computation so that the construction is guaranteed to succeed if $\bar{A}$ and $\bar{B}$ correspond to actual factors of C. This approach has two apparent drawbacks. The trial construction is only linear, being based on Algorithm H of §8. Secondly, the polynomials $\bar{S}$ and $\bar{T}$ must be obtained independently (by a version of the Extended Euclidean Algorithm), rather than as a byproduct of the Hensel construction, as they are in the generalized version.

In summary, these alternate approaches, while possibly less complex than the algorithm of §4, would probably be considerably less efficient.

6. Abstract Algorithms

In this paper we shall use "abstract algorithm" descriptions in order to present compactly the common theory behind factoring algorithms for both the univariate and multivariate cases and for a number of coefficient domains. An abstract algorithm is one in which the domains of the inputs and outputs are abstract sets or algebraic systems such as rings, integral domains, or fields. An example of an abstract algorithm is:

Algorithm D (Division of polynomials over a ring). Let $\mathcal{R}$ be a commutative ring with identity. Given polynomials A, B $\in \mathcal{R}[x]$ with lc(B) a unit of $\mathcal{R}$, this algorithm computes polynomials Z, R $\in \mathcal{R}[x]$ such that

$$A = BQ + R \text{ and } \deg(R) < \deg(B).$$

(1) Set $Q \leftarrow 0$ and $R \leftarrow A$.

(2) Now Q, R $\in \mathcal{R}[x]$ and $A = BQ + R$. If $\deg(R) < \deg(B)$, exit.

(3) Set $n \leftarrow \deg(R) - \deg(B)$, $T \leftarrow (\mathrm{lc}(R)/(\mathrm{lc}(B))x^n$, $Q \leftarrow Q + T$, $R \leftarrow R - TB$ (this reduces the degree of R), and go to (2).

In dealing with abstract algorithms we leave open the question of what assumptions are required about the abstract domains involved in order to prove effectiveness of the algorithm. (Such questions have been dealt with elsewhere, e.g. [RAB60].) We shall however, require that, under the assumption that each step can be effectively performed, the algorithm will terminate in a finite number of steps. A proof of termination of Algorithm D is indicated in the parenthetical assertion in step (3): by the choice of the term T of the quotient polynomial Q, both R and TB have the same leading coefficient, hence the new value of R, $R_1 = R - TB$, is of smaller degree than that of R, and thus the condition tested in step (2) must eventually be satisfied.

If we do not require effectiveness in our abstract algorithms, the reader may well ask, by what criteria do we construct them? For we could in some steps of our algorithms merely cite the existence of some quantity without any indication of a method of constructing the quantity. However, all of the algorithms to be presented have been written with the purpose of generalizing methods which are known not just to be effective in particular domains, but to be "very effective," or "efficient" methods. This is meant in the sense that each step of the abstract algorithm is of sufficient simplicity that there are known to be efficient algorithms for carrying it out in at least one particular domain. In Algorithm D, for example, each step involves only simple arithmetic op erations for which efficient algorithms are known, when R is the ring of integers, or the rational number field, or a finite field.

Besides the proof of termination, we are also interested in proving the validity of the algorithm: That when applied to inputs which satisfy the input assumptions, the algorithm produces outputs which satisfy the output assertions. The method of proof to be used is based on the method of "inductive assertions" described in [FLO67] and [KNU68, Section 1.2.1]. The basic idea of the method is to associate with some or all of the steps or substeps of the algorithm assertions about the current state of the computation, and to prove that each assertion is true each time control reaches the corresponding step, under the assumption that the previously encountered assertions are true. If this can be done in such a way that the assertions associated with the first step are the input assumptions and those associated with the terminal step(s) are the output assertions, then the algorithm is necessarily valid, by induction on the number of steps performed.

In applying the method we have usually not attempted to list all of the assertions which actually hold at each step; in general we have tried to maintain about the same degree of explicitness as is usual in a conventional proof of a theorem. In Algorithm D, we have included only two assertions, in step (2), for the purpose of proving validity (the assertion in step (3) was included for the sake of proving termination, as discussed previously). It is trivial that these assertions were true the first time step (2) is executed. Assuming them true at a given execution of step (2), they may be shown to be true at the next execution as follows: let $Q_1 = Q + T$ and $R_1 = R - TB$; since lc(B) is a unit, $T \in R[x]$, hence so are Q_1 and R_1; also $BQ_1 + R_1 = B(Q + T) + R - TB = BQ + R = A$; since Q is set to Q_1 and R to R_1 in step (3), the assertions $Q, R \in R[x]$ and $A = BQ + R$ still hold when step (2) is reached again.

The abstract algorithm concept may be easily formalized in terms of conventional set theory, and in fact such a formalization is given by Knuth in his initial formal definition of algorithms [KNU68, pp 7-8]. (Knuth goes on to modify this definition to include the property of effectiveness.) The inductive assertion method is also easily formalized in terms of Knuth's model, as shown in [MUS71].

7. Squarefree factorization

A polynomial is said to be squarefree if it has no nonconstant factor which is the square of another polynomial. If C is a polynomial over a UFD which is nonconstant, primitive and squarefree, then C has a complete factorization $C = P_1 P_2 \ldots P_n$ where the P_i are distinct prime polynomials of positive degree.

Elements x and y in a ring D are said to be associates if x = uy for some unit u of R. We write $x \sim y$ (this is an equivalence relation).

The characteristic of a ring D is the smallest positive integer n such that nx = 0 for all x in D, or zero if no such integer exists. (If D is an integral domain, the characteristic is prime if it is not zero.)

Theorem S. Let D be a UFD, C be a nonconstant, primitive polynomial over D, and B = gcd(C,C') where C' denotes the derivative of C. Let $C = P_1^{e_1} \ldots P_n^{e_n}$ be a complete factorization of C.

a. If deg(B) = 0 then C is squarefree.

b. If D has characteristic zero, then $B \sim P_1^{e_1 - 1} \ldots P_n^{e_n - 1}$.

c. If D has characteristic zero and C is squarefree, then $B \sim 1$.

d. If D has characteristic zero, then $C/B \sim P_1 \ldots P_n$, the greatest squarefree divisor of C.

Proof: a. Suppose C is not squarefree; thus $C = P^2 Q$ for some P and Q over D, deg(P) > 0. Then $C' = P^2 Q' + 2PP'Q$ is a multiple of P, hence $P|B$, hence deg(B) > 0. Thus deg(B) = 0 implies C is squarefree.

b. Since $B|C$, $B \sim P_1^{\delta_1} \ldots P_n^{\delta_n}$, where $0 \le \delta_i \le e_i$, $1 \le i \le n$. To show that $\delta_i = e_i - 1$, let $P = P_i$, $e = e_i$ and $Q = C/P^e$. Then $C = P^e Q$ and $C' = P^e Q' + eP^{e-1}P'Q$, hence $P^{e-1}|B$. Suppose $P^e|B$. Then $P^e|C'$, hence $P^e|eP^{e-1}P'Q$, and since D is an integral domain, $P|eP'Q$. But P and Q are relatively prime, so $P|eP'$. Since the characteristic of D is zero, $eP' \neq 0$, hence $\deg(eP') \geq \deg(P)$, a contradiction. Thus $P^e \nmid B$, while $P^{e-1}|B$, so $\delta_i = e - 1 = e_i - 1$.

c,d. Obvious from b.

Thus to factor C one could compute the greatest squarefree divisor $A = C/\gcd(C,C')$ and factor it to obtain the P_i, then divide C by P_i as many times as possible, to determine the e_i. However, we can do better than this if C is not already squarefree, for we will show that we can then partially factor C and determine the e_i by means of further gcd calculations.

Let $Q_i = \prod_{j \in E_i} P_j$, where $E_i = \{ j : e_j = i\}$. ($Q_i = 1$ when E_i is empty.) Then, for $t = \max\{e_1, \ldots, e_n\}$ we have

$$C = Q_1 Q_2^2 \cdots Q_t^t, \quad Q_i \text{ squarefree,}$$
$$\deg(Q_t) > 0, \ \gcd(Q_i, Q_j) \sim 1 \text{ for } i \neq j. \tag{1}$$

We call (1) a squarefree factorization of C, since each Q_i is either unity or a squarefree polynomial of positive degree. The Q_i are uniquely determined by the conditions in (1), except for unit factors.

By Theorem S, if $B = \gcd(C,C')$ and $A = C/B$ then $B \sim Q_2 Q_3^2 \cdots Q_t^{t-1}$ and $A \sim Q_1 Q_2 \cdots Q_t$. If $D = \gcd(A,B)$ then $D \sim Q_2 Q_3 \cdots Q_t$, hence $Q_1 \sim A/D$. The following algorithm shows how we can continue, computing $Q_2, \ldots, Q_t$:

Algorithm S (Squarefree factorization). Let $\mathcal{D}$ be a UFD of characteristic zero. Given a primitive polynomial C of positive degree, let $C = Q_1 Q_2^2 \cdots Q_t^t$ be a squarefree factorization of C. This algorithm computes t and $A_1 \sim Q_1, \ldots, A_t \sim Q_t$.

(1) Set $B \leftarrow \gcd(C,C')$, $A \leftarrow C/B$, $j \leftarrow 1$.

(2) (At this point $B \sim Q_{j+1} Q_{j+2}^2 \cdots Q_t^{t-j}$ and $A \sim Q_j Q_{j+1} \cdots Q_t$). If $B \sim 1$ then set $t \leftarrow j$, $A_t \leftarrow A$, and exit.

(3) Set $D \leftarrow \gcd(A,B)$, $A_j \leftarrow A/D$. (Then $D \sim Q_{j+1} Q_{j+2} \cdots Q_t$ and $A_j \sim Q_j$.)

(4) Set $B \leftarrow B/D$, $A \leftarrow D$, $j \leftarrow j+1$, and go to (2).

The reader may easily verify the inductive assertions in the algorithm.

Algorithm S is based on an algorithm presented by Horowitz in [HOR69, pp. 58-60, 69-70], which in turn was based on an algorithm due to Tobey. Horowitz' version is equivalent to Algorithm S with steps (3) and (4) replaced by:

(3') Set $E \leftarrow \gcd(B,B')$, $D \leftarrow B/E$, $A_j \leftarrow A/D$. (Then $E \sim Q_{j+2}Q_{j+3}^{2}\cdots Q_{j}^{t-j-1}$,

$D \sim Q_{j+1}Q_{j+2}\cdots Q_t$, $A_j \sim Q_j$.)

(4') Set $B \leftarrow E$, $A \leftarrow D$, $j \leftarrow j+1$, and go to (2) .

Note that D and E = B/D are computed in both versions, but in different ways. Algorithm S appears to require slightly less computation than Horowitz' version, but its main virtue seems to be that it can be easily adapted for squarefree factorization over finite fields (which are of prime rather than zero characteristic), whereas it appears to be rather difficult to adapt Horowitz' version for this problem. Algorithms for the finite field case are discussed in [MUS71].

8. Hensel algorithms

The algorithms of this section are based on the classical theory of p-adic fields, which was first investigated by Hensel about 1900. The application of Hensel's constructions to practical factorization of polynomials was suggested by Zassenhaus [ZAS69]. The next two algorithms are based, however, on Van der Waerden's presentation of Hensel's Lemma (Reducibility Criterion) ([VDW49], pp. 248-250).

a. Algorithm S (Solution of a polynomial equation).

Let E be a commutative ring with identity. Given $A,B,S,T,U \in E[x]$ such that $lc(A)$ is a unit of E and $AS + BT = 1$, this algorithm computes $Y,Z \in E[x]$ such that $AY + BZ = U$ and $\deg(Z) < \deg(A)$.

(1) Set $V \leftarrow TU$.

(2) Using Algorithm D of §6, compute $Q,Z \in E[x]$ such that

$V = AQ + Z$, $\deg(Z) < \deg(A)$.

(3) Set $Y \leftarrow SU + BQ$ and exit. (Then $AY + BZ = A(SU + BQ) + B(TU - AQ) = (AS + BT)U = U$).

Theorem S. Under the assumptions of Algorithm S, the polynomials Y and Z are uniquely determined.

Proof: Let $AY_1 + BZ_1 = U$ with $\deg(Z_1) < \deg(A)$. Then $AY_1 + BZ_1 = AY + BZ$, which may be written

$$A(Y_1 - Y) = B(Z - Z_1). \tag{1}$$

Upon multiplying both sides by T and adding $AS(Z - Z_1)$ to both sides, we obtain

$$A[S(Z - Z_1) + T(Y_1 - Y)] = (AS + BT)\ (Z - Z_1) = Z - Z_1.$$

Unless the polynomial in brackets is zero, the degree of the product on the left side is $\geq \deg(A)$, since $lc(A)$ is a unit. But $\deg(Z - Z_1) < \deg(A)$, so we conclude that $Z = Z_1$ and by (1) we then have $A(Y_1 - Y) = 0$, which, with the fact that lc(A) is a unit, implies $Y_1 = Y$.

b. Linear Hensel construction.

The following lemma will be required in the proof of the next algorithm.

Lemma 1. Let D be a commutative ring with identity and $a, b \in D$. If a is a unit modulo b then, for any positive j, a is a unit modulo b^j.

Proof: For some $s \in D$ we have $as \equiv 1 \pmod{b}$. Let $j > 1$; we may assume by induction that $as^* \equiv 1 \pmod{b^{j-1}}$ for some $s^* \in D$. Hence there exist $t, t^* \in D$ such that $as + bt = 1$, $as^* + b^{j-1}t^* = 1$. Therefore

$$asb^{j-1} + b^j t = b^{j-1},$$

$$1 = as^* + b^{j-1}t^* = as^* + (asb^{j-1} + b^j\ t)t^*,$$

$$= a(s^* + sb^{j-1}t^*) + b^j\ tt^*,$$

$$as^+ \equiv 1 \pmod{b^j},$$

where $s^+ = s^* + sb^{j-1}t^*$. Thus a is a unit modulo b^j.

Algorithm H. (Hensel method for constructing a factorization mod p^j from a given factorization mod p). Let D and E be commutative rings with identities, $p \in D$, and h be a homomorphism of D onto E with kernel (p). This algorithm takes as inputs p; a positive integer j; $C \in D[x]$; and $\bar{A}, \bar{B}, \bar{S}, \bar{T} \in E[x]$ such that $lc(\bar{A})$ is a unit of E, $h(C) = \bar{A}\bar{B}$, and $\bar{A}\bar{S} + \bar{B}\bar{T} = 1$. The outputs are $A, B \in D[x]$ such that $C \equiv AB \pmod{p^j}$, $h(A) = \bar{A}$, $h(B) = \bar{B}$, $\deg(A) = \deg(\bar{A})$, and lc(A) is a unit modulo p^j.

(1) Set $i \leftarrow 1$, $q \leftarrow p$ and choose $A, B \in D[x]$ such that $h(A) = \bar{A}$, $h(B) = \bar{B}$, $\deg(A) = \deg(\bar{A})$.

(2) (Now $A, B \in D[x]$, $C \equiv AB \pmod{q}$, $h(A) = \bar{A}$, $h(B) = \bar{B}$, $\deg(A) = \deg(\bar{A})$ and lc(A) is a unit modulo q.) If $i = j$, exit.

(3) Set $U \leftarrow (C - AB)/q$, $\bar{U} \leftarrow h(U)$. (Since $C \equiv AB \pmod{q}$, we know $U \in D[x]$, hence $\bar{U} \in E[x]$.) Using Algorithm S with inputs $\bar{A}, \bar{B}, \bar{S}, \bar{T}, \bar{U}$ solve $\bar{A}\bar{Y} + \bar{B}\bar{Z} = \bar{U}$ for $\bar{Y}, \bar{Z} \in E[x]$ such that $\deg(\bar{Z}) < \deg(\bar{A})$.

(4) Choose $Y,Z \in D[x]$ such that $h(Y) = \bar{Y}$, $h(Z) = \bar{Z}$, $\deg(Z) = \deg(\bar{Z})$. (Thus $AY + BZ \equiv U \pmod{p}$ and $\deg(Z) < \deg(A)$.)

(5) Set $A \leftarrow A + qZ$, $B \leftarrow B + qY$, $i \leftarrow i + 1$, $q \leftarrow qp$, and go to (2).

The assertions at step (2) obviously hold for the first execution of the step. To show that they still hold for subsequent executions, let $A^* = A + qZ$, $B^* = B + qY$, $q^* = qp$. Then

$$\begin{aligned} C - A^*B^* &= C - AB - q(AY + BZ) - q^2YZ \\ &= q(U - AY - BZ) - q^2YZ \\ &\equiv 0 \pmod{qp}, \end{aligned}$$

i.e. $C \equiv A^*B^* \pmod{q^*}$. Also we have $h(A^*) = h(A) = \bar{A}$, $h(B^*) = h(B) = \bar{B}$, $\deg(A^*) = \deg(A) = \deg(\bar{A})$ and $lc(A^*) = lc(A)$; since $lc(A)$ is a unit modulo p, $lc(A^*)$ is a unit modulo q^*, by Lemma 1. Thus from step (5) we return to step (2) with all of the assertions still valid.

The following theorem, concerning the uniqueness of the polynomials computed by Algorithm H, will be of central importance in the proof of the validity of later algorithms.

Theorem H. Let D be a commutative ring with identity, p be an element of D which is not a zero-divisor, and j be a positive integer. Let A,B,A_1, $B_1 \in D[x]$ satisfy

a. $A_1B_1 \equiv AB \pmod{p^j}$

b. $\deg(A_1) = \deg(A)$, $lc(A_1) \equiv lc(A) \pmod{p^j}$;

c. $A_1 \equiv A$ and $B_1 \equiv B \pmod{p}$;

d. $lc(A)$ is a unit mod p.

Then $A_1 \equiv A$ and $B_1 \equiv B \pmod{p^j}$.

Proof: From c we have the conclusion when $j = 1$. Let $j > 1$. From a, we have $A_1B_1 \equiv AB \pmod{p^{j-1}}$, and from b, $lc(A_1) \equiv lc(A) \pmod{p^{j-1}}$, so we may assume by induction that $A_1 \equiv A$ and $B_1 \equiv B \pmod{p^{j-1}}$. Hence there exist

$Y,Z \in D[x]$ such that $A_1 = A + p^{j-1}Z$, $B_1 = B + p^{j-1}Y$. Thus

$$A_1B_1 = AB + p^{j-1}(AY + BZ) + p^{2j-2}YZ,$$

$$0 \equiv p^{j-1}(AY + BZ) \pmod{p^j}.$$

From this congruence and the assumption that p is not a zero-divisor follows

$$AY + BZ \equiv 0 \pmod{p}.$$

Also, by c we have $\deg(Z) \leq \deg(A)$ and in fact $p|lc(Z)$. Hence by Theorem S applied to the ring $D/(p)$ we have $Y \equiv Z \equiv 0 \pmod{p}$, from which we obtain the conclusion of the theorem.

c. Quadratic Hensel construction.

In this section we discuss a variation on Algorithm H which was first proposed by Zassenhaus [ZAS69]. Given a factorization over a ring D modulo p, this algorithm computes factorizations modulo p^2, p^4, p^8,... in successive iterations. In the case D = Z, the algorithm turns out to be much more efficient than algorithm H for factoring polynomials with large coefficients. The algorithm also has another, perhaps more important, virtue. It allows the development of a "generalized Hensel algorithm" (Algorithm G in the next section) which is the basis of the practical method of factorization of multivariate polynomials described in §4. The algorithm is based on a version discussed by Knuth [KNU69, pp. 398 and 546].

Algorithm Q (Quadratic Hensel Algorithm). Let D and E be commutative rings with identities, $p \in D$, and h a homomorphism from D onto E with kernel (p). The inputs to the algorithm are p, a positive integer j; $C \in D[x]$; and $\bar{A},\bar{B},\bar{S},\bar{T} \in E[x]$ such that $lc(\bar{A})$ is a unit of E, $h(C) = \bar{A}\bar{B}$ and $\bar{A}\bar{S} + \bar{B}\bar{T} = 1$. The outputs are $A,B,S,T \in D[x]$ such that $lc(A)$ is a unit modulo p^j, $C \equiv AB$ and $AS + BT \equiv 1 \pmod{p^j}$, $h(A) = \bar{A}$, $h(B) = \bar{B}$ and $\deg(A) = \deg(\bar{A})$.

(1) Set $i \leftarrow 1$, $q \leftarrow p$ and choose $A,B,S,T \in D[x]$ such that $h(a) = \bar{A},\ldots,h(T) = \bar{T}$ and $\deg(A) = \deg(\bar{A})$.

(2) (Now $A, B, S, T \in D[x]$, $lc(A)$ is a unit mod q, $C \equiv AB$ and $AS + BT \equiv 1$ (mod q), $h(A) = \bar{A}$, $h(B) = \bar{B}$ and $\deg(A) = \deg(\bar{A})$.) If $i \geq j$, exit.

(3) Set $U \leftarrow (C - AB)/q$. (Since $C \equiv AB$ (mod q) we know $U \in D[x]$.) Using Algorithm S of §8a with inputs A,B,S,T,U, solve the congruence $AY + BZ \equiv U$ (mod q) for $Y, Z \in D[x]$ such that $\deg(Z) < \deg(A)$.

(4) Set $A^* \leftarrow A + qZ$, $B^* \leftarrow B + qY$. (Thus

$$\begin{aligned} C - A^*B^* &= C - AB - q(AY + BZ) - q^2YZ \\ &= q(U - AY - BZ) - q^2YZ \\ &\equiv 0 \pmod{q^2}; \end{aligned}$$

furthermore $h(A^*) = h(A)$, $h(B^*) = h(B)$; and, since $\deg(Z) < \deg(A)$, $\deg(A^*) = \deg(A) = \deg(\bar{A})$ and $lc(A^*) = lc(A)$. By Lemma 1 of §8b, $lc(A^*)$ is a unit modulo q^2.)

(5) Set $U_1 \leftarrow (A^*S + B^*T - 1)/q$. Using Algorithm S with inputs A,B,S,T,U_1, solve the congruence $AY_1 + BZ_1 \equiv U$ (mod q) for $Y_1, Z_1 \in D[x]$ such that $\deg(Z_1) < \deg(A)$.

(6) Set $S^* \leftarrow S - qY_1$, $T^* \leftarrow T - qZ_1$. (Thus

$$\begin{aligned} A^*S^* + B^*T^* &= A^*(S - qY_1) + B^*(T - qZ_1) \\ &= A^*S + B^*T - q(A^*Y_1 + B^*Z_1) \\ &= 1 + q(U_1 - A^*Y_1 - B^*Z_1) \\ &\equiv 1 + q(U_1 - AY_1 - BZ_1) \pmod{q^2} \\ &\equiv 1 \pmod{q^2}.) \end{aligned}$$

(7) Replace i, q, A, B, S, T by $2i, q^2, A^*, B^*, S^*, T^*$ and go to (2).

d. Generalized Hensel construction.

This section develops generalizations of Algorithm Q and Theorem H in which the kernel of the homomorphism h of D onto E may be generated by more than one element. These generalizations form the theoretical basis for the practical multivariate factorization described in §4.

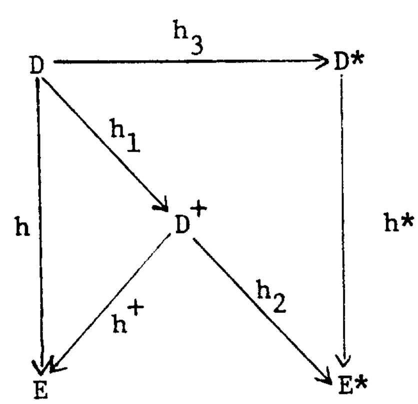
<u>Lemma 1</u>. Let R,S,T be commutative rings, ν be a homomorphism of R onto S and α be a homomorphism of R onto T. Let $Ker(\nu) \subset Ker(\alpha)$. Then:

a. there exists a (unique) homomorphism β from S onto T such that $\beta \circ \nu = \alpha$

b. $Ker(\beta) = \nu(Ker(\alpha))$.

This lemma can be derived as a corollary to the so-called "Rectangle Theorem" for rings (see [GOL70], p. 120), or can easily be proved directly.

<u>Algorithm G</u> (Generalized Hensel Algorithm). Let D and E be commutative rings with identities, $p_1,\ldots,p_n$ be elements of D and h be a homomorphism from D onto E with kernel $p = (p_1,\ldots,p_n)$. The inputs to this algorithm are $p_1,\ldots,p_n$; positive integers $j_1,\ldots,j_n$; $C \in D[x]$; and $\bar{A},\bar{B},\bar{S},\bar{T} \in E[x]$ such that $lc(\bar{A})$ is a unit of E, $h(C) - \bar{A}\bar{B}$ and $\bar{A}\bar{S} + \bar{B}\bar{T} = 1$. The outputs are A,B,S, $T \in D[x]$ such that $lc(A)$ is a unit modulo $m = (m_1,\ldots,m_n)$, where $m_i = p_i^{j_i}$; $C \equiv AB$ and $AS + BT \equiv 1 \pmod{m}$; $h(A) = \bar{A}$, $h(B) = \bar{B}$, and $\deg A = \deg \bar{A}$.



Remark: The above diagram will aid in following the statement and proof of the algorithm. (The definitions and proofs given in the algorithm will show that the diagram commutes.)

(1) If n = 1, apply Algorithm Q to $p_1, j_1, C, \bar{A}, \bar{B}, \bar{S}, \bar{T}$, obtaining $A,B,S,T \in D[x]$ satisfying the required conditions. Exit.

(2) Let h_1 be a homomorphism defined on D with kernel (p_1) and let $D^+ = h_1(D)$.

Let h^+ be the homomorphism of D^+ onto E such that $h^+ \circ h_1 = h$. (The existence of h^+ is guaranteed by Lemma 1; also $\mathrm{Ker}(h^+) = h_1(\mathrm{Ker}(h)) = h_1((p_1,\ldots,p_n)) = (h_1(p_1),\ldots,h_1(p_n)) = (0,h_1(p_2),\ldots,h_1(p_n)) = (h_1(p_2),\ldots,h_1(p_n))$.) Set $p_i^+ \leftarrow h_1(p_i)$ and let $m_i^+ = (p_i^+)^{j_i}$, for $2 \leq i \leq n$. Also set $C^+ \leftarrow h_1(C)$. Working in D^+ and E, apply this algorithm recursively to $p_1^+,\ldots,p_n^+$, $j_2,\ldots,j_n$, C^+, $\bar{A},\bar{B},\bar{S},\bar{T}$, obtaining outputs $A^+,B^+,S^+,T^+ \in D^+[x]$. (Thus $\mathrm{lc}(A^+)$ is a unit mod $m^+ = (m_2^+,\ldots,m_n^+)$, $C^+ \equiv A^+B^+$ and $A^+S^+ + B^+T^+ \equiv 1 \pmod{m^+}$, $h^+(A^+) = \bar{A}$, $h^+(B^+) = \bar{B}$ and $\deg(A^+) = \deg(\bar{A})$.)

(3) Let h_2 be a homomorphism defined on D^+ with kernel $(m_2^+,\ldots,m_n^+)$ and let $E^* = h_2(D^+)$. Set $\bar{A}^* \leftarrow h_2(A^+)$, $\bar{B}^* \leftarrow h_2(B^+)$, $\bar{S}^* \leftarrow h_2(S^+)$, $\bar{T}^* \leftarrow h_2(T^+)$. (Thus $\mathrm{lc}(\bar{A}^*)$ is a unit of E^*, $\deg(\bar{A}^*) = \deg(A^+)$, $h_2(h_1(C)) = \bar{A}^*\bar{B}^*$ and $\bar{A}^*\bar{S}^* + \bar{B}^*\bar{T}^* = 1$).

(4) Let h_3 be a homomorphism defined on D with kernel $(m_2,\ldots,m_n)$ and let $D^* = h_3(D)$. Let h^* be the homomorphism of D^* onto E^* such that $h^* \circ h_3 = h_2 \circ h_1$. ($\mathrm{Ker}(h_2 \circ h_1) = (p_1,m_2,\ldots,m_n)$, as will be shown below, hence Lemma 1 guarantees the existence of h^* and furthermore shows that $\mathrm{Ker}(h^*) = h_3(\mathrm{Ker}(h_2 \circ h_1)) = h_3((p_1,m_2,\ldots,m_n)) = (h_3(p_1))$.) Set $p_1^* \leftarrow h_3(p_1)$, $m_1^* \leftarrow (p_1^*)^{j_1}$, $C^* \leftarrow h_3(C)$. (Thus $h^*(C^*) = h^*(h_3(C)) = h_2(h_1(C)) = \bar{A}^*\bar{B}^*$.) Working in D^* and E^*, apply Algorithm Q to p_1^*, j_1, C^*, $\bar{A}^*,\bar{B}^*,\bar{S}^*,\bar{T}^*$, to obtain outputs $A^*,B^*,S^*,T^* \in D^*[x]$. (Thus $\mathrm{lc}(A^*)$ is a unit modulo m_1^*, $C^* \equiv A^*B^*$ and $A^*S^* + B^*T^* \equiv 1 \pmod{m_1^*}$, $h^*(A^*) = \bar{A}^*$, $h^*(B^*) = \bar{B}^*$ and $\deg(A^*) = \deg(\bar{A}^*)$.)

(5) Choose $A,B,S,T \in D[x]$ such that $h_3(A) = A^*$, $h_3(B) = B^*$, $h_3(S) = S^*$, $h_3(T) = T^*$ and $\deg(A) = \deg(A^*)$. (Then $h_3(C) \equiv h_3(A)h_3(B) \pmod{h_3(m_1)}$, hence $h_3(C - AB) = h_3(P)h_3(m_1)$ for some $P \in D[x]$, hence $h_3(C - AB - Pm_1) = 0$, hence $C - AB - Pm_1 \in (m_2,\ldots,m_n)$, hence $C - AB \in (m_1,\ldots,m_n)$, hence $C \equiv AB \pmod{(m_1,\ldots,m_n)}$. Similarly, $AS + BT \equiv 1 \pmod{(m_1,\ldots,m_n)}$. Since $h_2(h_1(A)) = h^*(h_3(A)) = h^*(A^*) = \bar{A}^* = h_2(A^+)$, we have $h_1(A) \equiv A^+ \pmod{m^+}$. Hence it

follows that $h_1(A) \equiv A^+ \pmod{(h_1(p_2),\ldots,h_1(p_n))}$; i.e. $h_1(A) - A^+ \in Ker(h^+)$, hence $h^+(h_1(A)) = h^+(A^+) = \bar{A}$, and finally $h(A) = \bar{A}$. Similarly, $h(B) = \bar{B}$. Also $deg(A) = deg(A^*) = deg(\bar{A}^*) = deg(A^+) = deg(\bar{A})$. Lastly, we have $h_3(lc(A)) = lc(A^*)$, from which it is easily shown that $lc(A)$ is a unit modulo $(m_1,\ldots,m_n)$.)

The assertion in step (4) that $Ker(h_2 \circ h_1) = (p_1,m_2,\ldots,m_n)$ may be proved as follows:

$d \in Ker(h_2 \circ h_1) \Leftrightarrow h_1(d) \in Ker(h_2) = (m_2^+,\ldots,m_n^+) \Leftrightarrow h_1(d) =$

$h_1(d_2)h_1(m_2) + \ldots + h_1(d_n)h_1(m_n)$ for some $d_2,\ldots,d_n \in D$

$\Leftrightarrow h_1(d - d_2m_2 - \ldots - d_nm_n) = 0$ for some $d_2,\ldots,d_n \in D$

$\Leftrightarrow d - d_2m_2 - \ldots - d_nm_n \in Ker(h_1) = (p_1)$ for some $d_2,\ldots,d_n \in D$

$\Leftrightarrow d - d_2m_2 - \ldots - d_nm_n = d_1p_1$ for some $d_1,\ldots,d_n \in D$

$\Leftrightarrow d \in (p_1,m_2,\ldots,m_n)$.

Before proceeding to generalize Theorem H of §8b we shall prove a lemma which generalizes Lemma 1 of that section.

<u>Lemma 2</u>. Let D be a commutative ring with identity, $p_1,\ldots,p_n \in D$, $m_1 = p_1^{j_1},\ldots,m_n = p_n^{j_n}$ for some positive integers $j_1,\ldots,j_n$, $\mathcal{p} = (p_1,\ldots,p_n)$, $\mathcal{m} = (m_1,\ldots,m_n)$. Let $a \in D$ be a unit modulo $\mathcal{p}$. Then a is a unit modulo $\mathcal{m}$.

Proof: We use the notation of Algorithm G and divide the proof into steps corresponding to those of Algorithm G:

(1) If $n = 1$ then Lemma 1 of §8b applies and we are done.

(2) Assume $n > 1$ and let $a^+ = h_1(a)$. Since a is a unit modulo $\mathcal{p}$, there is a $b \in D$ such that $ab - 1 \in \mathcal{p}$, hence $a^+b^+ - 1 \in \mathcal{p}^+$, where $b^+ = h_1(b)$ and $\mathcal{p}^+ = (h_1(p_2),\ldots,h_1(p_n))$. Hence a^+ is a unit modulo $\mathcal{p}^+$, and we may thus assume, by induction on n, that a^+ is a unit modulo $\mathcal{m}^+$.

(3) Let $\bar{a}^* = h_2(a^+)$. From the conclusion of step (2), $\bar{a}^*$ is a unit of E^*.

(4) Let $a^* = h_3(a)$. Then $h^*(a^*) = \bar{a}^*$, i.e. a^* is a unit modulo p_1^*. Applying Lemma 1 of §8b, we find that a^* is a unit modulo m_1^*.

(5) Suppose $a^*b^* \equiv 1 \pmod{m_1^*}$. Choose $b \in D$ such that $h_3(b) = b^*$. Then $h_3(a)h_3(b) - 1 = h_3(d)h_3(m_1)$ for some $d_1 \in D$, hence $h_3(ab - 1 - dm_1) = 0$, hence $ab - 1 - d_1m_1 \in (m_2,m_3,\ldots,m_n)$, hence $ab \equiv 1 \pmod{m}$.

Theorem G. Let D be a commutative ring with identity; $p_1,\ldots,p_n$ be elements of D which are not zero-divisors; $m_1 = p_1^{j_1},\ldots,m_n = p_n^{j_n}$ for some positive integers $j_1,\ldots,j_n$; $p = (p_1,\ldots,p_n)$; $m = (m_1,\ldots,m_n)$. Let A,B,A_1, $B_1 \in D[x]$ satisfy

a. $A_1B_1 \equiv AB \pmod{m}$;

b. $\deg(A_1) = \deg(A)$ and $\mathrm{lc}(A_1) \equiv \mathrm{lc}(A) \pmod{m}$;

c. $A_1 \equiv A$ and $B_1 \equiv B \pmod{p}$;

d. $\mathrm{lc}(A)$ is a unit modulo p.

Then $A_1 \equiv A$ and $B_1 \equiv B \pmod{m}$.

Proof: Again we use the notation of Algorithm G and divide the proof into steps corresponding to those of Algorithm G:

(1) If $n = 1$, Theorem H of §8b applies and we are done.

(2) Assume $n > 1$. Let A_1^+,B_1^+,A^+,B^+ be the images under h_1 of A_1,B_1,A,B. Let $p^+ = (h_1(p_2),\ldots,h_1(p_n))$. By c, $A_1 - A = d_1p_1 + \ldots + d_np_n$ for some $d_i \in D[x]$. Hence $h_1(A_1) - h_1(A) = h_1(d_2)h_1(p_2) + \ldots + h_1(d_n)h_1(p_n)$, hence $A_1^+ - A^+ \in p^+$. In this way we can prove

c^+. $A_1^+ \equiv A^+$ and $B_1^+ \equiv B^+ \pmod{p^+}$,

d^+. $\mathrm{lc}(A^+)$ is a unit modulo p^+,

a^+. $A_1^+B_1^+ \equiv A^+B^+ \pmod{m^+}$,

and, using Lemma 2, the second part of

b^+. $\deg(A_1^+) = \deg(A^+)$ and $\mathrm{lc}(A_1^+) \equiv \mathrm{lc}(A^+) \pmod{m^+}$.

The first part of b^+ follows from b and d. We may now assume, by induction on n, that $A_1^+ \equiv A^+$ and $B_1^+ \equiv B^+ \pmod{m^+}$.

(3) Let $\bar{A}_1^*,\bar{B}_1^*,\bar{A}^*,\bar{B}^*$ be the images under h_2 of A_1^+,B_1^+,A^+,B^+. Then, from the conclusion of step (2), $\bar{A}_1^* = \bar{A}^*$ and $\bar{B}_1^* = \bar{B}^*$.

(4) Let A_1^*, B_1^*, A^*, B^* be the images under h_3 of A_1, B_1, A, B. From the equation $h_2 \circ h_1 = h^* \circ h_3$, we have

$$h^*(A_1^*) = h^*(h_3(A_1)) = h_2(h_1(A_1)) = \bar{A}_1^* = \bar{A}^*$$

$$= h_2(h_1(A)) = h^*(h_3(A)) = h^*(A^*).$$

In this manner we obtain

c*. $A_1^* \equiv A^* \pmod{p_1^*}$ and $B_1^* \equiv B^* \pmod{p_1^*}$.

From a,b,c and the definition of h_3 and m_1^*, we have

a*. $A_1^* B_1^* \equiv A^* B^* \pmod{m_1^*}$

b*. $\deg(A_1^*) = \deg(A)$ and $lc(A_1^*) \equiv lc(A^*) \pmod{m_1^*}$.

We may prove

d*. $lc(A^*)$ is a unit modulo p_1^*

as follows: From d^+ and Lemma 2, we know that $lc(A^*)$ is a unit modulo m^+. Furthermore, $lc(\bar{A}^*) = h_2(lc(A^+))$, hence $lc(\bar{A}^*)$ is a unit of E^*. Finally, $h^*(lc(A^*)) = lc(\bar{A}^*)$, hence d* follows. Now Theorem H implies $A_1^* \equiv A^*$ and $B_1^* \equiv B \pmod{m_1^*}$.

(5) From the conclusion of step (4), $h_3(A_1 - A_1)$ is a multiple of $h_3(m_1)$, say $h_3(d_1)$. Thus $h_3(A_1 - A - dm_1) = 0$, hence $a_1 - A - d_1 m_1 = d_2 m_2 + \ldots + d_n m_n$ for some $d_2, \ldots, d_n \in D[x]$. Hence $A_1 \equiv A \pmod{m}$ and similarly, $B_1 \equiv B \pmod{m}$.

9. Corresponding factorizations and refinements of factorizations

a. Construction of a corresponding factorization modulo $\mathfrak{m}$.

In order to make use of any of the Hensel algorithms discussed in the previous section, we must be able to find polynomials $\bar{S}$, $\bar{T} \in E[x]$ which satisfy $\bar{A}\bar{S} + \bar{B}\bar{T} = 1$ for factors $\bar{A}$ and $\bar{B}$ of $h(C) \in E[x]$. Sufficient conditions for this are that E be a field and $\bar{A}$ and $\bar{B}$ be relatively prime over E, for then $\bar{S}$ and $\bar{T}$ can be computed by the Extended Euclidean Algorithm. We are thus led to the following abstract algorithm which provides the theoretical basis for the algorithms of sections 3f and 4g. In the algorithm we use the fact that if the input $\bar{A}$ to Algorithm H is monic then, in step (1) of Algorithm H, A can be chosen to be monic, in which case the final output A is monic.

Algorithm C (Construction of a sequence of factors modulo $\mathfrak{m}$ corresponding to a given sequence of factors modulo $\mathfrak{p}$). Let D be a commutative ring with identity, $p_1,\ldots,p_n \in D$, and h be a homomorphism of D onto a field E with kernel $\mathfrak{p} = (p_1,\ldots,p_n)$. The inputs to the algorithm are $p_1,\ldots,p_n$; positive integers $j_1,\ldots,j_n$; $C \in D[x]$ for which $h(C)$ is squarefree; and $G_1,\ldots,G_t$, a sequence of monic polynomials over E such that $h(C) = lc(h(C))\, G_1 \cdots G_t$.

The outputs are U, F_1, ..., $F_t \in D[x]$ such that $C \equiv UF_1 \ldots F_t \pmod{\mathfrak{m}}$, where $\mathfrak{m} = (p_1^{j_1},\ldots,p_n^{j_n})$; $h(F_i) = G_i$, $\deg(F_i) = \deg(G_i)$, and F_i is monic, for $i = 1,..,t$.

(1) Set $\bar{C} \leftarrow h(C)$, $i \leftarrow 1$.

(2) (Now we have:

a. $C_0 \equiv CF_1 \ldots F_{i-1} \pmod{\mathfrak{m}}$, where C_0 was the initial value of C;

b. $h(F_k) = G_k$, $\deg(F_k) = \deg(G_k)$, and F_k is monic for $k = 1,\ldots,i-1$;

c. $\bar{C} = h(C) = lc(h(C))G_iG_{i+1} \ldots G_t$;

d. $\bar{C}$ is squarefree.)

Set $\bar{A} \leftarrow G_i$, $\bar{B} \leftarrow \bar{C}/\bar{A}$. (Thus $h(C) = \bar{A}\bar{B}$, $\bar{A}$ is monic, and $\bar{A}$ and $\bar{B}$ are relatively prime over E, by d.)

(3) Using the Extended Euclidean Algorithm, obtain $\bar{S}$ and $\bar{T}$ over E such that $\bar{A}\bar{S} + \bar{B}\bar{T} = 1$.

(4) Apply Algorithm G to $p_1,\ldots,p_n$, $j_1,\ldots,j_n$, C, $\bar{A}$, $\bar{B}$, $\bar{S}$, $\bar{T}$ and let A and B be the output. (Thus $A,B \in D[x]$, $C \equiv AB \pmod{\mathfrak{m}}$, $h(A) = \bar{A}$, $h(B) = \bar{B}$, $\deg(A) = \deg(\bar{A})$, and we may assume that A is monic, as noted above.)

(5) Set $F_i \leftarrow A$, $C \leftarrow B$, $\bar{C} \leftarrow \bar{B}$, $i \leftarrow i+1$. (Thus conditions a,b,c, and d remain valid).

(6) If $i \leq t$, go to (2).

(7) Set $U \leftarrow C$ and exit.

In §9c we shall prove a key theorem about the output of Algorithm C. Some of the definitions and notations used in the statement and proof of this theorem and later algorithms and theorems involve the "multiset" concept recently introduced by Knuth [KNU69, Exercise 4.6.3-19].

b. Multisets

A multiset is like a set, but may contain identical elements repeated a finite number of times. If $\mathcal{A}$ and $\mathcal{B}$ are multisets we define new multisets $\mathcal{A} \uplus \mathcal{B}$, $\mathcal{A} \cup \mathcal{B}$, $\mathcal{A} \cap \mathcal{B}$, $\mathcal{A} - \mathcal{B}$ as follows: an element x occurring exactly a times in $\mathcal{A}$ and b times in $\mathcal{B}$ occurs exactly

$a + b$ times in $\mathcal{A} \uplus \mathcal{B}$

$\max(a,b)$ times in $\mathcal{A} \cup \mathcal{B}$

$\min(a,b)$ times in $\mathcal{A} \cap \mathcal{B}$

$\max(a-b,0)$ times in $\mathcal{A} - \mathcal{B}$

If $\mathcal{A}$ is a finite multiset with elements from a set on which a commutative addition operation is defined, then by $\Sigma\mathcal{A}$ we shall mean a sum in which an element x is included as a term exactly as often as it occurs in $\mathcal{A}$. We define $\Sigma\phi = 0$ where ϕ denotes the empty multiset. Assuming commutative multiplication, $\Pi\mathcal{A}$ is defined analogously, with the convention $\Pi\phi = 1$.

If $\mathcal{A} = \{a_1,\ldots,a_n\}$ is a finite multiset with elements from a set on which a function f is defined, then $f(\mathcal{A})$ is the multiset $\{f(a_1),\ldots,f(a_n)\}$.

As an example of this notation, let $\mathcal{A} = \{x^2 + 1,\ x - 1,\ x^3 - 3x + 7,\ x^2 + 1\}$ and f = deg, the degree function; then

$$\deg(\mathcal{A}) = \{2,1,3,2\},$$

$$\deg(\Pi\mathcal{A}) = \Sigma \deg(\mathcal{A}) = \Sigma\{2,1,3,2\} = 2+1+3+2 = 8.$$

In the context of factorization, if C is an element of a UFD and $\mathcal{J}$ is a multiset of prime elements such that $C = \Pi\mathcal{J}$, then it is convenient to refer to $\mathcal{J}$ as a complete factorization of C.

c. <u>Refinements of factorizations</u>

Let D be a commutative ring with identity and $\mathcal{A} = \{A_1,\ldots,A_r\}$ and $\mathcal{B}$ be multisets of polynomials over D for which $\Pi\mathcal{A} = e\Pi\mathcal{B}$ for some $e \in D$. Then $\mathcal{B}$ is said to be a refinement of $\mathcal{A}$ if there exists a partition $\mathcal{B} = \mathcal{B}_1 \uplus \cdots \uplus \mathcal{B}_r$ such that

$$A_j = e_j\Pi\mathcal{B}_j \quad \text{for some } e_j \in D,\ 1 \le j \le r.$$

(This definition is a restatement of one given in §3i, using multiset notation.) If E is another ring and h is a homomorphism of D onto E with kernel $\mathfrak{m}$, then $\mathcal{B}$ is said to be a <u>modulo $\mathfrak{m}$ refinement</u> of $\mathcal{A}$ if $h(\mathcal{B})$ is a refinement of $h(\mathcal{A})$.

Obviously, if D is a UFD and $\mathcal{B}$ is a complete factorization of a polynomial C over D, then $\mathcal{B}$ is a refinement of any other factorization of

C. In particular, if D and E are UFDs, h is a homomorphism of D onto E with kernel $\mathfrak{m}$, C is a polynomial over D and G is a complete factorization of pp(h(C)), then G is a refinement of h(F), where F is the complete factorization of C.

Theorem C. Given the assumptions of Algorithm C, let $G = \{G_1,\ldots,G_t\}$ and $F = \{F_1,\ldots,F_t\}$. Also assume $p_1,\ldots,p_n$ are not zero-divisors, $lc(C) \notin (p_1,\ldots,p_n)$, and G is a refinement of $h(\mathcal{J})$, where $\mathcal{J}$ is a complete factorization of C. Then F is a modulo $\mathfrak{m}$ refinement of $\mathcal{J}$.

Proof: We have, from the algorithm, $C \equiv U\Pi F \pmod{\mathfrak{m}}$. Since ΠF is monic, $\deg(C) = \deg(U) + \deg(\Pi F)$. Also, since $lc(C) \notin \mathcal{P} = (p_1,\ldots,p_n)$, $\deg(C) = \deg(h(C)) = \deg(\Pi G) = \Sigma \deg(G) = \Sigma \deg(F) = \deg(\Pi F)$. Hence $\deg(U) = 0$ and $U \equiv lc(C) \pmod{\mathfrak{m}}$, so

$$\Pi\mathcal{J} = C \equiv lc(C)\Pi F \pmod{\mathfrak{m}} \qquad (1)$$

Now let $\mathcal{J} = \{\mathcal{J}_1,\ldots,\mathcal{J}_r\}$. By assumption, there is a partition $G = H_1 \uplus \cdots \uplus H_r$ such that $h(\mathcal{J}_j) = e_j\Pi H_j$ for some $e_j \in E$, $1 \le j \le r$. Partition F into $K_1 \uplus \cdots \uplus K_r$ such that K_j contains the F_i corresponding to the G_i in H_j. Define

$$A = \mathcal{J}_1$$
$$B = C/\mathcal{J}_1$$
$$A_1 = lc(A)\Pi K_1$$
$$B_1 = lc(B)\Pi(F-K_1)$$

Then from (1) we have $A_1B_1 = lc(C)\Pi F \equiv C \pmod{\mathfrak{m}}$, so $A_1B_1 \equiv AB \pmod{\mathfrak{m}}$. We will show that the other assumptions of Theorem G of §8d are satisfied: First, since $lc(C) \notin \mathcal{P}$ and $lc(A) | lc(C)$ we have $lc(A) \notin \mathcal{P}$, so $\deg(h(A)) = \deg(A)$ and $h(lc(A)) = lc(h(A))$. Thus $h(A_1) = h(lc(A))\, h(\Pi K_1) = lc(h(A))\Pi H_1 = h(A)$, so $A_1 \equiv A \pmod{\mathcal{P}}$. Similarly, $B_1 \equiv B \pmod{\mathcal{P}}$. Next, $\deg(A_1) = \deg(\Pi K_1) =$

$\Sigma \deg(K_1) = \Sigma \deg(H_1) = \deg(\Pi H_1) = \deg(h(A)) = \deg(A)$. Finally, $lc(C) \notin \mathfrak{p}$ implies that $lc(C)$ is a unit modulo $\mathfrak{p}$, since $D/\mathfrak{p}$ is a field. From Theorem G we therefore conclude that $A \equiv A_1 \pmod{\mathfrak{m}}$, hence that $\mathcal{J}_1 \equiv lc(\mathcal{J}_1)\Pi K_1 \pmod{\mathfrak{m}}$. By symmetry, $\mathcal{J}_j \equiv lc(\mathcal{J}_j)\Pi K_j \pmod{\mathfrak{m}}$, and this proves that $F = K_1 \uplus \ldots \uplus K_r$ is a modulo $\mathfrak{m}$ refinement of $\mathcal{J} = \{\mathcal{J}_1, \ldots, \mathcal{J}_r\}$.

10. Construction of a complete factorization

In preparation for the discussion of an abstract algorithm for this final stage of the factorization process, we first review in §10a some concepts and terminology relating to the use of homomorphisms in practical computation, and in §10b the concept of an "R-factorable" polynomial, a generalization of the property of polynomials over the integers that a bound can be given for the coefficients of factors.

a. Homomorphisms and sets of representatives

Throughout this section we assume that D and E are sets and $h: D \to E$ is a mapping of D onto E. The set $P = \{h^{-1}(e): e \in E\}$, where $h^{-1}(e) = \{d \in D: h(d) = e\}$, is a partition of D. Let R be a subset of D such that for each set $S \in P$, $R \cap S$ contains exactly one element. Then R is a (complete) set of representatives of P. In other words, for each $e \in E$ there is a unique $d \in R$ such that $h(d) = e$. We assume this property of R in what follows.

We denote by h_R the restriction of h to R. The map $h_R: R \to E$ is one-to-one. We denote the inverse map of E onto R by h_R^{-1}.

We now assume that $(D,+,\cdot)$ and $(E,+_1,\cdot_1)$ are commutative rings with identity, and that h is a homomorphism of D onto E. We recall that the residue class ring $D/\text{Ker}(h) = \{d+\text{Ker}(h): d \in D\}$ is isomorphic to E. The partition P of D, as defined above, is in this case the set $D/\text{Ker}(h)$.

The set R of representatives of $D/\text{Ker}(h)$ may be made into a ring as follows. Let $\hat{h} = h_R^{-1} \circ h$, and for $a,b \in R$ define

$$a +_2 b = \hat{h}(a+b), \quad a \cdot_2 b = \hat{h}(ab). \tag{1}$$

We know that $h_R^{-1}: E \to R$ is one-to-one and onto. Let $a_1,b_1 \in E$ and

$a = h_R^{-1}(a_1)$, $b = h_R^{-1}(b_1)$. Then

$$h_R^{-1}(a_1 +_1 b_1) = h_R^{-1}(h(a) +_1 h(b)) = h_R^{-1}(h(a + b))$$
$$= \hat{h}(a+b) = a +_2 b = h_R^{-1}(a_1) +_2 h_R^{-1}(b_1),$$

and similarly $h_R^{-1}(a_1 \cdot_1 b_1) = h_R^{-1}(a_1) \cdot_2 h_R^{-1}(b_1)$. From these relations the ring axioms for $(R, +_2, \cdot_2)$ is a ring isomorphic to $(E, +_1, \cdot_1)$ under h_R^{-1}.

Furthermore, $\hat{h}$ is a homomorphism of D onto R: for $a,b \in D$,

$$\hat{h}(a+b) = h_R^{-1}(h(a+b)) = h_R^{-1}(h(a) +_1 h(b))$$
$$= h_R^{-1}(h(a)) +_2 h_R^{-1}(h(b))$$
$$= \hat{h}(a) +_2 \hat{h}(b),$$

and similarly for multiplication.

The point of this is that if we can do arithmetic in the ring D (i.e. if we have algorithms for performing the operations + and $\cdot$ on symbols representing the elements of D), then we can also do arithmetic in E, provided that we also have an algorithm for $\hat{h}$: we represent the elements of E by the symbols representing the elements of R and perform addition and multiplication on these symbols according to (1).

As an example, take D = Z (integers); m a positive integer; E = Z/(m), the residue class ring of integers modulo m; and h the canonical map $n \to n+(m)$. $R = \{0,1,\ldots,m-1\}$ is a set of representatives of E. Define a map $\phi_m: Z \to Z$ by $\phi_m(n)$ = least non-negative remainder on division of n by m. Then $h_R^{-1}(n+(m)) = \phi_m(n)$ and $\hat{h} = \phi_m$. Thus if we define

$$a +_2 b = \phi_m(a+b), \quad a \cdot_2 b = \phi_m(ab) \tag{2}$$

on R then $(R, +_2, \cdot_2)$ is a ring isomorphic to E, and is the homomorphic image of Z under ϕ_m.

Now suppose we take $D = Z$, $E = \{0,1,\ldots,m-1\}$ with $+_2$ and $\cdot_2$ defined on E by (2), and $h = \phi_m$. If we take $R = E = \{0,1,\ldots,m-1\}$, then we have a particularly simple situation: h_R and h_R^{-1} are the identity map of R and $\hat{h} = h$. The same situation occurs in general when we have $E \subset D$ and take $R = E$.

When $D = Z$ and E is isomorphic to $Z/(m)$, it is often most convenient to take

$$R = \{-\left\lfloor\frac{m}{2}\right\rfloor, \ldots, 0, 1, \ldots, \left\lfloor\frac{m}{2}\right\rfloor\}$$

(with, say, $-\left\lfloor\frac{m}{2}\right\rfloor$ omitted if m is even); this will be seen to be true in the applications discussed in the following section. With $E = \{0,1,\ldots,m-1\}$ and $h = \phi_m$, as above, we have

$$h_R^{-1}(n) = \begin{cases} n, & \text{if } n < \lfloor m/2 \rfloor, \\ n-m, & \text{otherwise;} \end{cases}$$

$$\hat{h}(n) = \begin{cases} \phi_m(n), & \text{if } \phi_m(n) < \lfloor m/2 \rfloor, \\ \phi_m(n)-m, & \text{otherwise.} \end{cases}$$

Another important example is $D = F[x]$, F is a field; $E = F[x]/I$, where I is an ideal generated by a polynomial $G(x)$ of degree $n > 0$; and h = canonical homomorphism $A(x) \mapsto A(x) + I$. $R = \{A(x) \in F[x]: \deg(A) < n\}$ is a set of representatives of E. We then have

$$h_R^{-1}: A(x) + I \to A(x) \bmod G(x),$$

$$\hat{h}: A(x) \to A(x) \bmod G(x),$$

where $A(x) \bmod G(x)$ is the remainder on division by $G(x)$.

Although we have in this section distinguished between the binary operations of D and E, we shall in the remaining discussion follow the usual convention of allowing the context to determine which operation is intended; for example if $a,b \in E$ then $a \mp b$ means $a +_1 b$ where $+_1$ is the addition operation on E.

b. R-factorability

If R is a subset of an integral domain D and C is a polynomial over D, let us say that C is R-factorable if R contains the coefficients of every factor A* of $C^* = lc(C)\cdot C$ for which $\deg(A^*) \leq \lfloor \deg(A^*)/2 \rfloor$ and $lc(A^*)|lc(C)$. (Let us regard 0 to be a coefficient of every polynomial, so that $0 \in R$.)

For D = Z, if $R = \{i: |i| \leq b\cdot lc(C)\}$, where b is a bound on the coefficients of any factor of C, then of course C is R-factorable.

Lemma 1. Let D be an integral domain, R be a subset of D, and C be a polynomial over D which is R-factorable. Then any factor of C is also R-factorable.

Proof: For C = AB we shall show that A is R-factorable. Let $lc(A)\cdot A = A_1A_2$ where $\deg(A_1) \leq \lfloor \deg(A)/2 \rfloor$ and $lc(A_1)|lc(A)$. Then $A_1|lc(C)\cdot C$, $\deg(A_1) \leq \lfloor \deg(C)/2 \rfloor$ and $lc(A_1)|lc(C)$; hence R contains the coefficients of A_1, proving that A is R-factorable.

c. A general factorization algorithm

The following abstract algorithm and its proof provide the theoretical basis for the algorithms of §3i and §4j.

Algorithm P (Factoring a primitive polynomial via a factorization of its homomorphic image). Let D be a UFD and E be a commutative ring with identity. Let h be a homomorphism from D onto E with kernel $\mathfrak{m}$ and R be a set of representatives of $D/\mathfrak{m}$. The inputs are a primitive polynomial C and a multiset G of polynomials over E such that

a. C is R-factorable;

b. G is a refinement of h(F), where the multiset F is a complete factorization of C.

c. The leading coefficient of each polynomial in G is not a zero-divisor.

The output of the algorithm is a multiset F, a complete factorization of C.

(1) Set $F \leftarrow \phi$, $d \leftarrow 1$.

(2) Set $c \leftarrow lc(C)$, $\bar{c} \leftarrow h(c)$, $C^* \leftarrow c \cdot C$.

(3) If $d > \lfloor deg(C)/2 \rfloor$, set $F \leftarrow F \uplus \{C\}$ and exit.

(4) For each $H \subset G$ such that $\Sigma deg(H) = d$:

(a) Set $A_0 \leftarrow \pi H$, $A_1 \leftarrow (\bar{c}/lc(A_0))A_0$, $A^* \leftarrow h_R^{-1}(A_1)$.

(b) If $A^* | C^*$, set $B^* \leftarrow C^*/A^*$ and go to (6).

(5) Set $d \leftarrow d + 1$ and go to (3).

(6) Set $A \leftarrow pp(A^*)$ (with $lc(A) \in D_0$), $F \leftarrow F \uplus \{A\}$, $C \leftarrow B^*/lc(A)$, $G \leftarrow G-H$, and go to (2).

Note: The meaning of step (4) is that steps (4a) and (4b) are to be performed for every distinct multiset $H \subset G$ for which the sum of the degrees of the members of H is d. An efficient way of generating all of the multisets $H \subset G$ with $deg(H) = d$ is discussed in [MUS71, §1.6, §3.3].

<u>Validity proof</u>: We let C_0 be the initial value of C and prove that at the beginning of each execution of step (3) the conditions a,b, and c of the input assumptions and the following conditions all hold:

d. $C_0 = C\pi F$;

e. all $A \in F$ are prime, primitive polynomials;

f. $c = lc(C)$, $\bar{c} = h(c)$, $C^* = cC$.

g. C has no factor B such that $0 < deg(B) < d$.

These conditions hold after performing steps (1) and (2). Assume then that we arrive at step (3) with the conditions valid and that $d \leq \lfloor (deg\ C)/2 \rfloor$.

We first show that $lc(h(C)) = h(lc(C))$. Since C is R-factorable, we have both 0 and $c = lc(C)$ in R. Since h is a homomorphism, $h(0) = 0$, and thus since R is a set of representatives of D/m we cannot also have $h(c) = 0$. Thus $\deg(h(C)) = \deg(C)$ and $lc(h(C)) = h(c)$.

By f, we thus have that $lc(h(C)) = \bar{c}$.

Next we show that, for any $H \subset G$ such that $\Sigma\deg(H) = d$, the product $A_0 = \pi H$ is of degree d and $lc(A_0)|\bar{c}$. By d, we have $\deg(A_0) = \deg(\pi H) = \Sigma\deg(H) = d$. Also, for some $e \in E$, $h(C) = e\pi G = e\pi(H \uplus (G-H)) = eA_0B_0$ where $B_0 = \pi(G-H)$. $Lc(A_0)$ cannot be a zero-divisor, so $\deg(h(C)) = \deg(A_0) + \deg(B_0)$ and $\bar{c} = lc(h(C)) = e \cdot lc(A_0)lc(B_0)$. Thus $lc(A_0)|\bar{c}$.

Thus if $A_1 = (\bar{c}/lc(A_0))\, A_0$ then $A_1 \in E[x]$ and $\deg(A_1) = d$, and if $A^* = h_R^{-1}(A_1)$ then $A^* \in D[\ ]$ and $\deg(A^*) = d$.

Now consider the case in which C has no factor of degree d. For any A^* which divides C^*, $pp(A^*)$ divides C, hence C^* can have no factor of degree d. But, as we have just shown, every A^* computed in step (4a) is of degree d, so the division test in step (4b) must fail for each H. Control thus passes to step (5) where d is increased, but condition g, as well as all of the other conditions, remains valid as we return to step (3).

Now assume C does have a factor A of degree d. By f and the fact that C is primitive, we know that A is irreducible. Thus, by b there exists an $H \subset G$ such that $h(A) = e\pi H$ for some $e \in E$.

Let $A_0 = \pi H$, $A_1 = (\bar{c}/lc(A_0))A_0$ and $A^* = h_R^{-1}(A_1)$. We shall now show that $A^* = bA$, where $b = lc(C/A)$.

We have $h(bA) = h(b)\ h(A) = h(b)eA_0$. Since $e = lc(h(A))\ /\ lc(A_0)$ and $h(b) \cdot lc(h(A)) = lc(h(C)) = \bar{c}$, we have $h(bA) = (\bar{c}/lc(A_0))A_0 = A_1$. But $lc(bA) = c$, $(bA)|C^*$, $\deg(bA) \le \lfloor \deg(C)/2 \rfloor$, and C is R-factorable; hence R contains the coefficients of bA. Thus $bA = h_R^{-1}(A_1) = A^*$, as was to be shown.

We thus have pp(A*) = pp(bA) = pp(A) = A. We see that step (4) computes from H a polynomial A* such that $A^*|C^*$ and pp(A*) = A. Thus we must eventually find a factor of C of degree d: either A or some other D_0 factor of degree d.

Assume that A is the factor found; then in step (6) we put A into F. A is irreducible, so condition e remains valid. Also B*/lc(A) = (C*/A*)/lc(A) = (cC/bA)/lc(A) = C/A, so the new value of C satisfies condition d. By Lemma 1, condition a remains valid and conditions b,c and g obviously do also. After executing step (2), condition f is again valid. We thus return to step (3) with all of the conditions holding.

Termination of the algorithm is ensured by the fact that the non-negative integer deg(C) - d decreases between successive executions of step (3). When we find $d \le \lfloor \deg(C)/2 \rfloor$, we put C into F and terminate. By g, C has no factors of positive degree $\le \lfloor \deg(C)/2 \rfloor$, hence no proper factors at all. Thus C is prime and by d and e, the final value of F contains only prime polynomials and $C_0 = \pi F$. This concludes the proof.

d. Another application

In the algorithm of §3i, we have an application of Algorithm P with $D = Z$; $m = (m)$, where $m = p^j$, $E = Z_m$, and $R = \{i : i \in Z \text{ and } |i| < m/2\}$. In §4j, we have $D = Z[v_1,\ldots,v_n]$, $m = (m, (v_1 - a_1)^{j_1},\ldots,(v_n - a_n)^{j_n})$, and $R = \{A \in D : |A|_1 < m/2 \text{ and } \deg_{v_i}(A) < j_i, 1 \le i \le n\}$. As another example, take $D = F[v]$ where F is a field and m to be an ideal generated by an irreducible polynomial A[v]. A set R of representatives of $E = D/m$ is given by the set of all polynomials in F[v] of degree < n = deg(A). A polynomial $C \in F[v,x]$ will be R-factorable if all of its coefficients (as polynomials in F[v]) are of degree < n.

Thus, if we know how to factor over E, which is an extension field of F, we can use Algorithm P to factor polynomials in F[v,x] of degree < n in v. In particular, if $F = Z_p \simeq GF(p)$, then $E \simeq GF(p^n)$, the Galois field of order p^n. [BER71] describes a reasonably efficient algorithm for factorization over $GF(p^n)$. For factorization in $Z_p[v,x]$, the Hensel construction would thus be unnecessary, but the complexity of the algorithm of [BER71] probably makes the whole process more complex than if a Hensel construction were used.

Acknowledgement. The author gratefully acknowledges his indebtedness to Professor George E. Collins for his suggestions, encouragement and guidance throughout the course of this work.

REFERENCES

[BER68] E. R. Berlekamp, Algebraic Coding Theory, McGraw-Hill, Inc., New York, 1968.

[BER71] E. R. Berlekamp, Factoring Polynomials over Large Finite Fields, Mathematics of Computation, November, 1971.

[COL69] George E. Collins, L. E. Heindel, E. Horowitz, M. T. McClellan, and D. R. Musser, the SAC-1 Modular Arithmetic System, University of Wisconsin Technical Report No. 10, June, 1969.

[COL71] George E. Collins, the SAC-1 System: an Introduction and Survey, Preceedings of the Second Symposium on Symbolic and Algebraic Manipulation, Los Angeles, March, 1971.

[COL72] George E. Collins and D. R. Musser, the SAC-1 Polynomial Factorization System, Computer Sciences Technical Report No.

[COi72a] George E. Collins, unpublished lecture notes.

[FLO67] Robert W. Floyd, Assigning Meanings to Programs, Proceedings of a Symposium in Applied Mathematics, Vol, 19 - Mathematical Aspects of Computer Science (J. T. Schwartz, ed.). American Mathematical Society, Providence, R. I., 1967, pp. 19-32.

[GOL70] Jacob K. Goldhaber and Gertrude Ehrlich, Algebra, Macmillan Company, 1970.

[HOR69] Ellis Horowitz, Algorithms for Symbolic Integration of Rational Functions, Ph.D. Thesis, Computer Sciences Dept., University of Wisconsin, 1969.

[JOH66] S. C. Johnson, Tricks for Improving Kronecker's Method, Bell Laboratories Report, 1966.

[KNU68] Donald E. Knuth, The Art of Computer Programming, Vol I: Fundamental Algorithms, Addison-Wesley Publishing Co., Reading, Mass., 1968.

KNU69] Donald E. Knuth, The Art of Computer Programming, Vol. II. Seminumerical Algorithms, Addison-Wesley Publishing Co., Reading, Mass., 1969.

[MUS71] D. R. Musser, Algorithms for Polynomial Factorization, Computer Sciences Dept., Technical Report No. 134 (Ph.D. Thesis), September 1971.

[RAB60] Michael O. Rabin, Computable Algebra, General Theory and Theory of Computable Fields, Transactions of the American Mathematical Society 95 (1960), pp. 341-360.

[VDW49] B. L. Van der Waerden, Modern Algebra, Vol. 1, trans. by Fred Blum, Frederick Ungar Publishing Co., New York, 1949.

[ZAS69] Hans Zassenhaus, On Hensel Factorization, I, Journal of Number Theory 1, 291-311 (1969).