Sparse Hensel Lifting*

Erich Kaltofen

Rensselaer Polytechnic Institute
Department of Computer Science
Troy, New York 12180

Abstract

A new algorithm is introduced which computes the multivariate leading coefficients of polynomial factors from their univariate images. This algorithm is incorporated into a sparse Hensel lifting scheme and only requires the factorization of a single univariate image. The algorithm also provides the content of the input polynomial in the main variable as a by-product. We show how we can take advantage of this property when computing the GCD of multivariate polynomials by sparse Hensel lifting.

Table of Contents

Keywords: Sparse polynomials, Hensel lifting, polynomial factorization, polynomial greatest common divisors.

* Extended abstract in Proc. EUROCAL 1985, Springer Lec. Notes Comp. Sci., vol. 204, pp. 4-17 (1985).

1. Introduction

Various problems on multivariate polynomials such as exact division, factorization, and greatest common divisor computation, can be solved by lifting a univariate factorization by a Hensel algorithm of some sort. Four major problems can greatly impede the efficiency of the lifting process when applied to sparse polynomials. For sake of clarity we shall discuss those problems in the context of sparse factorization, $f(x_1, \ldots, x_v)$ being the polynomial to be factored. The first problem is referred to as *bad zeros problem* which is observed when the univariate polynomial $f(x_1, 0, \ldots, 0)$ either drops in degree compared to $f(x_1, \ldots, x_v)$ or has, unlike f, multiple roots. Both events make lifting virtually impossible. One is forced to consider in place of f the polynomial $f(x_1, x_2+a_2, \ldots, x_v+a_v)$, a_i non-zero elements from the coefficient domain. The disadvantage of this transformation is that the latter polynomial will have many more terms than f, in case f originally was sparse. It is Zippel's [19] contribution to show how sparseness can be preserved during the lifting of the translated polynomial. The second problem is referred to as *extraneous factors problem* which occurs when $f(x_1, a_2, \ldots, a_v)$ has more factors than $f(x_1, \ldots, x_v)$ does. Then the lifted factors tend to be very dense. For certain coefficient domains such as the rationals this is unlikely to happen, in practice, though theoretical justification has only been obtained recently, cf. von zur Gathen [6] and Kaltofen [8]. The third problem is what we shall call the *abundant factors problem*, which has been recognized only very recently [7].† The problem arises when the factorization of f to be lifted consists of many polynomials. Then the intermediate linear systems created by the standard Hensel lifting techniques, described e.g. in [15], [18], and [19], have exponentially many equations in polynomially many unknowns. Von zur Gathen's clever solution to this problem shows how polynomially many equations still "strong enough" to determine all unknowns can by selected in advance. The fourth problem is called the *leading coefficient problem* which results from an ambiguity in the lifting process. One can essentially impose any leading coefficient on the factors and lift, but only the correct leading coefficients will keep the intermediate results sparse. Until now, no complete algorithm seems to be described, except the heuristic by Wang [15] for rational coefficients. The main result of this paper fills this gap. Our algorithm is surprising in that it works with just the factorization of $f(x_1, a_2, \ldots, a_v)$ and does not require, unlike Wang's method, the factorization of the leading coefficient of f.

Before we discuss further properties of our leading coefficient determination procedures, we wish to point out that von zur Gathen's [7] algorithm by-passes the leading coefficient problem. However, von zur Gathen's resolution compares, in most cases, unfavorably to ours because of two reasons. First, his algorithm needs to lift a univariate polynomial whose degree is the total degree of f whereas our algorithm works with a univariate polynomial whose degree is the minimum of the degrees in the individual variables. Second, von zur Gathen substitutes a linear form in three variables for the minor variable that is lifted, thus possibly cubing the number of monomials, whereas we can use a standard scheme that substitutes a linear form in just one variable.

† Thanks go to Joachim von zur Gathen for bringing this problem to my full attention.

Our leading coefficient determination algorithm is a probabilistic algorithm in the sense that the correct leading coefficients are returned only if the original evaluations are lucky, that is they do not nullify a certain polynomial derived from f. The algorithm invokes the sparse lifting algorithm recursively. Therefore, we will describe the sparse lifting process as well, including additional conditions on the evaluations under which this procedure will return the correct results. We will also prove that we only need to select the evaluations from a moderately large set in order to ensure that all recursive calls of our algorithms succeed with high chance. It is not an easy task to prove the correctness of the sparse lifting algorithm even if we have the leading coefficients correctly. We will also supply an entirely different and simpler proof than that given in [7].

Our leading coefficient determination algorithm has another interesting property. The content of f in the main variable can be determined along with the leading coefficients. This feature allows another important application of our algorithms. We can use it for substantially improving Moses's and Yun's EZ-GCD algorithm. Instead of computing the GCD of the contents of the input polynomials, we can do away with one single recursive call to the GCD procedure.

Notation: We will use $w_{i \ldots j}$ as a short-hand for the vector $(w_i, w_{i+1}, \ldots, w_j)$, $i \le j$. Thus $F[x_{1 \ldots v}]$ is the domain of polynomials in $x_1, \ldots, x_v$ over F. The degree of $f \in F[x_{1 \ldots v}]$ in x_i, $1 \le i \le v$, is denoted by $\deg_{x_i}(f)$, the total degree of f by $\deg(f)$; $\mathrm{ldcf}_{x_1}(f) \in F[x_{2 \ldots v}]$ denotes the leading coefficient of f in x_1, $\mathrm{cont}_{x_1}(f) \in F[x_{2 \ldots v}]$ the content of f in x_1 (which makes sense only if F is a GCD domain). The *content decomposition* of f,

$$f = f^{(x_1)} \cdot\,\cdot\,\cdot\, f^{(x_v)},\ f^{(x_i)} \in F[x_{i \ldots n}],\ 1 \le i \le v,$$

requires

$$\mathrm{cont}_{x_i}(f^{(x_i)} \cdot\,\cdot\,\cdot\, f^{(x_v)}) = f^{(x_{i+1})} \cdot\,\cdot\,\cdot\, f^{(x_v)},\ 1 \le i \le v-1.$$

Notice that $f^{(x_1)}$ is the primitive part of f in x_1. If F is a field, the content decomposition is unique except for association, that is multiplication with elements in $F \setminus \{0\}$ itself. In general, we write $a \sim b$ if a and b from a unique factorization domain (UFD) are associates. By $\mathrm{mon}(f)$ we denote the number of non-zero monomials in $f \in F[x_{1 \ldots v}]$, that is

$$\mathrm{mon}(f) = \mathrm{card}(I) \quad \text{with} \quad f = \sum_{i_{1 \ldots v} \in I} c_{i_{1 \ldots v}}\, x_1^{i_1} \cdot\,\cdot\,\cdot\, x_v^{i_v},\ \ c_{i_{1 \ldots v}} \in F \setminus \{0\}.$$

By $\mathbf{Z}$ we denote the integers and by $\mathbf{Z}^+ = \{1, 2, 3, \cdot\,\cdot\,\cdot\}$.

2. The Newton-Hensel Lemma

In this section we present another version of the Hensel lemma. The factorization to be lifted in this lemma is allowed to have multiple factors. The special case of one multiple factor becomes Newton iteration for computing roots, which explains the choice for our title. In allowing multiplicities greater than one we have, however, weakened the Hensel lemma significantly. Existence of a lifted factorization is, in general, not guaranteed any longer, only uniqueness. It requires the deep Hilbert irreducibility theorem and its effective versions to realize that in the multivariate case we do not need that aspect of the Hensel lemma, which in the univariate case is such a crucial one.

Lemma 2.1: Let $k \in \mathbf{Z}^+$, $\phi, \gamma_1, \ldots, \gamma_r \in K[y, x] \setminus K[y]$, $\lambda_0, \lambda_1, \ldots, \lambda_r \in K[y]$, K a field, $\deg_y(\gamma_i) < k$, $e_1, \ldots, e_r$ not divisible by the characteristic of K. Assume that

i) $\lambda(y) = \mathrm{ldcf}_x(\phi) = \lambda_0 \lambda_1^{e_1} \cdots \lambda_r^{e_r}$, $\lambda(0) \neq 0$,

ii) $\lambda_0 \gamma_1^{e_1} \cdots \gamma_r^{e_r} \equiv \phi \bmod y^k$,

iii) $\mathrm{ldcf}_x(\gamma_i) \equiv \lambda_i \bmod y^k$, $1 \le i \le r$,

iv) $\mathrm{GCD}(\gamma_i(0, x), \gamma_j(0, x)) = 1$ for $1 \le i < j \le r$.

Furthermore assume that there exist $\bar{\gamma}_i \in K[y, x]$ with $\deg_y(\bar{\gamma}_i) < k + 1$ such that

v) $\lambda_0\, \bar{\gamma}_1^{e_1} \cdots \bar{\gamma}_r^{e_r} \equiv \phi \bmod y^{k+1}$,

vi) $\mathrm{ldcf}_x(\bar{\gamma}_i) \equiv \lambda_i \bmod y^{k+1}$, $1 \le i \le r$,

vii) $\bar{\gamma}_i \equiv \gamma_i \bmod y^k$, $1 \le i \le r$.

Then the $\bar{\gamma}_i$ are uniquely determined.

Proof: Let

$$\tilde{\gamma}_i = \gamma_i + \lambda_{i,k+1}\, y^{k+1}\, x^{d_i}, \quad 1 \le i \le r,$$

where $\lambda_{i,k+1} \in K$ is the coefficient of y^{k+1} in λ_i, $d_i = \deg_x(\gamma_i)$. By assumption vi) and vii) we can write

$$\bar{\gamma}_i = \tilde{\gamma}_i + \hat{\gamma}_i\, y^{k+1}, \quad \hat{\gamma}_i \in K[x], \quad \deg(\hat{\gamma}_i) < d_i. \tag{A}$$

It suffices to prove that the $\hat{\gamma}_i$ are uniquely determined. Plugging (A) into v) we get

$$\lambda_0 \prod_{i=1}^{v} \tilde{\gamma}_i^{e_i - 1}\, (\tilde{\gamma}_i + e_i\, \hat{\gamma}_i\, y^k) \equiv \phi \bmod y^{k+1}$$

or

$$\lambda_0\, \tilde{\gamma}_1^{e_1-1} \cdot\cdot\cdot\, \tilde{\gamma}_r^{e_r-1} \sum_{i=1}^{r} (e_i\, \hat{\gamma}_i\, y^k \prod_{\substack{j=1\\ j\neq i}}^{r} \tilde{\gamma}_j) \equiv \phi - \lambda_0 \prod_{i=1}^{r} \tilde{\gamma}_i^{e_i} \bmod y^{k+1}. \tag{B}$$

Since $\tilde{\gamma}_i \equiv \gamma_i \bmod y^k$, it follows from ii) that

$$\phi - \lambda_0 \prod_{i=1}^{r} \tilde{\gamma}_i^{e_i} \equiv \tau(x)\, y^k \bmod y^{k+1},\ \tau(x) \in K[x].$$

Furthermore, since by iii) and (A) $\mathrm{ldcf}_x(\tilde{\gamma}_i) \equiv \lambda_i \bmod y^{k+1}$ it follows that $\deg(\tau) < \deg_x(\phi)$. Let

$$\eta_i(x) = \gamma_i(0, x),\ \ \eta_i^* = \prod_{\substack{j=1\\ j\neq i}}^{r} \eta_j, \quad 1 \le i \le r.$$

Notice that by iv) $\mathrm{GCD}(\eta_i, \eta_j) = 1$, $1 \le i < j \le r$. Now (B) is satisfied only if

$$\lambda_0(0)\, \eta_1^{e_1-1} \cdot\cdot\cdot\, \eta_r^{e_r-1} \sum_{i=1}^{v} e_i\, \eta_i^*\, \hat{\gamma}_i = \tau.$$

Thus $\eta_1^{e_1-1} \cdot\cdot\cdot\, \eta_r^{e_r-1}$ must divide τ, and let the quotient be $\rho \in K[x]$. By i) $\sum_{i=1}^{r} e_i \deg(\eta_i) = \deg_x(\phi)$, therefore $\deg(\rho) < \deg(\eta_1) + \cdot\cdot\cdot + \deg(\eta_r)$ and we have

$$\frac{\rho}{\eta_1 \cdot\cdot\cdot\, \eta_r} = \frac{\lambda_0(0) e_1 \hat{\gamma}_1}{\eta_1} + \cdot\cdot\cdot + \frac{\lambda_0(0) e_r \hat{\gamma}_r}{\eta_r}\,.$$

From this we conclude that $\hat{\gamma}_i$ is the numerator of the ith term in the partial fraction decomposition of $\rho/(\eta_1 \cdot\cdot\cdot\, \eta_r)$ divided by $\lambda_0(0)\, e_i \neq 0$, and therefore it is uniquely determined. □

3. GCD-Free Basis Construction

In the Newton-Hensel lemma we needed the assumption of pairwise relative primality on the factors we lift. If one wants to lift a factorization $g_1 \cdots g_r \equiv f \bmod y$, where $\mathrm{GCD}(g_i, g_j) \not\sim 1$, one has to refine the factorization by repeated GCD computation. We introduce the notion of GCD-free basis for this problem. We shall, however, formulate our definitions and lemmas over general unique factorization domains, though we have polynomial domains in mind.

Definition 3.1: Let $A = \{a_1, \ldots, a_r\} \subset D \setminus \{0\}$, D a UFD. A set $B = \{b_1, \ldots, b_s\}$ is called a *GCD-free basis* for A if

i) For all $1 \le i < j \le s$: $\mathrm{GCD}(b_i, b_j) \sim 1$.

ii) For all $1 \le i \le r$, $1 \le j \le s$ there exist $e_{ij} \in \mathbf{Z}$, $e_{ij} \ge 0$: $a_i \sim \prod_{j=1}^{s} b_j^{e_{ij}}$.

Lemma 3.1: Let A, B as above, $g = \mathrm{GCD}(a_1, a_2)$. Then

$$g = \prod_{j=1}^{s} b_j^{\min(e_{1j}, e_{2j})}.$$

Proof: By considering the factorization of b_j into primes. □

The question arises whether GCD-free bases with a minimum number of elements are unique. Of course, we exclude distinctions introduced by multiplying with units. Since $\{b_1, b_2, \cdots\}$ is a basis provided $\{b_1^2, b_2, \cdots\}$ was one, we necessarily need an additional condition.

Definition 3.2: A GCD-free basis B for $A \subset D \setminus \{0\}$ is called *standard* if

i) $s = \mathrm{card}(B) = \min\{\mathrm{card}(C) \mid C \text{ is a GCD-free basis for } A\}$,

ii) For all $1 \le j \le s$, $m > 1$: $\{b_1, \ldots, b_{j-1}, b_j^m, b_{j+1}, \ldots, b_s\}$ is not a GCD-free basis for A.

Lemma 3.2: A standard GCD-free basis B for $A \subset D \setminus \{0\}$ is uniquely determined (except for association).

Proof: By induction on the number of prime factors in $\prod_{i=1}^{v} a_i$.

Case 1: For all $i \ne j$, $\mathrm{GCD}(a_i, a_j) \sim 1$. It is easy to see that $A \setminus U(D)$, $U(D)$ the units of D, must be the standard GCD-free basis for A.

Case 2: There exist $i \ne j$, such that $\mathrm{GCD}(a_i, a_j) \sim g \not\sim 1$. Consider

$$A' = \{a_1, \ldots, \frac{a_i}{g}, \ldots, \frac{a_j}{g}, \ldots, a_r, g\}.$$

By lemma 3.1, B is a GCD-free basis for A if and only if B is a GCD-free basis for A'. This condition remains true for standard bases. Since $(\prod_{i=1}^{v} a_i)/g$, the product of all elements in A', has fewer prime factors than $\prod_{i=1}^{v} a_i$, the induction hypothesis implies the uniqueness. □

The above proof also supplies the following algorithm.

Standard GCD-free basis algorithm

Input: $A = \{a_1, \ldots, a_r\} \subset D \setminus (\{0\} \cup U(D))$, D a UFD, $U(D)$ its units.

Output: $B = \{b_1, \ldots, b_s\}$ such that B is a standard GCD-free basis for A.

```
$t \leftarrow r$; $c_i \leftarrow a_i$ for $i = 1, \ldots, r$; $I = \{1, \ldots, r\}$.
FOR $i \leftarrow 1, 2, \cdots$ WHILE $(i < t)$ DO
    FOR $j \leftarrow i+1, i+2, \ldots,$ WHILE $(j \le t)$ DO
        IF $i \in I$ and $j \in I$ THEN
            $c_{t+1} \leftarrow \mathrm{GCD}(c_i, c_j)$.
            IF $c_{t+1} \nsim 1$ THEN   $c_i \leftarrow c_i / c_{t+1}$; $c_j \leftarrow c_j / c_{t+1}$.
                                     IF $c_i \sim 1$ THEN $I \leftarrow I \setminus \{i\}$.
                                     IF $c_j \sim 1$ THEN $I \leftarrow I \setminus \{j\}$.
                                     $I \leftarrow I \cup \{t+1\}$; $t \leftarrow t+1$.
RETURN $\{b_i \leftarrow c_i\}_{i \in I}$.   □
```

We will compute standard GCD-free bases of univariate polynomials obtained from multivariate polynomials by evaluation. The question arises under which condition these bases are just the evaluated multivariate bases. The following lemma answers this question in the general setting.

Lemma 3.3: Let $A = \{a_1, \ldots, a_r\} \subset D \setminus (\{0\} \cup U(D))$, D a UFD, $B = \{b_1, \ldots, b_s\}$ a standard GCD-free basis for A. Let E be a UFD, $\phi: D \to E$ a ring homomorphism. Assume that

i) For all $1 \le i \le s$: $\phi b_i \ne 0$, $\phi b_i \notin U(E)$, the units in E.

ii) For all $1 \le i < j \le s$: $\mathrm{GCD}(\phi b_i, \phi b_j) \sim 1$.

Then $\phi B = \{\phi b_1, \ldots, \phi b_s\}$ is a standard GCD-free basis for $\phi A = \{\phi a_1, \ldots, \phi a_r\}$.

Proof: We show that the following is a loop invariant for our algorithm.

$$\phi(\mathrm{GCD}(c_i, c_j)) \sim \mathrm{GCD}(\phi c_i, \phi c_j) \quad \text{for } i, j \in I. \tag{A}$$

Since $\{b_1, \ldots, b_s\}$ is a GCD-free basis for $\{c_i\}_{i \in I}$ there exist integers $e_{ik}, e_{jk} \ge 0$ such that

$$c_i = \prod_{k=1}^{s} b_k^{e_{ik}}, \quad c_j = \prod_{k=1}^{s} b_k^{e_{jk}}.$$

Therefore

$$\phi c_i = \prod_{k=1}^{s} (\phi b_k)^{e_{ik}}, \quad \phi c_j = \prod_{k=1}^{s} (\phi b_k)^{e_{jk}}.$$

By lemma 3.1 and the assumption we must have

$$\mathrm{GCD}(c_{i,}\, c_j) = \prod_{k=1}^{s} b_k^{\min(e_{ik},\, e_{jk})}, \quad \mathrm{GCD}(\phi c_{i,}\, \phi c_j) = \prod_{k=1}^{s} (\phi b_k)^{\min(e_{ik},\, e_{jk})},$$

which proves (A). But (A) implies that our algorithm when applied to ϕA will compute ϕB which is thus a standard GCD-free basis for ϕA. $\square$

Historical Note: GCD-free bases have been used by Epstein [5] to determine pseudo-multiplicative independence of sets of polynomials. Epstein insists, however, that the b_i are squarefree, which in retrospect is an unnecessary condition. Also Epstein does not establish uniqueness of the bases. Furthermore, lemma 3.3 gives a probabilistic algorithm to determine the multiplicative relationship among polynomials, using evaluation for ϕ. We shall not go into the details. Finally, the standard GCD-free basis algorithm can be viewed as another instance of the so-called *critical-pair completion* algorithms (cf. Buchberger and Loos [3]). We have not used this notion because our results turn out to be obtainable quite straight-forwardly.

4. The Sparse Lifting Algorithm

We now present our version of the sparse Hensel lifting procedure. The algorithm combines essentially three ideas. The first is Zippel's [19] on the preservation of sparseness by assuming that any zero coefficient in the randomly evaluated polynomial indicates a zero polynomial. The second idea is von zur Gathen's [7] on lifting factors with high multiplicities and on restricting the number of equations in the arising linear systems. The third idea is ours on computing the leading coefficients and contents by recursive application of the sparse lifting algorithm. We also prove that under certain assumptions the algorithm computes the correct result. Our proof, based on lemma 2.1, not only seems simpler than von zur Gathen's [7] but also leads to a factorization procedure for polynomials given by straight-line programs [11].

Sparse Hensel Lifting Algorithm

Input: $f \in F[x_{1\ldots v}]$, F a field, $a_2, \ldots, a_v \in F$, and $g_1, \ldots, g_r \in F[x_1] \setminus F$, $e_1, \ldots, e_r \in \mathbf{Z}^+$ such that

$$\mathrm{ldcf}_{x_1}(f)(a_{2\ldots v}) \neq 0$$

and

$$g_1^{e_1} \cdots g_r^{e_r} = f(x_1, a_{2\ldots v}),\ \ \mathrm{GCD}(g_i, g_j) = 1,\ 1 \le i < j \le r.$$

We make the following assumptions on $a_2, \ldots, a_v$:

True Factors Assumptions: There exist polynomials $h_0, \ldots, h_r \in F[x_{1\ldots v}]$ such that

$$h_0 \sim \mathrm{cont}_{x_1}(f),\ \ h_0 h_1^{e_1} \cdots h_r^{e_r} = f,\ \ h_i(x_1, a_{2\ldots v}) = g_i,\ 1 \le i \le r. \qquad (\dagger)$$

Notice that the h_i are uniquely determined.

Zero Preservation Assumption: Let $q_i^{(j_{1\ldots n})} \in F[x_{n+1\ldots v}]$ be the coefficient of $x_1^{j_1} \cdots x_n^{j_n}$ in h_i, $1 \le i \le r$, $1 \le n < v$. Then for all i, n, and $j_{1\ldots n} \in (\mathbf{Z}^+ \cup \{0\})^n$

$$q_i^{(j_{1\ldots n})} \neq 0 \ \text{ implies } \ q_i^{(j_{1\ldots n})}(a_{n+1\ldots v}) \neq 0.$$

Third, we assume that the call to the Leading Coefficient algorithm in step 1 produces the correct result. This will add more conditions for the $a_2, \ldots, a_v$.

Output: $h_0, \ldots, h_r \in F[x_{1\ldots v}]$ satisfying (†).

Step 1: Call the Leading Coefficient Determination Algorithm of §5 with the above inputs. We will get back h_0, $l_i = \mathrm{ldcf}_{x_1}(h_i)$ and $g_{i2} = h_i(x_1, x_2, a_{3\ldots v})$, $1 \le i \le r$. If $r = e_1 = 1$ then return h_0, $h_1 \leftarrow f/h_0$.

Step 2: Special handling for positive characteristic.
If $\mathrm{char}(F) = p > 0$ then do for all i with $p \mid e_i$:

Compute μ_i, $\nu_i \in \mathbf{Z}^+$ such that $e_i = p^{\nu_i}\mu_i$, GCD$(p, \mu_i) = 1$.
Set $g_{i2} \leftarrow g_{i2}^{p^{\nu_i}}$, $e_i \leftarrow \mu_i$.

Step 3: FOR $n \leftarrow 3 ,\ldots, v$ DO Step 4.

Step 4: Lift a single variable. At his moment we have polynomials $g_{i,n-1} \in F[x_{1\ldots n}]$ with $g_{i,n-1} = h_i(x_{1\ldots n-1}, a_{n\ldots v})$.

Let $g_{i,n}^{(1)} = g_{i,n-1}$ for $i = 1 ,\ldots, r$.

FOR $k \leftarrow 1, 2, \cdots$ WHILE $(\deg_{x_n}(f) > \deg_{x_n}(h_0) + \sum_{i=1}^{r} e_i \deg_y(g_{i,n}^{(k)}))$ DO Step 5.

Step 5: Lift $y = x_n - a_n$ by one degree. At this moment we have polynomials $g_{i,n}^{(k)} \in F[x_{1\ldots n-1}, y]$ such that

$$g_{i,n}^{(k)} = h_i(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v}) \bmod y^k. \tag{‡}$$

5.1: For $1 \le i \le r$ set $\hat{g}_i = \sum c_i^{(j_{1\ldots n-1})}\, x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ where the $c_i^{(j_{1\ldots n-1})}$ are unknown coefficients and the summation is taken over all vectors $j_{1\ldots n-1}$ such that the coefficient of the monomial $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ in $g_{i,n-1}$ is not zero.

5.2: For each exponent vector $j_{1\ldots n-1}$ such that $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}\, y^k$ occurs as a monomial in $l_i(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v})$ with non-zero coefficient, replace $c_i^{(j_{1\ldots n-1})}$ by this non-zero field element.

5.3: In the following we use the set $E_{n-1}^{(k)}$ of $n-1$-dimensional exponent-vectors whose initialization we shall discuss in a moment. For all $m_{1\ldots n-1} \in E_{n-1}^{(k)}$ collect the coefficients of the monomials $x_1^{m_1} \cdots x_{n-1}^{m_{n-1}}\, y^k$ in

$$f(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v}) - h_0(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v}) \prod_{i=1}^{r} \left(g_{i,n}^{(k)} + \hat{g}_i\, y^k \right)^{e_i}. \tag{*}$$

We will prove in theorem 4.1 that these coefficients are linear forms in the $c_i^{(j_{1\ldots n-1})}$ which have a unique solution in F. Now we concern ourselves with the initialization of $E_2^{(1)}$. In that case we simply collect all non-zero monomials occurring in the expansion of (*), whose exponent-vectors constitute $E_2^{(1)}$. Notice that the expansion of (*) can contain exponentially many non-zero monomials. The set $E_{n-1}^{(k)}$ will restrict us to polynomially many equations which are still “strong enough” (in the terminology of [7]) to determine all unknowns.

5.4: Compute the solution for the arising linear system, evaluate the $\hat{g}_i$ at that solution and set $g_{i,n}^{(k+1)} \leftarrow g_{i,n}^{(k)} + \hat{g}_i y^k$.

5.5: If $k = 1$ we compute $E_{n-1}^{(2)} = E_{n-1}^{(3)} = \cdots$ as follows. Determine in the linear system solved in step 5.4 a maximal set of linearly independent homogeneous equations. Assign to $E_{n-1}^{(\kappa)}$, $\kappa \ge 2$, the set of the exponent-vectors $m_{1\ldots n-1}$ of the monomials $x_1^{m_1} \cdots x_{n-1}^{m_{n-1}}\, y^k$ corresponding to those equations.

Step 4 cont.: Back translate: $g_{i,n} \leftarrow g_{i,n}^{(k)}(x_{1\ldots n-1}, x_n - a_n)$.
Set $E_n^{(1)} = \{(m_{1\ldots n-1}, d) \mid m_{1\ldots n-1} \in E_{n-1}^{(k)}, 0 \le d \le \deg_{x_n}(f)\}$.

Step 2 cont.: For each i with $e_i = \mu_i$ replace $g_{i,v} = \sum b_i^{(j_{1\ldots v})} x_1^{j_1} \cdot\cdot\cdot x_v^{j_v}$, $b_i^{(j_{1\ldots v})} \in F$, by $\sum (b_i^{(j_{1\ldots v})} x_1^{j_1} \cdot\cdot\cdot x_v^{j_v})^{1/p^{v_i}}$. It follows from our assumptions that $p^{v_i} \mid j_n$, $1 \le n \le v$, and that $(b_i^{(j_{1\ldots v})})^{1/p^{v_i}} \in F$. Here we must also assume that we can effectively compute p-th roots in F.

Step 6: Return h_0, $h_i \leftarrow g_{i,v}$ for $1 \le i \le r$. □

Remark 1: It might be unclear why we do not factor out the content h_0 of f before lifting or why we do not compute the squarefree factors of g_i and lift them with respect to the squarefree part of f. The answer is that then some intermediately needed polynomials such as the primitive or squarefree part of f or a squarefree factor of g_i may become dense, although the final h_i still turn out to be sparse. Following are some examples in which these intermediate factors have super-polynomially more non-zero monomials.

Example: Let

$$f_1(x_1,\ldots,x_v) = \prod_{i=2}^{v} (x_i - 1) \prod_{i=2}^{v} (x_1 - \sum_{j=0}^{d} x_i^j).$$

Then $\mathrm{mon}(f_1) \le 4^{v-1}$ whereas $\mathrm{mon}(f_1^{(x_1)}) \ge (d+1)^{v-1}$. Let

$$f_2(x_1,\ldots,x_v) = \prod_{i=2}^{v} (\sum_{j=0}^{d} x_1^j\, x_i^{d-j}) \prod_{i=2}^{v} (x_1 - x_i)^2.$$

Then $\mathrm{mon}(f_2) \le 4^{v-1}$ whereas the squarefree factor with multiplicity 1 has $(d+1)^{v-1}$ monomials. Finally, let

$$f_3(x_1,\ldots,x_v) = \prod_{i=2}^{v} ((\sum_{j=0}^{d} x_1^j\, x_i^{d-j})^2 + x_1) \prod_{i=2}^{v} (x_1 - x_i)^2.$$

Then $\mathrm{mon}(f_3) \le 5^{v-1}$ whereas the squarefree part of f_3 has at least $(d+1)^{v-1}$ monomials.

One could argue, however, that the above cases are pathological and normally factors tend to have fewer monomials. Under that hypothesis it is, of course, more efficient to lift the squarefree factors of the g_i. Finally, we wish to remark that we conjecture that powers of dense polynomials must be dense.

Remark 2: We described our algorithm for coefficients from a field. If we wish to lift over a unique factorization domain, such as the integers, it is possible to avoid working in the quotient field, e.g., the rationals. First we note the following version of Gauss's lemma, a generalization of which to algebraic number fields is proven in [9].

Lemma 4.1: We call a polynomial $f = \sum_{i_{1\ldots v} \in I} c_{i_{1\ldots v}}\, x_1^{i_1} \cdot\cdot\cdot x_v^{i_v} \in D[x_{1\ldots v}]$, D a UFD, coefficient primitive iff its *coefficient content* $\mathrm{GCD}_{i_{1\ldots v} \in I}(c_{i_{1\ldots v}}) \sim 1$. Then the product of two coefficient primitive polynomials is coefficient primitive. □

Thus, we can first make f coefficient primitive and then compute coefficient primitive h_0 $,\ldots,$ h_i such that $h_i(x_i, a_{2\ldots v}) \sim g_i$, $1 \le i \le r$. This means, of course, that the leading coefficient algorithm can only determine the coefficient primitive parts of $\mathrm{ldcf}_{x_1}(h_i)$. Using these parts the solution to (*) can become a vector of quotients. However, Bareiss's [1] exact division algorithm for solving linear systems can be modified, using the coefficient content of $\mathrm{ldcf}_{x_1}(f)$, to compute the h_i by ring operations in D and exact divisions. We hope the reader will not find it too difficult to work out the details.

Remark 3: A special case of our algorithm occurs when $r = 1$. Then the algorithm computes the sparse e_1-st root of a sparse polynomial polynomial f. Closer inspection reveals that then the algorithm performs a sparse version of Newton iteration, which is normally explicitly coded for various other problems.

Remark 4: Though it is quite unlikely that our assumptions are violated provided the a_i are randomly selected from a sufficiently large sample set, as we will establish in §6, we want to note how such an exceptional situation can be detected. It is clear that one always can check the final answer by multiplying the results together. Von zur Gathen [7] even suggests to randomly evaluate the results to get great confidence in the correctness of the answer, but to avoid the sometimes costly multiplication. It is also clear that if the linear system computed in Step 5.3 turns out to be singular, an assumption, most probably the zero preservation assumption, must be violated. If we use the exact division version (see remark 2) and an arising division is not exact, most probably we do not lift true factors. Finally, if the intermediate $g_{l,n}$ are not sparse, as we might suspect they should be, we either have the wrong leading coefficients or false factor images.

Now we prove the correctness of the sparse lifting algorithm.

Theorem 4.1 (*Main Theorem on Sparse Lifting*): In the Sparse Hensel Lifting algorithm the coefficients of all monomials $x_1^{m_1} \cdots x_{n-1}^{m_{n-1}}\, y^k$ in (*) are linear forms in the unknowns $c_i^{(j_{1\ldots n-1})}$. Furthermore, under the true factors and zero preservation assumptions and the provision that Step 2 computes the correct leading coefficients, the derived linear system has a unique solution.

Proof: The linearity of the forms is easy to see. Any monomial containing a product of the $c_i^{(j_{1\ldots n-1})}$ has degree at least $2k$ in y. From this we also conclude that the homogeneous parts of these forms do not change as k increases in step 4, thus justifying step 5.5. The arising system obviously has a solution for $c_i^{(j_{1\ldots n-1})}$, namely the coefficients $b_i^{(j_{1\ldots n-1})}$ of $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}\, y^k$ in $h_i(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v})$. The hard part is to prove that the solution is unique. We prove this by showing that (‡) is a loop invariant. First we prove the theorem for the algorithm using the following modification in step 5.

Step 5': Version of step 5 for the sake of the proof.

5.1': For $1 \le i \le r$ set $\hat{g}_i = \sum c_i^{(j_{1\ldots n-1})} x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ where the $c_i^{(j_{1\ldots n-1})}$ are unknown coefficients and the summation is taken over all vectors $j_{1\ldots n-1}$ such that $0 \le j_t \le d_t = \deg_{x_t}(g_{i,n-1})$, $1 \le t \le n-1$.

5.2': For each exponent vector $(d_1, j_{2\ldots n-1})$ replace $c_i^{(d_1, j_{2\ldots n-1})}$ by the coefficient of the corresponding monomial in $l_i(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v})$, be it zero or not.

5.3': Collect the coefficients of *all* monomials $x_1^{m_1} \cdots x_{n-1}^{m_{n-1}} y^k$ in

$$f(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v}) - h_0(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v}) \prod_{i=1}^{r} \left(g_{i,n}^{(k)} + \hat{g}_i \, y^k \right)^{e_i}. \tag{*}$$

5.4': Compute the solution for the arising linear system, evaluate the $\hat{g}_i$ at that solution and set $g_{i,n}^{(k+1)} \leftarrow g_{i,n}^{(k)} + \hat{g}_i y^k$.

5.5': This step is empty in the modified version.

We now can apply lemma 2.1 to Step 5' with $K = F(x_{2\ldots n-1})$, $x = x_1$, $\phi = f(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v})$, $\gamma_i = g_{i,n}^{(k)}$, $\lambda_0 = h_0(x_{2\ldots n-1}, y + a_n, a_{n+1\ldots v})$, $\lambda_i = l_i(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v})$, $\bar{\gamma}_i = g_{i,n}^{(k+1)} = g_{i,n}^{(k)} + \hat{g}_i \, y^k$. The assumptions to lemma 2.1 are satisfied as follows: i) because we have the correct content and leading coefficients, ii) and iii) because of the hypothesis, that is (‡) for k; v) follows from the fact that we constructed $g_{i,n}^{(k+1)}$ from a solution of (*); vi) and vii) follow from the way we constructed $\hat{g}_i$ in step 5.2' and $g_{i,n}^{(k+1)}$ in step 5.4'. Finally iv) is checked easily: The $\gamma_i(0, x) = h_i(x_{1\ldots n-1}, a_{n\ldots v})$ are pairwise relatively prime in $F(x_{2\ldots n-1})[x_1]$ because the $h_i(x_1, a_{2\ldots v}) = g_i(x_1)$ are. The conclusion is that $g_{i,n}^{(k+1)}$ are uniquely determined.

We now consider the restrictions of step 5 itself. The linear system constructed in step 5.3 has both fewer variables and fewer equations than that of step 5.3'. We first show that all $c_i^{(j_{1\ldots n-1})}$ considered in step 5.1' but not in step 5.1 are solved as 0 in step 5'. This means that we have deprived the linear system in step 5 of variables which become 0 in the solution. It is clear that this restriction by itself cannot turn the system into a singular one. We shall proceed by assuming that the solution for some $c_i^{(j_{1\ldots n-1})}$ in step 5.4' is non-zero and $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ does not occur $g_{i,n-1}$. Continuing the loop on step 5' with $k+1, k+2, \cdots$ we already know that for some k_0 we will get $g_{i,n}^{(k_0)} = h_i(x_{1\ldots n-1}, y + a_n, a_{n+1\ldots v})$. Let $q_{i,n}^{(j_{1\ldots n-1})}(y) \in F[y]$ be the coefficient of $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ in $g_{i,n}^{(k_0)}$. Since $c_i^{(j_{1\ldots n-1})} \ne 0$, $q_i^{(j_{1\ldots n-1})}(y) \ne 0$. Thus the coefficient of $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ in $g_{i,n} = g_{i,n}^{(k_0)}(x_{1\ldots n-1}, x_n - a_n, a_{n+1\ldots v})$ is $q_{i,n}^{(j_{1\ldots n-1})}(x_n - a_n) \ne 0$, hence the coefficient $g_i^{(j_{1\ldots n-1})}(x_{n\ldots v})$ of $x_1^{j_1} \cdots x_{n-1}^{j_{n-1}}$ in h_i is non-zero, in violation to the zero presentation assumption.

We finally prove that restricting the linear system of step 5.3' to equations corresponding to monomials derived via their exponent vectors in $E_{n-1}^{(k)}$ does not yield multiple solutions. By step 5.5 this is clearly true for $E_{n-1}^{(k)}$, $k \ge 2$, provided it was true for $E_{n-1}^{(1)}$. The latter follows by considering the next execution of step 5 for

$$f(x_{1\ldots n-1}, x_n, y + a_{n+1}, a_{n+2\ldots v}) \in F(x_n)[x_{1\ldots n-1}, y].$$

Instead of working over the field F we now work over $F(x_n)$. The arising linear system can still be solved uniquely when restricted to $E_{n-1}^{(k)}$. But we know that the entries in the unique solution are, in fact, in $F[x_n]$. Therefore we can set these entries up as polynomials of maximum degree $\deg_{x_n}(f)$ and still must get a unique solution. This is, of course, equivalent to restricting ourselves to $E_n^{(1)}$ as determined in the continuation of step 4. $\square$

5. The Leading Coefficient Algorithm

It is crucial for the sparse lifting algorithm to work properly that the correct leading coefficients are know beforehand. In this section we show how they can be computed using the sparse lifting algorithm recursively on polynomials in one less variable. The main idea is quite simple. We determine a bivariate factorization of $f(x_1, x_2, a_{3\ldots v})$ and lift the leading coefficients of that factorization with respect to the leading coefficient of f. Two problems arise. First, the leading coefficients of the factors may not contain x_2 at all. Thus we must try all factorizations of $f(x_1, a_{2\ldots n-1}, x_n, a_{n+1\ldots v})$. Second, the leading coefficients of the bivariate factors may have common GCDs. Thus we must compute a standard GCD-free basis for them. It is more or less incidental that the whole mechanism also produces the content of f.

Leading Coefficient and Content Determination Algorithm

Input: As in algorithm 1.

In addition to the true factors assumption we make the following assumptions on $a_2, \ldots, a_v$.

True leading terms assumption: Let $\lambda_0 = h_0$, $\lambda_i = \mathrm{ldcf}_{x_1}(h_i)$, $1 \le i \le r$, where $h_0, \ldots, h_r$ are the outputs of the Sparse Hensel Lifting algorithm. Let

$$\lambda_i^{(x_2)} \cdot \cdot \cdot \lambda_i^{(x_v)} = \lambda_i, \quad 0 \le i \le r,$$

be the content decomposition of λ_i and let

$$\{\beta_{n1}, \ldots, \beta_{n,s_n}\}, \quad 2 \le n \le v,$$

be a standard GCD-free basis for $\{\lambda_0^{(x_n)}, \ldots, \lambda_r^{(x_n)}\}$. We assume that for all $2 \le n \le v$

i) $\deg_{x_n}(\lambda_i) = \deg_{x_n}(\lambda_i(a_{2\ldots n-1}, x_n, a_{n+1\ldots v}))$ for all $0 \le i \le r$,

ii) $\lambda_0(a_{2\ldots n-1}, x_n, a_{n+1\ldots v}) \sim \mathrm{cont}_{x_1}(f(x_1, a_{2\ldots n-1}, x_n, a_{n+1\ldots v}))$,

iii) $(\mathrm{GCD}(\beta_{ni}, \beta_{nj}))(x_n, a_{n+1\ldots v}) \sim \mathrm{GCD}(\beta_{ni}(x_n, a_{n+1\ldots v}), \beta_{nj}(x_n, a_{n+1\ldots v}))$ for all $1 \le i < j \le s_n$.

Second, we assume that the calls to Sparse Lifting in step 3 return the correct results. Notice that the true factors assumption is guaranteed already by previous assumptions.

Output: $h_0, l_1, \ldots, l_r \in F[x_2, \ldots, x_n]$ such that $h_0 \sim \mathrm{cont}_{x_1}(f)$ and $l_i = \mathrm{ldcf}_{x_1}(h_i)$, $1 \le i \le r$. Furthermore, $g_{12}, \ldots, g_{r2} \in F[x_1, x_2]$ such that $g_{i2} = h_i(x_1, x_2, a_{3\ldots v})$, $1 \le i \le r$.

Step 1: This step drives the iteration.

Initialization: $l \leftarrow \mathrm{ldcf}_{x_1}(f)$; $l_i \leftarrow 1$ for $i = 0, \ldots, r$.

FOR $n \leftarrow 2, \ldots, v$ WHILE $l \notin F$ DO

```
    IF $\deg_{x_n}(l) > 0$ THEN   Perform step 2.
                                  IF $I \neq \emptyset$ THEN perform step 3.
```

Step 2: At this point the following is always true.

$$l_i \sim \prod_{j=2}^{n-1} \lambda_i^{(x_j)}, \quad 0 \le i \le r, \quad l \sim \frac{\mathrm{ldcf}_{x_1}(f)}{l_0 l_1 \cdot \cdot \cdot l_r}. \tag{†}$$

Lift $\tilde f(x_1, x_n) = f(x_1, a_{2\ldots n-1}, x_n, a_{n+1\ldots v})$ into $\tilde g_0, \ldots, \tilde g_r \in F[x_1, x_n]$ such that

$$\tilde g_0 = \mathrm{cont}_{x_1}(\tilde f),\ \tilde g_i(x_1, a_n) = g_i,\ 1 \le i \le r,\ \tilde g_0\, \tilde g_1^{e_1} \cdot \cdot \cdot \tilde g_r^{e_r} = \tilde f.$$

This amounts to a standard lifting step (see remark 2 below for improvements).
IF $n = 2$ THEN set $g_{i2} \leftarrow \tilde g_i$ for $i = 1, \ldots, r$.
Compute the images of $\lambda_i^{(x_n)}$:
FOR $i \leftarrow 0, \ldots, r$ DO

```
    IF $\deg_{x_n}(l_i) > 0$   THEN $\tilde l_i \leftarrow \mathrm{ldcf}_{x_1}(\tilde g_i)/l_i(a_{2\ldots n-1}, x_n, a_{n+1\ldots v})$
                               ELSE $\tilde l_i \leftarrow \mathrm{ldcf}_{x_1}(\tilde g_i)$.
```

At this point

$$\tilde l_i \sim \lambda_i^{(x_n)}(x_n, a_{n+1\ldots v}), \quad \deg_{x_n}(\tilde l_i) = \deg_{x_n}(\lambda_i^{(x_n)}), \quad 0 \le i \le r. \tag{‡}$$

Set $I = \{i_1, \ldots, i_m\} \subset \{0, \ldots, r\}$ such that $i \in I$ iff $\tilde l_i \notin F$.

Step 3: Find a standard GCD-free basis $\{\tilde b_1, \ldots, \tilde b_s\}$ for $\{\tilde l_i\}_{i \in I}$. Let $e_0 = 1$ and $\tilde l_i = \prod_{j=1}^{s} \tilde b_j^{u_{ij}}$, $0 \le i \le r$.

Call Sparse Lifting recursively with

$$cl = \prod_{j=1}^{s} \tilde b_j^{(\Sigma_{i \in I} e_i u_{ij})} \bmod (x_{n+1} - a_{n+1}, \ldots, x_v - a_v).$$

where $c \in F$ is properly determined. The polynomials $b_0, b_1, \ldots, b_s$ are returned.

Set $l \leftarrow b_0$; $l_i \leftarrow l_i \times \prod_{j=1}^{s} b_j^{u_{ij}}$ for all $i \in I$.

Step 1 cont.: Adjust outputs:
$l_0 \leftarrow l_0/l_0(a_{2\ldots v})$. FOR $i \leftarrow 1, \ldots, r$ DO $l_i \leftarrow l_i\ \mathrm{ldcf}(g_i)/l_i(a_{2\ldots v})$.
Return $h_0 \leftarrow l_0, l_1, \ldots, l_r, g_{l2}, \ldots, g_{r2}$. □

Remark 1: In step 2 we must compute the factorization of $\tilde f(x_1, x_n)$. This amounts to lifting

$$g_1^{e_1} \cdot \cdot \cdot g_r^{e_r} \equiv \tilde f(x_1, x_n) \bmod (x_n - a_n)$$

similarly to step 4 of the Sparse Hensel Lifting algorithm. However, the content $\tilde g_0$ and the leading coefficients of $\tilde g_i$ are unknown and we must use special tricks. The following approach, that adapts Wang's [17] early factor detection algorithm in substep 3, appears to be the most efficient.

Substep 1: Compute $\tilde{g}_0 = \mathrm{cont}_{x_1}(\tilde{f})$, $\tilde{f}^{(x_1)}(x_1, x_n)$ and the squarefree factorizations

$$\frac{g_i}{\mathrm{ldcf}(g_i)} = g_{i1}\, g_{i2}^2 \cdot \cdot \cdot g_{i,t_i}^{t_i},\;\; g_{ij} \text{ monic },\, 1 \le i \le r,$$

by univariate GCD computations. Furthermore, compute the squarefree part $\bar{f}(x_1, x_n)$ of $\tilde{f}^{(x_1)}(x_1, x_n)$ by a bivariate GCD computation. Here we must assume that we can effectively determine p-th roots in case F has characteristic $p > 0$.

Substep 2: Lift

$$\mathrm{ldcf}_{x_1}(\tilde{f}(x_1, y+a_n)) \prod_{i=1}^{r} \prod_{j=1}^{t_i} g_{ij} \equiv \bar{f}(x_1, y+a_n) \bmod y$$

such that the lifted $g_{ij}^{(k)} \in F[x, y]$ remain monic in x_1 and are of sufficiently high degree k in y (see substep 3).

Substep 3: The $g_{ij}^{(k)}$ correspond to true factors $\tilde{g}_{ij}/l_{ij}$, $\tilde{g}_{ij} \in F[x, y]$, $l_{ij} \in F[y]$, of $\bar{f}(x_1, y+a_n)$ in $F(y)[x]$. Let

$$g_{ij}^{(k)} = x^d + \gamma_{d-1}(y)x^{d-1} + \cdot \cdot \cdot + \gamma_0(y), \quad \gamma_m(y) \in F[y].$$

For $0 \le m \le d-1$ compute $p_m(y)/q_m(y)$, the highest order Padé approximation for $\gamma_m(y)/y^k$. We refer to Czapor and Geddes [4] for a comparison of several Padé approximation algorithms. Set $l_{ij} \leftarrow \mathrm{LCM}_{0 \le m \le d-1}(q_m(y))$. Notice that in practice a few m will already determine the l_{ij} correctly. Also, the degree bound k depends on $\deg(l_{ij})$ and thus the $\tilde{g}_{ij}$ are correctly detected at different lifting levels k.
Return $\tilde{g}_0$, $\tilde{g}_i \leftarrow \prod_{j=1}^{t_i} \tilde{g}_{ij}^{j}(x_1, x_n - a_n)$. $\square$

Notice that the above method might fail under two different circumstances, namely when the squarefree factorization of g_i is not a true image of that of $\tilde{g}_i$ and when the GCD of γ_m and y^k is of too high a degree. We leave it to the reader to establish the conditions for a_n such that no failure can occur. For randomly chosen a_n this turns out to be very rare. The reason why we suggest the squarefree factorization in substep 1 whereas we do not unconditionally use it in the Sparse Hensel Lifting algorithm (see also remark 1 following the algorithm) is because in the bivariate case the size of the lifting problem is certainly reduced.

Remark 2: In step 1 we processed the minor variables in the order of their subscripts. This order may by no means be optimal and in practice it appears better to process the minor variable of maximum individual degree first. It should also be clear that the Leading Coefficient algorithm applies for the pre-determination of trailing coefficients as well. We recommend to employ this additional process in step 1 of the Sparse Lifting algorithm if the trailing coefficient of f has many monomials. Then the linear systems of step 5.4 have a greatly reduced number of unknowns.

Remark 3: In step 3 we not only need the standard GCD-free basis $\{\tilde{b}_1, \ldots, \tilde{b}_s\}$ of $\{\tilde{l}_i\}$ but also the exponent vectors $u_{i,1\ldots s}$. It is, however, an easy modification of the Standard GCD-Free Basis algorithm to compute the vectors along with the c_{t+1}. Again the details are left to the reader.

We now establish the correctness of the leading coefficient algorithm.

Theorem 5.1: The true factors and true leading terms assumptions imply that the true factors assumption is satisfied for the (first level) calls from the Leading Coefficient Determination algorithm to the Sparse Hensel Lifting algorithm. Moreover, the correct content and leading coefficients are computed provided that the recursive calls have also returned the correct results.

Proof: We show the statements along with providing that (†) is a loop invariant. Following the execution of steps 2 and 3 with loop index n we first get by the true factors assumption, assumption i), ii) and Gauss's lemma that

$$\tilde{g}_0 \sim \lambda_0(a_{2\ldots n-1}, x_n, a_{n+1\ldots v}),\ \ \mathrm{ldcf}_{x_1}(\tilde{g}_i) \sim \lambda_i(a_{2\ldots n-1}, x_n, a_{n+1\ldots v}),\ \ 1 \le i \le r.$$

From (†) for $n-1$ and i) we thus conclude (‡). By lemma 3.2 and 3.3 (ϕ being evaluation $x_{n+1} = a_{n+1}, \ldots, x_v = a_v$) we conclude from iii) that there exist $c_1, \ldots, c_s \in F$ such that

$$\{c_1\tilde{b}_1, \ldots, c_s\tilde{b}_n\} = \{\beta_{n1}(x_n, a_{n+1\ldots v}), \ldots, \beta_{n,s_n}(x_n, a_{n+1\ldots v})\}.$$

Therefore the true factors assumption is satisfied for the lifting process of step 3 and associates of $\prod_{j=n+1}^{v} f^{(x_j)}$ and $\beta_{n1}, \ldots, \beta_{n,s_n}$ are returned provided the called algorithm works correctly. This depends, of course, on whether the zero preservation assumption is satisfied for the given lifting problem as well as whether further recursive calls to the leading coefficient algorithm return the correct results. Therefore, $\prod_{j=1}^{s} b_j^{u_{ij}} \sim \lambda_i^{(x_n)}$ and the final updates of l_i in step 3 imply (†) for n. □

6. Analysis

We first establish a general condition under which all except the true factors assumption are satisfied.

Theorem 6.1: Consider the sparse Hensel lifting algorithm with the provision that the true factors assumption is satisfied. Let $d = \deg(h_0 \cdots h_r)$. The there exists a polynomial $\pi^{(h_{0\ldots r})}(x_{2\ldots v}) \in F[x_{2\ldots v}]$ such that

$$\deg\,(\pi^{(h_{0\ldots r})}) < v\,(2d+2)^{v+1}$$

and $\pi^{(h_{0\ldots r})}(a_{2\ldots v}) \neq 0$ implies that the algorithm returns the correct results.

The proof of this theorem is rather tedious due to the recursive nature of our algorithms and should be skipped in the first reading of this paper. Part ii) of the true leading terms assumption is guaranteed by the following general lemma which will become important also in §7.

Lemma 6.1: Let $f_1, \ldots, f_t, h \in D[x]$, D a UFD, $\deg(f_i) \leq d$, $\mathrm{GCD}(f_1, \ldots, f_t) \sim h$. Then there exists an $m \times m$ determinant $\Delta \in D \setminus \{0\}$, $m \leq 2d$, whose entries are coefficients of the f_i, such that for any ring-homomorphism $\phi\colon D \to E$, E a field, $\phi\Delta \neq 0$ implies $\mathrm{GCD}(\phi f_1, \ldots, \phi f_t) \sim \phi h$ in $E[x]$. (For a proof see lemma 3 in [8].) □

Proof of theorem 6.1: First consider the zero-preservation assumption on the top level. Let

$$\sigma_i = \prod_{n=1}^{v-1} \prod_{j_{1\ldots n}} q_i^{(j_{1\ldots n})}, \quad 1 \leq i \leq r.$$

It is easy to see that for each i there are at most $(v-1)\ \mathrm{mon}(h_i)$ polynomials $q_i^{(j_{1\ldots n})}$. Let $d_i = \deg(h_i)$. Then $\mathrm{mon}(h_i) < (d_i+1)^v$, $\deg(q_i^{(j_{1\ldots n})}) \leq d_i$. (Notice that the first estimate is a worst case estimate. If the h_i are sparse then the degree of $\pi^{(h_{0\ldots r})}$ will turn out much smaller.) Therefore

$$\deg(\sigma_i) < (v-1)\,(d_i+1)^v\ d_i \leq (v-1)\,(d+1)^v\ d_i.$$

We set $\pi_1 = \prod_{i=1}^r \sigma_i$. Hence

$$\deg(\pi_1) = \sum_{i=1}^r \deg(\sigma_i) < (v-1)\,(d+1)^v \sum_{i=1}^r d_i < (v-1)\ d\ (d+1)^v.$$

Now if $\pi_1(a_{2\ldots v}) \neq 0$ then the zero preservation assumption is satisfied. Second, we consider the true leading terms assumption on the top level. Let $\bar{f} = h_0 \cdots h_r$, $\bar{\lambda} = \mathrm{ldcf}_{x_1}(\bar{f})$. Assumption i) is equivalent to $(\mathrm{ldcf}_{x_n}(\bar{\lambda}))(a_{2\ldots n-1}, a_{n+1\ldots v}) \neq 0$. To guarantee assumption ii) we appeal to lemma 6.1. Let $\bar{f} = \sum_{i=1}^t f_i x_1^{i-1}$, $f_i \in F[x_{2\ldots v}]$, $x = x_n$, $D = F[x_{2\ldots n-1}, x_{n+1\ldots v}]$, ϕ evaluation at $x_2 = a_2, \ldots, x_{n-1} = a_{n-1}$, $x_{n+1} = a_{n+1}, \ldots, x_v = a_v$ and Δ_n the asserted determinant. Then $\Delta_n(a_{2\ldots n-1}, a_{n+1\ldots v}) \neq 0$ implies that

$$(\mathrm{cont}_{x_1}(f))(a_{2\ldots n-1}, a_{n+1\ldots v}) = \phi(\mathrm{GCD}_{1\leq i\leq t}(f_i)) = \mathrm{GCD}_{1\leq i\leq t}(\phi f_i)$$

$$= \mathrm{cont}_{x_1}(\bar{f}(x_1, a_{2\ldots n-1}, x_n, a_{n+1\ldots v})).$$

By Gauss's lemma it now follows that

$$\prod_{i=0}^{r} \phi(\mathrm{cont}_{x_1}(h_i)) = \phi(\mathrm{cont}_{x_1}(\bar{f})) = \mathrm{cont}_{x_1}(\phi \bar{f}) = \prod_{i=0}^{r} \mathrm{cont}_{x_1}(\phi h_i)$$

and, since $\phi(\mathrm{cont}_{x_1}(h_i))$ divides $\mathrm{cont}_{x_1}(\phi h_i)$, $\phi(\mathrm{cont}_{x_1}(h_i)) = \mathrm{cont}_{x_1}(\phi h_i)$ for all $0 \le i \le r$. Again by Gauss's lemma the last property implies assumption ii). We set $\pi_2 = \prod_{n=2}^{v} \Delta_n \, \mathrm{ldcf}_{x_n}(\bar{\lambda})$ and observe that

$$\deg(\pi_2) \le (v-1)\,(2d^2+d).$$

We now deal with assumption iii). Let $d_i = \deg_{x_n}(\beta_{ni})$, $d_j = \deg_{x_n}(\beta_{nj})$ and $k = \deg_{x_n}(\mathrm{GCD}(\beta_{ni}, \beta_{nj}))$. As a special case of lemma 6.1, assumption iii) is satisfied if and only if the leading coefficient of the k-th subresultant $\sigma_{ij} \in F[x_{n+1 \ldots v}]$ of β_{ni} and β_{nj} with respect to x_n does not vanish on evaluation at $x_{n+1} = a_{n+1}, \ldots, x_v = a_v$ (cf. Brown [2]). We observe that $\deg(\sigma_{ij}) \le 2 d_i d_j$. Let

$$\tau_n = \prod_{1 \le i < j \le s_n} \sigma_{ij} \quad \text{with} \quad \deg(\tau_n) \le \sum_{1 \le i < j \le s_n} 2 d_i d_j < \left(\sum_{i=1}^{s_n} d_i \right)^2 .$$

It is any easy consequence of the standard basis construction algorithm that

$$\sum_{i=1}^{s_n} d_i \le \sum_{i=0}^{r} \deg(\lambda_i^{(x_n)}) = \deg(\bar{\lambda}^{(x_n)}).$$

We now set $\pi_3 = \prod_{n=2}^{v} \tau_n$ and note that

$$\deg(\pi_3) < \sum_{n=2}^{v} \deg^2(\bar{\lambda}^{(x_n)}) < \left(\sum_{n=2}^{v} \deg(\bar{\lambda}^{(x_n)}) \right)^2 = \deg^2(\bar{\lambda}) < d^2.$$

We finally set

$$\pi^{(h_{0 \ldots r})} = \pi_1 \; \pi_2 \; \pi_3 \prod_{n=2}^{v} \pi^{(b_{n,0 \ldots s_n})},$$

where b_{nj} is b_j computed in step 3 of the Leading Coefficient algorithm for loop index n. Clearly, if $\pi^{(h_{0 \ldots r})}(a_{2 \ldots v}) \ne 0$ then the assumptions are satisfied and theorem 4.1 and 5.1 apply. It remains to estimate the degree of $\pi^{(h_{0 \ldots r})}$.

Let $d_n = \deg(b_{n0} \cdots b_{n,s_n})$. We now can prove by induction that for any $\delta \ge d_n$ and $w \ge v$, v the number of variables in $b_{n0}, \ldots, b_{n,s_n}$,

$$\deg(\pi^{(b_{n,0 \ldots s_n})}) < 2^{v-1} \, d_n \, ((v-1)(\delta+1)^w + 3wd).$$

Following is the induction argument for $h_0, \ldots, h_r$:

$$\deg(\pi^{(h_{0 \ldots r})}) < \deg(\pi_1 \, \pi_2 \, \pi_3) + \sum_{n=2}^{v} \deg(\pi^{(b_{n,0 \ldots s_n})})$$

$$< (v-1)\, d\, (d+1)^v + 3vd^2 + ((v-1)(d+1)^v + 3vd) \sum_{n=2}^{v} 2^{v-n} \, d_n$$

$$< ((v-1)(\delta+1)^w + 3w\delta)\ (d + \sum_{n=2}^{v} 2^{v-n}\ d)$$

$$= 2^{v-1}\ d\ ((v-1)(\delta+1)^w + 3w\delta).$$

We finally obtain the degree bound from

$$2^{v-1}\ d((v-1)(d+1)^v + 3vd) < v\ (2d+2)^{v+1}. \quad \square$$

It now follows from a lemma by Schwartz [13] that if we select the a_i randomly from a set with (deg $\pi^{(h_{0\ldots r})}$) / ε elements then the probability that $\pi^{(h_{0\ldots r})}(a_{2\ldots v}) = 0$ becomes less then ε. It should be noted, however, that the very high degree bound of theorem 6.1 is only of theoretical interest. First, because if all intermediately computed h_i are sparse, the bound is much smaller. Second, because even if the degree of $\pi^{(h_{0\ldots r})}$ is of the same order as our estimate, the chance that we select a zero point of $\pi^{(h_{0\ldots r})}$ is, in practice, much smaller than ε.

The correctness of the true factors assumption must be imported into our algorithm. In factorization applications effective Hilbert irreducibility theorems are needed, cf. von zur Gathen [6] and Kaltofen [8]. For GCD applications conditions guaranteeing the true factors assumption are not as difficult to find (see theorem 7.1).

We wish to point out that our sparse Hensel lifting algorithm can have super-polynomial expected running time in the number of monomials of the result. The reason is that dense factors might accumulate in l_i during the execution of the leading coefficient determination algorithm, e.g., if

$$\lambda_i = \prod_{i=2}^{v} (\sum_{j=0}^{d} x_i^j),$$

or that the b_j might be dense while the l_i are not. An example for this case is

$$r = 2,\ \lambda_0 = 1,\ \lambda_1 = \prod_{i=3}^{v} (x_2^d - x_i^d),\ \lambda_2 = \prod_{i=3}^{v} (x_2 - x_i),$$

where mon(λ_1) = mon(λ_2) = 2^{v-2} but mon(λ_1/λ_2) = d^{v-2}.

Note† that also von zur Gathen's algorithm [7] has super-polynomial running time in terms of r. The reason is that his as well as our algorithm is based on explicit sparse Hensel lifting. However, then step 5.3 and the corresponding step in von zur Gathen's algorithm cannot be carried out in polynomial time. The first polynomial-time solution to the complete sparse factoring, based on implicit representation, can be found in [11].

† added on December 23, 1988

7. Application: The Sparse EZ-GCD Algorithm

We now discuss how we can apply our sparse lifting algorithm to computing GCDs of sparse polynomials. We follow Moses's and Yun's [12] E(xtended) - Z(assenhaus) − GCD approach but incorporate several important changes. First, we do no compute the primitive parts of the inputs by recursive application of the GCD algorithm. This saves us the task of computing the GCD of many polynomials in one less variables, namely the coefficients of the inputs with respect to the main variable. Second, we do not impose any leading coefficient upon the image GCD before lifting, but determine the correct leading coefficient by our leading coefficient algorithm. The overall improvement to the EZ-GCD algorithm is a substantial reduction in the number of uni- and multivariate GCD operations performed. It should be clear that the reference to the Sparse Lifting algorithm by itself speeds up the EZ-GCD process. One major advantage is, as we shall see below, that we can lift factors with higher multiplicities. We remark that Wang [16] also suggests assorted improvements to the EZ-GCD algorithm, most significantly the leading coefficient predetermination method. Since Wang had such a method only for integral coefficients, this of his improvements to the EZ-GCD algorithm was necessarily restricted to the integral case.

Sparse EZ-GCD algorithm

Input: $f_1, \ldots, f_r \in F[x_{1\ldots v}]$, F a field, $a_1, \ldots, a_v \in F$ and $g \in F[x_1] \setminus F$ such that

$$\mathrm{ldcf}_{x_1}(f_1 \cdot \cdot \cdot f_r)(a_{2\ldots v}) \neq 0 \quad \text{and} \quad g = \mathrm{GCD}_{1 \le i \le r}(f_i(x_1, a_{2\ldots v})).$$

We make the following assumptions on $a_1, \ldots, a_v$:

True GCD assumption: Let $h = \mathrm{GCD}_{1 \le i \le r}(f_i)$. Then $h(x_1, a_{2\ldots v}) \sim g$.

True cofactor assumption: Let $\{b_1, \ldots, b_s\}$ be a standard GCD-free basis for $\{f_1^{(x_1)}, h^{(x_1)}\}$. We assume that

$$\mathrm{GCD}(b_i, b_j)(x_1, a_{2\ldots v}) \sim \mathrm{GCD}(b_i(x_i, a_{2\ldots v}), b_j(x_1, a_{2\ldots v})) \text{ for all } 1 \le i < j \le s.$$

True content assumption: We assume that

$$\mathrm{GCD}(\mathrm{cont}_{x_1}(f_1), f_2(a_1, x_{2\ldots v}), \ldots, f_r(a_1, x_{2\ldots v})) = \mathrm{cont}_{x_1}(h).$$

Finally, we assume that the calls to Sparse Lifting and the recursive calls to Sparse EZ-GCD return the correct results. Notice, however, that the true factors assumption for the call to Sparse Lifting follows from assumptions already made.

Output: $h \in F[x_{1\ldots v}]$ such that $h(x_1, a_{2\ldots v}) = g$ and $h \sim \mathrm{GCD}_{1 \le i \le r}(f_i)$. Notice that h is uniquely determined.

Step 1: Compute a standard GCD-free basis $\{\tilde{b}_1, \ldots, \tilde{b}_s\}$ for $\{\tilde{f}_1(x_1) = f_1(x_1, a_{2\ldots v}), g\}$. Let d_j, e_j, $1 \le j \le s$ be such that $\tilde{f}_1 = \prod_{j=1}^{s} \tilde{b}_j^{d_j}$, $g = \prod_{j=1}^{s} \tilde{b}_j^{e_j}$.

Call Sparse Lifting with

$$f_1 \equiv \prod_{j=1}^{s} \tilde{b}_j^{d_j} \bmod (x_2 - a_2, \ldots, x_v - a_v).$$

The polynomials $b_0 = \mathrm{cont}_{x_1}(f_1)$, $b_1, \ldots, b_s$ are returned.
Set $h^{(x_1)} \leftarrow \prod_{j=1}^{s} b_j^{e_j}$.

Step 2: Compute

$$\tilde{g} = \mathrm{GCD}(b_0(x_2, a_{3\ldots v}),\ f_2(a_1, x_2, a_{3\ldots v}), \ldots, f_r(a_1, x_2, a_{3\ldots v}))$$

by a univariate GCD algorithm (see remark 2).
Call Sparse EZ-GCD recursively with

$$b_0,\ f_2(a_1, x_{2\ldots v}), \ldots, f_r(a_1, x_{2\ldots v}) \text{ and } \tilde{g}.$$

The polynomial $\bar{h} \in F[x_{2\ldots v}]$ is returned.

Step 3: Adjust output: Set $h \leftarrow \bar{h}\, h^{(x_1)}$.
Return $h \leftarrow \mathrm{ldcf}(g) / \mathrm{ldcf}_{x_1}(h)(a_{2\ldots v})\, h$. □

Remark 1: We arbitrarily chose f_1 for the lifting process. However, it can be more efficient to select the polynomial with the fewest terms among the f_i in place of f_1.

Remark 2: To provide the input g and to compute $\tilde{g}$ requires a univariate GCD computation of r polynomials. It can be shown that with high probability

$$\mathrm{GCD}(g_1, \ldots, g_r) = \mathrm{GCD}(\sum_{i=1}^{r} \lambda_i g_i, \sum_{i=1}^{r} \mu_i g_i)$$

for $g_1, \ldots, g_r \in F[x]$, $\lambda_1, \ldots, \lambda_r$, $\mu_1, \ldots, \mu_r \in F$ randomly selected (cf. [10], Theorem 5.2). This observation probabilistically reduces the problem to a single univariate GCD computation. We should add that the univariate GCD problem over **Q** can also be solved by the EZ-GCD algorithm. The process of lifting with multiplicities still applies because the true factors assumption can be enforced under these special circumstances.

Remark 3: If one of the assumptions is violated, the algorithm might return an incorrect result. We can, however, check whether h divides $f_2, \ldots, f_r$. If that is so h is guaranteed to be the correct GCD. In order to perform the sparse polynomial division we can make use of Sparse Lifting again, similarly to the Sparse EZ-GCD algorithm. This check will also produce the co-factors.

We now carry out the analysis.

Theorem 7.1: Let $d = \max\{\deg(f_i) \mid 1 \le i \le r\}$. Then there exist a polynomial $\rho^{(f_{1\ldots r})}(x_{1\ldots v}) \in F[x_{1\ldots v}]$ such that

$$\deg(\rho^{(f_{1\ldots r})}) < (v^2 + 1)\,(2d + 2)^{v+1}$$

and $\rho^{(f_{1\ldots r})}(a_{1\ldots v}) \neq 0$ implies that all assumptions to the Sparse EZ-GCD algorithm are satisfied.

Proof: We first consider the true GCD assumption. We can apply lemma 6.1 to $f_1, \ldots, f_r$, $x = x_1$, $D = F[x_{2\ldots v}]$, ϕ evaluation at $x_2 = a_2, \ldots, x_v = a_v$ and Δ_1 the asserted determinant. Then $\phi\Delta_1 \neq 0$ implies that $\mathrm{GCD}(\phi f_1, \ldots, \phi f_r) \sim \phi h$, which is the true GCD assumption, and we observe that $\deg(\Delta_1) \leq 2d^2$. The true cofactor assumption gives raise to a polynomial $\sigma \in F[x_{2\ldots v}]$ of degree smaller than $\deg^2(f_1) \leq d^2$ whose zeros have to be avoided by $a_2, \ldots, a_v$. The construction of σ is the same as that of π_3 in the proof of theorem 6.1.

In order to guarantee the true content assumption we resort again to lemma 6.1 with $f_1^{(x_n)}$, $f_2, \ldots, f_r$, $x = x_n$, $D = F[x_{1\ldots n-1}, x_{n+1\ldots v}]$, $E = F(x_{2\ldots n-1}, x_{n+1\ldots v})$ and ϕ evaluation at $x_1 = a_1$ ($2 \leq n \leq v$). Let Δ_n be the determinant asserted in the lemma, $\tau_n \in F[x_1] \setminus \{0\}$ a coefficient of $x_2^{j_2} \cdots x_{n-1}^{j_{n-1}} x_{n+1}^{j_{n+1}} \cdots x_v^{j_v}$ in Δ_n. Then $\phi\tau_n = \tau_n(a_1) \neq 0$ implies that

$$\mathrm{GCD}(f_1^{(x_1)}, \phi f_1, \ldots, \phi f_r) \sim h^{(x_n)}$$

in $E[x_n]$. Since $f_1^{(x_1)}$ is primitive in x_1, we get the same GCD in $F[x_{2\ldots v}]$. We notice that $\deg(\tau_n) \leq 2d^2$. Now if $\tau_n(a_1) \neq 0$ for all $2 \leq n \leq v$ then

$$\mathrm{GCD}(\mathrm{cont}_{x_1}(f_1), \phi f_2, \ldots, \phi f_r) \sim \prod_{n=2}^{v} h^{(x_n)} = \mathrm{cont}_{x_1}(h),$$

because all $f_1^{(x_n)}$ are relatively prime. We set $\tau = \prod_{n=2}^{v} \tau_n$ and remark that $\deg(\tau_n) \leq 2(v-1)d^2$.

We now can set

$$\rho^{(f_{1\ldots r})} = \Delta_1 \; \sigma \; \tau \; \pi^{(b_{0\ldots s})} \; \rho^{(b_0,\, f_2(a_1,\, x_{2\ldots v}), \ldots,\, f_r(a_1,\, x_{2\ldots v}))},$$

where $\pi^{(b_{0\ldots s})}$ is defined as in theorem 6.1. Since $\deg(b_0 \cdots b_s) \leq \deg(f_1) \leq d$ it follows from theorem 6.1 and by induction that

$$\deg(\rho^{(f_{1\ldots r})}) < v\,((2v+1)\,d^2 + v\,(2d+2)^{v+1}) < (v^2+1)\,(2d+2)^{v+1}. \quad \square$$

One final remark is in order. The Sparse EZ-GCD algorithm does not run in expected polynomial-time in the size of the inputs and outputs for similar reasons as the ones given in §6 for the Sparse Lifting algorithm. Our results in [10] have as a consequence a polynomial-time solution for the sparse GCD problem.

8. Conclusion

Our results hopefully put the last of the problems for sparse multivariate lifting, the leading coefficient problem, to rest. We have also established, by example of the sparse EZ-GCD algorithm, that content computations can be avoided prior to the lifting process. Finally we have introduced the notion of standard GCD-free basis and made precise its critical-pair completion construction, a process that has been used in the folklore of computer algebra in the past.

Our formulation of the Newton-Hensel Lemma has helped us to simplify the proof of the Main Theorem of Sparse Lifting. The full power of this approach becomes most apparent when applying it to factoring polynomials given by straight-line programs, such as polynomial determinants. In [10] and [11] we begin to develop a theory for computing with polynomials given by straight-line programs and established as one of the major results an expected polynomial-time procedure for factoring a polynomial given by a straight-line program into its sparse factors.

References:

[1] Bareiss, E.H.: Sylvester's identity and multistep integers preserving Gaussian elimination. *Math. Comp.* **22**, 565-578 (1965).

[2] Brown, W.S.: On Euclid's algorithm and the computation of polynomial greatest common divisors. *J. ACM* **18**, 478-504 (1971).

[3] Buchberger, B., Loos, R.: Algebraic Simplification. *Computing*, Supplement **4**, 11-43 (1982).

[4] Czapor, S.R., Geddes, K.O.: A comparison of algorithms for the symbolic computation of Padé approximants. Proc. EUROSAM 1984, *Springer Lec. Notes Comp. Sci.* **174**, 248-259.

[5] Epstein, H.I.: Using basis computation to determine pseudo-multiplicative independence. *Proc. 1976 ACM Symp. Symbolic Algebraic Comp.*, 229-237.

[6] von zur Gathen, J.: Irreducibility of multivariate polynomials. *J. Comp. System Sci.*, to appear.

[7] von zur Gathen, J.: Factoring sparse multivariate polynomials. *Proc. 24th IEEE Symp. Foundations Comp. Sci.*, 172-179 (1983).

[8] Kaltofen, E.: Effective Hilbert Irreducibility. Proc. EUROSAM 1984, *Springer Lec. Notes Comp. Sci.* **174**, 277-284.

[9] Kaltofen, E.: On a theorem by R. Dedekind. Manuscript 1984.

[10] Kaltofen, E.: Computing with polynomials given by straight-line programs I; Greatest Common Divisors. *Proc. 17th ACM Symp. Theory Comp.*, to appear (1985).

[11] Kaltofen, E.: Computing with polynomials given by straight-line programs II; Factorization. In preparation.

[12] Moses, J., Yun, D.Y.Y.: The EZ-GCD algorithm. *Proc 1973 ACM National Conf.*, 159-166.

[13] Schwarz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. *J. ACM* **27**, 701-717 (1980).

[14] Viry, G.: Factorisation des polynômes a plusieurs variables. *R.A.I.R.O. Informatique Théoretique* **17**, 209-223 (1980).

[15] Wang, P.S.: An improved multivariate polynomial factorization algorithm. *Math. Comp.* **32**, 1215-1231 (1978).

[16] Wang, P.S.: The EEZ-GCD algorithm. *SIGSAM Bulletin* **14**-2, 50-60 (May 1980).

[17] Wang, P.S.: Early detection of true factors in univariate polynomial factorization. Proc. EUROCAL 1983, *Springer Lec. Notes Comp. Sci.* **162**, 225-235.

[18] Yun, D.Y.Y.: The Hensel lemma in algebraic manipulation. Ph.D. dissertation, MIT 1974. Reprint: Garland Publ. Co., New York 1980.

[19] Zippel, R.E.: Newton's iteration and the sparse Hensel algorithm. *Proc. 1981 ACM Symp. Symbolic Algebraic Comp.*, 68-72.