On Factorization of Multivariate Polynomials over Algebraic Number and Function Fields *

Seyed Mohammad Mahdi Javadi
School of Computing Science
Simon Fraser University
Burnaby, B.C. Canada.
sjavadi@cecm.sfu.ca.

Michael Monagan
Department of Mathematics
Simon Fraser University
Burnaby, B.C. Canada.
mmonagan@cecm.sfu.ca.

ABSTRACT

We present an efficient algorithm for factoring a multivariate polynomial $f \in L[x_1, \ldots, x_v]$ where L is an algebraic function field with $k \geq 0$ parameters and $r \geq 0$ field extensions. Our algorithm uses Hensel lifting and extends the EEZ algorithm of Wang which is designed for factorization over rationals. We also give a multivariate p-adic lifting algorithm which uses sparse interpolation. This enables us to avoid using poor bounds on the size of the integer coefficients in the factorization of f when using Hensel lifting.

We have implemented our algorithm in Maple 13. We provide timings demonstrating the efficiency of our algorithm.

1. INTRODUCTION

In a computer algebra system, computations with polynomials over algebraic function fields such as computing GCDs and factorization arise, for example, when one solves nonlinear polynomial equations involving parameters.

One way to factor f is to use Trager's algorithm [**?**]. His algorithm computes and factors the $\text{norm}(f)$ which is a polynomial in $x_1, \ldots, x_v$ over $\mathbb{Q}(t_1, \ldots, t_k)$ where $t_1, \ldots, t_k$ are parameters. Trager's algorithm exploits the fact that if $f_i \mid f$ then $h_i = \text{norm}(f_i) \mid \text{norm}(f)$ and h_i is an irreducible factor of $\text{norm}(f)$ if and only if $f_i = \gcd(f, h_i)$ is an irreducible factor of f. One problem is that the $\text{norm}(f)$ can be much larger than f. For example the norm of

$$f = \frac{19}{2}c_4^2 - \sqrt{11}\sqrt{5}\sqrt{2}c_5c_4 - 2\sqrt{5}c_1c_2 - 6\sqrt{2}c_3c_4 + \frac{3}{2}c_0^2 + \frac{23}{2}c_5^2 +$$

$$\frac{7}{2}c_1^2 - \sqrt{7}\sqrt{3}\sqrt{2}c_3c_2 + \frac{11}{2}c_2^2 - \sqrt{3}\sqrt{2}c_0c_1 + \frac{15}{2}c_3^2 - \frac{10681741}{1985},$$

is degree 64 in $c_0, c_1, c_2, c_3, c_4, c_5$ and has about 3 million terms and the integers in the rational coefficients have over 200 digits so it is not easy to compute $\text{norm}(f)$ let alone factor it. Here $L = \mathbb{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \sqrt{7}, \sqrt{11})$ is a number field and $f \in L[c_0, \ldots, c_5]$. But we can easily prove that f is irreducible by evaluating the variables $c_0, \ldots, c_4$ at small integers and then using Trager's algorithm. In this paper we generalize this to factor polynomials in $L[x_1, \ldots, x_v]$ using Hensel lifting. We evaluate all parameters and all variables except one, thus reducing the factorization in $L[x_1, \ldots, x_v]$ to $L(\alpha)[x_1]$.

Some algorithms (See [**?**, **?**, **?**]) have been developed for factorization over an algebraic field L. A challenge is to solve the leading coefficient problem for lifting non-monic polynomials. Abbott in [**?**], suggests using a trick by Kaltofen in [**?**] which recursively computes the leading coefficients from their bivariate images using Hensel lifting. Our approach is to modify Wang's ingenious method given in [**?**] for factoring polynomials over $\mathbb{Z}$. His idea is to first factor the leading coefficient $l(x_2, \ldots, x_v) = \text{lc}_{x_1}(f)$ of the input polynomial f in the main variable x_1 recursively. Then evaluate all the variables except x_1 at an evaluation point α and factor the univariate polynomial $f(\alpha)$. Now using the integer leading coefficients of the univariate factors, one can determine which factor of $l(x_2, \ldots, x_v)$ belongs to the leading coefficient of which factor of $f(\alpha)$. To do this, Wang identifies unique prime divisors for each factor of $l(x_2, \ldots, x_v)$ evaluated at α by computing integer GCDs only. Unfortunately this does not generalize. We show an example.

Example 1 *Let* $L = \mathbb{Q}(\sqrt{-5})$ *and*

$$f = ((y + \sqrt{-5} + 1)x + 1)((y + \sqrt{-5} - 1)x + 1)$$

$$= (y^2 + 2\sqrt{-5}y - 6)x^2 + 2(y + \sqrt{-5})x + 1.$$

We have $lc_{x,y}(f) = 1 \in \mathbb{Z}$ *but* $lc_x(f) = y^2 + 2\sqrt{-5}y - 6 \in L[y]$*, so if we evaluate* y *at an evaluation point, we will obtain an element of* $\mathbb{Z}[\sqrt{-5}]$*. But* $\mathbb{Z}[\sqrt{-5}]$ *is not a unique factorization domain and so GCDs do not always exist in this ring. For example, for* $y = 0$ *we have* $lc_x(f)(y = 0) = -6 = -2 \times 3 = -(1 - \sqrt{-5}) \times (1 + \sqrt{-5})$.

Another problem is that one needs to do computations with fractions in Hensel lifting. To solve this problem, one can work modulo a power of a prime, p^l. This modulus must be at least twice the size of the biggest coefficient in any factor of f. Unfortunately the known bounds on the sizes of the integer coefficients in the factors of f are usually very big, which makes the computations really slow. In [**?**] it is suggested that it is better not to do the calculations modulo p^l because of the bad bounds but instead to lift over $\mathbb{Q}$. In our algorithm we choose a prime p of a modest size and then lift the integer coefficients to their correct values using a

*Supported by NSERC of Canada and the MITACS NCE of Canada

ISSAC'09, July 28–31, 2009, Seoul, Korea.

new multivariate p-adic lifting algorithm which uses a sparse interpolation method similar to Zippel's algorithm [**?**].

Our paper is organized as follows. In Section 2 we present an example showing the main flow and the key features of our algorithm. We then identify possible problems that can occur and how the new algorithm deals with them in Section 3. In Section 4 we present our new algorithm. Finally, in Section 5 we compare Maple implementations of our algorithm with Trager's algorithm for a set of polynomials. In our implementation of Trager's algorithm we use the SparseModGcd algorithm [**?**] to compute GCDs of polynomials over L, which makes Trager's algorithm much more efficient than it is otherwise.

2. AN EXAMPLE

Let $F = \mathbb{Q}(t_1, \ldots, t_k)$, $k \geq 0$. For i, $1 \leq i \leq r$, let $m_i(z_1, \ldots, z_i) \in F[z_1, \ldots, z_i]$ be monic and irreducible over $F[z_1, \ldots, z_{i-1}]/\langle m_1, \ldots, m_{i-1}\rangle$. Let $L = F[z_1, \ldots, z_r]/\langle m_1, \ldots, m_r\rangle$. L is an algebraic function field in k parameters $t_1, \ldots, t_k$ (this also includes number fields). Let f be a non-zero square-free polynomial in $L[x_1, \ldots, x_v]$. Our problem is given square-free f, compute $f_1, f_2, \ldots, f_n$ such that $f = \mathrm{lc}_{x_1,\ldots,x_v}(f) \times f_1 \times f_2 \times \cdots \times f_n$ where f_i is a monic irreducible polynomial in $L[x_1, \ldots, x_v]$.

Our algorithm works with the *monic associate* of the input and *primitive associates* of the minimal polynomials which we now define.

Definition 1 *Let $D = \mathbb{Z}[t_1, \ldots, t_k]$. A non-zero polynomial in $D[z_1, \ldots, z_r, x_1, \ldots, x_v]$ is said to be primitive wrt $(z_1, \ldots, z_r, x_1, \ldots, x_v)$ if the GCD of its coefficients in D is 1. Let f be non-zero in $L[x_1, \ldots, x_v]$. The denominator of f is the polynomial $den(f) \in D$ of least total degree in $(t_1, \ldots, t_k)$ and with smallest integer content such that $den(f)f$ is in $D[z_1, \ldots, z_r, x_1, \ldots, x_v]$. The primitive associate $\tilde{f}$ of f is the associate of $den(f)f$ which is primitive in $D[z_1, \ldots, z_r, x]$ and has positive leading coefficient in a term ordering. The* monic associate *$\check{f}$ of f is defined as $\check{f} = \check{h}$ where $h = monic(f)$. Here $monic(f)$ is defined by $lc_{x_1,\ldots,x_v}(f)^{-1} f$.*

Example 2 *Let $f = 3tx^2 + 6tx/(t^2-1) + 30tz/(1-t)$ where $m_1(z) = z^2 - t$. Here $f \in L[x]$ where $L = \mathbb{Q}(t)[z]/\langle z^2 - t\rangle$ is an algebraic function field in one parameter t. We have $\mathrm{den}(f) = t^2 - 1$ and $\tilde{f} = \check{f} = \mathrm{den}(f)f/(3t) = (t^2-1)x + 2x - 10z(t+1)$. For $f = zx^2 + 1/t$ we have $\check{f} = tzx^2 + 1$, $monic(f) = x^2 + z/t^2$ and $\tilde{f} = t^2x^2 + z$. Note $\tilde{f} \neq \check{f}$ in general.*

We demonstrate our algorithm using the following example. Here we use t for the parameter and x and y for variables.

Example 3 *Let*

$$f = (t^3 - t)y^2x^2 + (20t^3z - t^2z - 20tz + z)yx^2 +$$

$$(-20t^5 + 40t^3 - 20t)x^2 + (-tz + 21z)yx + (421t^3 - 421t)x - 21t$$

$$= (t^3 - t)(xy + 20zx - \frac{z}{t^2-1})(xy - \frac{zx}{t} + \frac{21z}{t^3 - t})$$

and $m(z) = z^2 - t^3 + t$. Here $L = \mathbb{Q}(t)[z]/\langle z^2 - t^3 + t\rangle$ and $f \in L[x, y]$. We have $\check{f} = f$ and $\check{m} = m$. The first step in our algorithm is to eliminate any field extensions in $l = lc_{x,y}(\check{f}) = t^3 - t$ by computing $\tilde{f}$. Since l does not involve the algebraic extension z, we have $\tilde{f} = \check{f}$.

Suppose we choose x as the main variable. In order to use Hensel lifting we need to factor the leading coefficient

$$lc_x(\tilde{f}) = (t^3 - t)y^2 + (20t^3z - t^2z - 20tz + z)y - 20t^5 + 40t^3 - 20t.$$

We do this by recursively using our algorithm, this time with one less variable. We will obtain

$$lc_x(\tilde{f}) = \gamma \times \bar{l}_1 \times \bar{l}_2 = (t^2 - 1)(ty - z)(y + 20z).$$

We factor $\gamma = t^2 - 1 = \bar{l}_3 \times \bar{l}_4 = (t-1)(t+1)$. In order to factor $\tilde{f}$, we evaluate it at the point α for all the parameters and variables except the main variable, x. We factor the resulting univariate polynomial in $\mathbb{Q}[z][x]/\langle \check{m}(\alpha)\rangle$ using Trager's algorithm and then we lift the variables and parameters one by one using Hensel lifting. Suppose we choose the evaluation point to be $\alpha = (t = 12, y = 5)$. This evaluation point must satisfy certain conditions that we will discuss in Section 3.1. We have

$$\tilde{f}(\alpha) = (170885z - 4864860)\,x^2 + (45z + 722436)\,x - 252$$

and $\check{m}(\alpha) = z^2 - 1716$ hence $L(\alpha)$ is a field. Using Trager's algorithm we obtain

$$\tilde{f}(\alpha) = lc_x(\tilde{f}(\alpha)) \times u_1 \times u_2 = (170885z - 4864860) \times$$

$$(x + \frac{1}{19630325}z - \frac{48}{137275}) \times (x + \frac{105}{22451}z + \frac{21}{157}).$$

Before doing Hensel lifting, we need to determine the true leading coefficient of each factor of $\tilde{f}$. To do this, we use the denominators of u_1 and u_2. We know that

$$d_i = den(u_i) \mid den(\frac{1}{lc_x(\tilde{f}_i(\alpha))})$$

where $\tilde{f}_i$ is a factor of $\tilde{f}$. We have

$$d_1 = den(u_1) = 19630325 = (5)^2(11)(13)(17)^2(19),$$
$$d_2 = den(u_2) = 22451 = (11)(13)(157),$$
$$\bar{d}_1 = den(1/\bar{l}_1(\alpha)) = 1884 = (2)^3(3)(157),$$
$$\bar{d}_2 = den(1/\bar{l}_2(\alpha)) = (5)^2(17)^2(19),$$
$$\bar{d}_3 = den(1/\bar{l}_3(\alpha)) = 11,$$
$$\bar{d}_4 = den(1/\bar{l}_4(\alpha)) = 13.$$

The evaluation point α was chosen so that $\bar{d}_i$'s have a set of distinct prime divisors, namely $\{3, 17, 11, 13\}$. Here $\bar{d}_i$'s are relatively prime so we have

$$\gcd(d_i, \bar{d}_j) > 1 \Rightarrow \bar{l}_j \mid \tilde{l}_i$$

where $\tilde{l}_i = lc_{x_1}(\tilde{f}_i)$. Using this we obtain $\tilde{l}_1 = (t^2-1)(y + 20z)$ and $\tilde{l}_2 = (t^2-1)(ty - z)$ and we have

$$\tilde{f} \equiv \frac{1}{t^2 - 1} \times (\tilde{l}_1(\alpha)u_1) \times (\tilde{l}_2(\alpha)u_2) \bmod \langle t - 12, y - 5\rangle.$$

To avoid fractions we set

$$\tilde{f} := \frac{\tilde{l}_1 \times \tilde{l}_2}{lc_{x_1}(\tilde{f})} = (t^2 - 1) \times \tilde{f}.$$

Now we use Hensel lifting to lift the parameter t and the variable y in the other coefficients of $\tilde{f}_i$. To avoid any computations with fractions, we do the calculations modulo a

prime, say $p = 17$. *After applying Hensel lifting we obtain* $\tilde{f}_1 = ((t^2-1)(y+20z)x-z)$ *and* $\tilde{f}_2 = ((t^2-1)(ty-z)x+4z)$ *s.t.* $\tilde{f} \equiv \tilde{f}_1 \times \tilde{f}_2 \pmod{17}$. *The final task is to lift the integer coefficients of* $\tilde{f}_1$ *and* $\tilde{f}_2$. *To do this, we use sparse interpolation. We have* $e_1 = \tilde{f} - \tilde{f}_1 \times \tilde{f}_2$, *the first error polynomial. We want to find* $\sigma_1, \sigma_2 \in L[x,y]$ *s.t.*

$$\tilde{f} \equiv (\tilde{f}_1 + \sigma_1 \times p)(\tilde{f}_2 + \sigma_2 \times p) \bmod p^2.$$

Assuming that neither α *nor* p *has caused any terms in the polynomials* $\tilde{f}_1$ *and* $\tilde{f}_2$ *to vanish, we know that the monomials in* σ_1 *and* σ_2 *are the same as those in* $\tilde{f}_1$ *and* $\tilde{f}_2$ *respectively, so we have the* assumed forms *for* σ_1 *and* σ_2. *Since* $\tilde{f}_1$ *and* $\tilde{f}_2$ *have correct leading coefficients we have* $\sigma_1 = Az$ *and* $\sigma_2 = Bz$ *for unknown coefficients* A *and* B. *To find the values for* A *and* B *we have*

$$\sigma_1 \times \tilde{f}_2 + \sigma_2 \times \tilde{f}_1 - \frac{e_1}{p} \equiv 0 \bmod p.$$

After equating every coefficient in x, y, z *and* t *in the above expression to zero, we will get the following system of linear equations*

$$\{A = 0, -B + 1 = 0, B - 1 = 0, -1 - 4A + B = 0, 1 - B + 4 = 0,$$
$$A = 0, -A - 20 + 20B = 0, 2A + 40 - 40B = 0, -A = 0\}.$$

After solving this linear system of equations modulo p, *we find that* $A = 0$ *and* $B = 1$ *so we update*

$$\tilde{f}_1 := \tilde{f}_1 + \sigma_1 \times p = ((t^2-1)(y+20z)x - z)$$

and

$$\tilde{f}_2 := \tilde{f}_2 + \sigma_2 \times p = ((t^2-1)(ty-z)x + 21z).$$

Now we have $\tilde{f} \equiv \tilde{f}_1 \times \tilde{f}_2 \bmod p^2$. *This time the new error* $e_2 = \tilde{f} - \tilde{f}_1 \times \tilde{f}_2$ *is zero, so we have* $\tilde{f} = \tilde{f}_1 \times \tilde{f}_2$. *To complete the factorization of* f *we have* $f = lc_{x,y}(f) \times monic(\tilde{f}_1) \times monic(\tilde{f}_2)$, *thus*

$$f = (t^3 - t)(xy + 20zx - \frac{z}{t^2-1})(xy - \frac{zx}{t} + \frac{21z}{t^3-t})$$

and we are done.

3. PROBLEMS

In the example we mentioned that the evaluation point must satisfy certain conditions in order for the algorithm to work properly. Another issue is the defect of the algebraic field which is the biggest denominator in an integral basis for the algebraic field L. Here we identify all problems.

The Defect

Unlike factorization over rationals, when factoring a polynomial $\tilde{f}$ over the algebraic field L, the leading coefficient of a factor $\tilde{f}_i$ in the variables $x_1, \ldots, x_v$ might not divide the leading coefficient of $\tilde{f}$, i.e. $\mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f}_i) \nmid \mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f})$.

Example 4 *Let* $m = z^2 - t^3$, $L = \mathbb{Q}(t)[z]/\langle m \rangle$ *and* $f = x^2 - t$. *We have*

$$f = (x - \frac{z}{t})(x + \frac{z}{t}) = \frac{1}{t^2}(tx - z)(tx + z).$$

Here $\tilde{f}_1 = tx - z$ *but* $lc_x(\tilde{f}_1) = t \nmid lc_x(\tilde{f}) = 1$.

The denominator t in this example is a factor of the *defect* of the algebraic function field L. The defect is the biggest denominator appearing in an integral basis for the algebraic field L (See [**?**, **?**]).

Theorem 1 (See [?]) *The defect is the biggest square that divides* Δ, *the discriminant of the algebraic field.*

When $r = 1$ (one field extension), $\Delta = \mathrm{res}_{z_1}(\check{m}_1, \check{m}_1')$. For example, for $\check{m} = z^2 - t^3$ we have $\Delta = \mathrm{res}_z(z^2 - t^3, 2z) = -4t^3$ and hence $2t$ is the defect.

Theorem 2 (See [?]) *Let* $d_i = \deg_{z_i}(\check{m}_i)$. *The discriminant of* L *is*

$$\Delta = \prod_{i=1}^{r-1} N_1(N_2(\ldots(N_{i-1}(discr(\check{m}_i)^{d_{i+1}\cdots d_r}))\ldots))$$

where $N_i(f) = res_{z_i}(f, \check{m}_i)$ *and* $discr(\check{m}_i) = res_{z_i}(\check{m}_i, \check{m}_i')$.

Suppose using Theorem 2 we have computed the discriminant $\Delta \in \mathbb{Z}[t_1, \ldots, t_k]$. Let $\delta \times D_1^{e_1} \times \cdots \times D_k^{e_k}$ be a square-free factorization of Δ where $\delta \in \mathbb{Z}$. Since we want to avoid integer factorization, we choose $\mathbb{D}$ to be an integer multiple of the defect:

$$\mathbb{D} = \delta \times D_1^{\lfloor \frac{e_1}{2} \rfloor} \times \cdots \times D_k^{\lfloor \frac{e_k}{2} \rfloor}.$$

Theorem 3 *If* $\tilde{f}_i$ *is a factor of* $\tilde{f}$ *and* $\mathbb{D}$ *is an integral multiple of the defect, then*

$$lc_{x_1,\ldots,x_v}(\tilde{f}_i) \mid \mathbb{D} \times lc_{x_1,\ldots,x_v}(\tilde{f})$$

Remark 1 To compute an integral multiple of $\mathbb{D}$ in our algorithm, we compute Δ using Theorem 2. We then do a square-free factorization of Δ/c where $c = \mathrm{cont}_{t_1,\ldots,t_k}(\Delta) \in \mathbb{Z}$ is the integer content of Δ, to find the biggest square D which divides Δ/c. We use $c \times D$ as the integral multiple of the defect.

Remark 2 As seen in Example 3, the leading coefficient of $\tilde{f}$ ($\mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f}) \in \mathbb{Z}[t_1, \ldots, t_k]$) may not *split* among the leading coefficients of the factors. That is $\prod_{i=1}^{n} \mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f}_i)$ may not divide $\mathbb{D}^l \times \mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f})$ for any $l \in \mathbb{Z}^+$.

3.1 Good evaluation points

Let $\alpha = (t_1 = \alpha_1, \ldots, t_k = \alpha_k, x_2 = \beta_2, \ldots, x_v = \beta_v) \in \mathbb{Z}^{k+v-1}$ be the evaluation point that we choose in our algorithm to factor the univariate polynomial $\tilde{f}(\alpha)$. It must satisfy certain conditions. We say α is good if:

1. The leading coefficient of $\tilde{f}$ in the main variable x_1 and the leading coefficient of $\check{m}_i$ in z_i must not vanish after evaluating at α, i.e. $\deg_{x_1}(\tilde{f}) = \deg_{x_1}(\tilde{f}(\alpha))$ and $\deg_{z_i}(\check{m}_i) = \deg_{z_i}(\check{m}_i(\alpha))$.

2. The second requirement on the evaluation point α is that $L(\alpha)$, the algebraic field evaluated at α, must remain a field in order to have a unique factorization. As an example, the evaluation point $t = 0$ is not a *good* choice for our Example 3 because the minimal polynomial evaluated at this point is no longer irreducible. This also happens for evaluation points $t = 1, 4, 9, 16, \ldots$ (a curiosity?).

3. The input polynomial evaluated at α, i.e. $\tilde{f}(\alpha) \in L(\alpha)[x_1]$, must remain square-free in x_1 so that we can apply Hensel lifting.

4. The fourth condition on the evaluation point α is to be able to distribute factors of $\mathrm{lc}_{x_1}(\tilde{f})$ to the monic univariate factors $u_1, \ldots, u_n$ where $u_i \in L(\alpha)[x_1]$ and

$$\tilde{f}(\alpha) = \mathrm{lc}_{x_1}(\tilde{f})(\alpha) \times u_1 \times \cdots \times u_n.$$

Suppose $\gamma \times \hat{l}_1^{e_1} \times \cdots \times \hat{l}_m^{e_m}$ is the factorization of $\mathrm{lc}_{x_1}(\tilde{f})$ and $\mathbb{D}$ is the defect. Here $\gamma \in \mathbb{Z}[t_1, \ldots, t_k]$ and $\hat{l} \in L[x_2, \ldots, x_n]$. Let $\beta = \mathbb{D} \times \gamma = \Omega \times \beta_1^{c_1} \times \beta_2^{c_2} \times \cdots \times \beta_k^{c_k}$ where $\Omega \in \mathbb{Z}$ and $\beta \in \mathbb{Z}[t_1, \ldots, t_k]$. Let $\bar{d}_i = \mathrm{den}(1/\hat{l}_i(\alpha))$. In order to be able to uniquely distribute the factors of $\mathbb{D} \times \mathrm{lc}_{x_1}(\tilde{f})$ to the univariate factors, the numbers in the set

$$A = \{\beta_1(\alpha), \ldots, \beta_k(\alpha), \bar{d}_1, \ldots, \bar{d}_m\}$$

must have distinct prime divisors that do not divide Ω.

Example 5 *In Example 3 we have* $lc_x(\tilde{u}_1) = 19630325$, $lc_x(\tilde{u}_2) = 22451$. *We have* $\beta_1 = t - 1, \beta_2 = t + 1$, $\hat{l}_1 = ty - z, \hat{l}_2 = y + 20z$ *and* $\Omega = 2$. *We can not use the evaluation point* $\alpha = (t = 7, y = 5)$ *because the numbers in* $A = \{6 = (2)(3), 8 = (2)^3, 889 = (7)(127), 26875 = (5)^4(43)\}$ *do not have distinct prime divisors. This is because the only prime that divides* 8 *is* 2 *which also divides* 6.

Remark 3 Condition 4 will not be satisfied, no matter what α is, if any two of the irreducible factors of $\mathrm{lc}_{x_1}(\tilde{f})$ have the same norm, i.e. $\exists i, j : \mathrm{norm}(\hat{l}_i) = \mathrm{norm}(\hat{l}_j)$ where $\hat{l}_i$ and $\hat{l}_j$ are irreducible factors of $\mathrm{lc}_{x_1}(\tilde{f})$. In this case, the denominators $\bar{d}_i = \mathrm{den}(1/\hat{l}_i(\alpha))$ and $\bar{d}_j = \mathrm{den}(1/\hat{l}_j(\alpha))$ will be images of the same polynomial $\mathrm{norm}(\hat{l}_i) = \mathrm{norm}(\hat{l}_j)$. In this case we shift the variables $x_2, x_3, \ldots$ in the input polynomial by computing

$$\tilde{f} := \tilde{f}(x_1, x_2 + S_2, x_3 + S_3, \ldots, x_v + S_v)$$

where $S_i = s_{i1}z_1 + s_{i2}z_2 + \cdots + s_{iz}z_r$ and $s_{ij} \in \mathbb{Z}$ (See [**?**]). The following is an example.

Example 6 *Let* $\breve{m} = z^2 - t$ *and* $\tilde{f} = ((y+z)x+t)((y-z)x+t)$. *We have* $lc_x(\tilde{f}) = \hat{l}_1 \times \hat{l}_2 = (y-z)(y+z)$ *and* $norm(\hat{l}_1) = norm(\hat{l}_2) = y^2 - t$. *If we choose* $\alpha = (y = 1, t = 6)$ *we will have* $\bar{d}_1 = den(1/\hat{l}_1(\alpha)) = 5$ *and* $\bar{d}_2 = den(1/\hat{l}_2(\alpha)) = 5$ *and the set* $A = \{5, 5\}$ *will not have a set of distinct prime divisors. If we shift the variable* y *to* $y + 3z$, *we will get* $\tilde{f} := \tilde{f}(x, y+3z) = ((y+4z)x+t)((y+2z)x+t)$ *and factors of* $lc_{x_1}(\tilde{f})$ *have different norms and the numbers in* $A = \{den(1/\hat{l}_1(\alpha)) = 23, den(1/\hat{l}_2(\alpha)) = 95\}$ *have distinct prime divisors.*

An evaluation point α is said to be *unlucky* if it does not satisfy any of the following conditions:

1. The number of irreducible factors of $\tilde{f}(\alpha)$ is the same as the number of irreducible factors of $\tilde{f}$.
2. If $\hat{l}_i \mid \mathrm{lc}_{x_1}(\tilde{f}_j)$ where $\tilde{f}_j$ is an irreducible factor of $\tilde{f}$ then $\gcd(\mathrm{den}(1/\hat{l}_i(\alpha)), \mathrm{lc}_{x_1}(\tilde{u}_j)) \neq 1$.
3. If $\beta_i \mid \mathrm{lc}_{x_1}(\tilde{f}_j)$ then $\gcd(\beta_i(\alpha), \mathrm{lc}_{x_1}(\tilde{u}_j)) \neq 1$

If α does not satisfy condition 2,3 or 4 then we will not be able to compute the correct leading coefficients of the factors. If the evaluation point α is unlucky, the algorithm must restart and choose a new evaluation point.

Remark 4 If α is unlucky and there are extraneous factors in the factorization of $\tilde{f}(\alpha)$ then Hensel lifting will fail with high probability. Hensel lifting may succeed with low probability if the prime p in Hensel lifting is also unlucky and results in extraneous factors in $\tilde{f} \bmod p$ corresponding to those of $\tilde{f}(\alpha)$.

Example 7 *Suppose* $\tilde{f} = x^2 + 17(t-1)zx - t^2$. *The evaluation point* $\alpha = (t = 1)$ *is unlucky because* $\tilde{f}$ *is irreducible but* $\tilde{f}(\alpha) = (x-1)(x+1)$. *If we choose* α *as the evaluation point and* $p = 17$, *Hensel lifting will succeed and return* $(x-t)(x+t)$.

If Hensel lifting does not fail when α is unlucky, then we will not be able to lift the integer coefficients of factors of $\tilde{f}$ and the algorithm will restart by choosing a new evaluation point.

A good idea is to choose a few evaluation points (instead of only one) and not start lifting until we get the same number of factors with consistent degrees for all the evaluation points.

Remark 5 Since we will use sparse interpolation to lift the integer coefficients of the factors computed using Hensel lifting, the evaluation point α must not have introduced any *missing terms* in any factors of $\tilde{f}$. That is no term in any factor of $\tilde{f}$ must vanish (including the leading coefficient) after evaluating at α. Unfortunately we will not be able to identify such bad evaluation points in advance. Instead, if α is unlucky and the forms for any of the correcting polynomials $\sigma_1, \sigma_2, \ldots$ is wrong, the system of linear equations in the sparse interpolation would be inconsistent with high probability. A similar problem can happen for the prime p that we choose as the modulus for Hensel lifting.

To decrease the probability of choosing an evaluation point (or a prime) that introduces *missing terms* in factors of $\tilde{f}$, one should choose α (and p) at random from a large set of evaluation points (or primes), e.g. $p = 2^{31} - 1$ and $\alpha \in \mathbb{Z}_p$ at random.

Degree Bound for the Parameters

In order to use Hensel lifting, we need to have bounds on the degrees of the parameters and variables in the factors of $\tilde{f}$. Unlike factorization over rationals, $\deg_{t_i}(\tilde{f}_i)$ is not necessarily less than or equal to $\deg_{t_i}(\tilde{f})$.

Example 8 *Let* $m = z^2 - \frac{1}{t^3}$ *and* $\tilde{f} = x^2 - t$. *We have*

$$\tilde{f} = \tilde{f}_1 \tilde{f}_2 = (x + t^2 z)(x - t^2 z).$$

Here $\deg_t \tilde{f}_1 = \deg_t \tilde{f}_2 = 2 > \deg_t \tilde{f} = 1$.

In [**?**] Abbott, gives a possible bound T_i on the degree of each factor in t_i based on Trager's algorithm which is usually much bigger than the actual degrees of the factors. In our algorithm when we lift the parameter t_i in the factorization of $\tilde{f}$, as soon as the factors have been lifted to the correct degree, the error would be zero with high probability and the algorithm succeeds. Unfortunately if the evaluation point is unlucky, our algorithm will have to lift the parameter t_i to the degree T_i before realizing it. This happens with low probability. Instead of using the bad bound T_i, we start the algorithm with a heuristic bound T for the degree

of the parameters. Now Hensel lifting fails if either the evaluation point is unlucky or the heuristic bound T is not big enough. In this case, we will double the heuristic bound, i.e. $T := 2 \times T$, and restart the algorithm by choosing a new evaluation point. This way, we will eventually get a good evaluation point and a big enough bound T and Hensel lifting will eventually succeed.

In our implementation we choose the initial bound T based on the following conjecture from Abbott [**?**]:

Conjecture 1 (Abbott [?])

$$\deg_{t_i}(\tilde{f}_i) \leq \deg_{t_i}(\tilde{f}) + \sum_{j=1}^{r} \deg_{t_i}(\check{m}_j).$$

Numerical Bound

If one uses Hensel lifting modulo a power of a prime, one also needs a numerical bound B on the size of the integer coefficients in the factors of $\tilde{f}$. Abbott in [**?**] presents a bound. Most algorithms that use Hensel lifting (See [**?**, **?**]) either avoid working modulo a power of a prime or work with a very huge modulus (based on the numerical bound). Both cases result in Hensel lifting having a very poor performance. The following is an example from [**?**].

Example 9 *Let* $\check{m} = z^2 - 4t - 1$ *and* $\tilde{f} = x^2 + x - t = (x + \frac{1+z}{2})(x + \frac{1-z}{2})$. *The bound given by Abbott for factoring* $\tilde{f}$ *is greater than* 5000000.

We avoid both these problems by working modulo a prime p of a modest size and then lift the integer coefficients using our sparse p-adic lifting algorithm if necessary.

We still need a bound B for the case where α is unlucky and Hensel lifting has not detected this due to the unlucky choice of the prime p (See Example 7). For this, we choose $p > B'$ for some B' of a modest size. Now if the sparse p-adic lifting fails, either α is unlucky or the bound B' is not big enough. In this case, we square the bound, i.e. $B' := B'^2$, and restart the algorithm by choosing a new evaluation point α. This way, we will eventually get a good evaluation point and a bound big enough to lift the integer coefficients.

The case that both the evaluation point α and the prime p are unlucky happens with very low probability if they have been chosen at random from a large set of candidates.

4. THE ALGORITHM

Algorithm efactor

Input: $f \in L[x_1, x_2, \ldots, x_v]$ where L is the algebraic function field.

Output: Factorization of f: $f = l \times f_1^{e_1} \times \cdots \times f_n^{e_n}$ where f_i is a monic irreducible polynomial and $l = \mathrm{lc}_{x_1,\ldots,x_v}(f)$.

1: Let $c = \mathrm{cont}_{x_1}(f)$. If $c \neq 1$ then factor c and f/c separately using Algorithm efactor and return the combined result.
2: Do a square-free factorization of f. Call algorithm 1 on each square-free factor and return the result.

Algorithm 1: Main algorithm

Input: $f \in L[x_1, x_2, \ldots, x_v]$ where $\mathrm{cont}_{x_1}(f) = 1$ and f is square-free.

Output: Factorization of f: $f = l \times f_1 \times f_2 \times \cdots \times f_n$ where f_i is monic and $l = \mathrm{lc}_{x_1,\ldots,x_v}(f)$.

1: Compute $\tilde{f}$ (See Definition 1).
2: Compute $\mathbb{D}$, an integral multiple of the defect of the algebraic field L (*See Theorem 2*).
3: **if** $v = 1$ (*univariate case*) **then**
4: Call algorithm 3 on $\tilde{f}$ and $\mathbb{D}$ and return the result.
5: **end if**
6: Let $\bar{l} = \mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f}) \in \mathbb{Z}[t_1, \ldots, t_k]$.
7: Choose a bound B of a modest size (*Heuristic numerical bound*).
8: Let $T = \max_{i=1}^{k} (\deg_{t_i}(\tilde{f}) + \sum_{j=1}^{r} \deg_{t_i} \check{m}_j)$ (*Heuristic bound on the degree of* $\tilde{f}$ *in any parameter. See Conjecture 1*).
9: Factor $\mathrm{lc}_{x_1}(\tilde{f}) \in L[x_2, \ldots, x_v]$ by calling algorithm efactor. Let $\mathrm{lc}_{x_1}(\tilde{f}) = \gamma \times l_1^{e_1} \times l_2^{e_2} \times \cdots \times l_m^{e_m}$ where l_i is monic.
10: Compute $\tilde{l}_i$. Find $\bar{\gamma}, \bar{D} \in \mathbb{Z}[t_1, \ldots, t_k]$ s.t. $\bar{D} \times \mathrm{lc}_{x_1}(\tilde{f}) = \bar{\gamma} \times \prod_{i=1}^{m} \mathrm{lc}_{x_2,\ldots,x_v}(\tilde{l}_i)$. Update $\tilde{f} := \bar{D} \times \tilde{f}$. (*Note* $\bar{D} \mid \mathbb{D}^c$ *for some* $c \in \mathbb{Z}^+$).
11: **Main Loop:** Choose a new good evaluation point $\alpha = (t_1 = \alpha_1, t_2 = \alpha_2, \ldots, t_k = \alpha_k, x_2 = \beta_2, \ldots, x_v = \beta_v)$ that satisfies the requirements in Section 3.1.
12: Let $D_i = \mathrm{den}(\tilde{l}_i(\alpha)^{-1})$. If $\exists i, j : i \neq j, D_i = D_j$ then shift the variables $x_2, \ldots, x_v$ in $\tilde{f}$ and $\tilde{l}_1, \ldots, \tilde{l}_m$ and go to step 11 (*See Example 6*).
13: Using Trager's algorithm factor $\tilde{f}(\alpha)$ to obtain $\tilde{f}(\alpha) = \Omega' \times u_1 \times \cdots \times u_n$ where $\Omega' = \mathrm{lc}_{x_1}(\tilde{f})(\alpha) \in \mathbb{Q}[z_1, \ldots, z_r]$. If $n = 1$ then return $l \times \mathrm{monic}(\tilde{f})$ ($\tilde{f}$ *is irreducible*)
14: Using algorithm 5 on inputs $\{u_1, \ldots, u_n\}$, $\mathrm{lc}_{x_1}(\tilde{f}) = \bar{\gamma} \times \tilde{l}_1^{e_1} \times \tilde{l}_2^{e_2} \times \cdots \times \tilde{l}_m^{e_m}$, the evaluation point α, $\mathbb{D}$ and $\{D_1, \ldots, D_m\}$ compute the true leading coefficients of each univariate factor $\{\bar{l}_1, \bar{l}_2, \ldots, \bar{l}_n\}$. If this fails, go to step 11. Note that $\tilde{f}$ may be updated in order to distribute the integer content of $\mathbb{D} \times \mathrm{lc}_{x_1}(\tilde{f})$.
15: Compute $\delta, \hat{l} \in \mathbb{Z}[t_1, \ldots, t_k]$ s.t. $\delta \times \mathrm{lc}_{x_1,\ldots,x_v}(\tilde{f}) = \hat{l} \times \prod_{i=1}^{n} \mathrm{lc}_{x_2,\ldots,x_v}(\bar{l}_i)$. ($\delta \mid \mathbb{D}^c$ *for some* $c \in \mathbb{Z}^+$ *and* $\hat{l}$ *is a factor of* $lc_{x_1,\ldots,x_v}(\tilde{f})$ *that is not in* $l_1, \ldots, l_n$).
16: Set $\tilde{f} := \delta \tilde{f}$. At this point we have

$$\tilde{f}(\alpha) = \hat{l}(\alpha) \times (\bar{l}_1(\alpha) u_1) \times \cdots \times (\bar{l}_n(\alpha) u_n).$$

17: Choose a prime p s.t. $p > 2B$ satisfying $\mathrm{lc}_{x_1}(\tilde{f}(\alpha)) \bmod p \neq 0$ and $\mathrm{lc}_{z_i}(\check{m}_i(\alpha)) \bmod p \neq 0$.
18: Using algebraic Hensel lifting on inputs $\tilde{f}, \hat{l}$, the set of univariate images $\{u_1, \ldots, u_n\}$, the set of corresponding true leading coefficients $\{\bar{l}_1, \bar{l}_2, \ldots, \bar{l}_n\}$, the prime p, the bound T and the evaluation point α, lift the variables $x_2, x_3, \ldots, x_v$ and the parameters $t_1, \ldots, t_k$ to obtain $\tilde{f} = \hat{l} \times \bar{f}_1 \times \bar{f}_2 \times \cdots \times \bar{f}_n \bmod p$.
19: If Hensel lifting fails then Set $T := 2 \times T$ and go to Step 11 (Main loop).
20: Call algorithm 2 on inputs $\tilde{f} = \hat{l} \times \bar{f}_1 \times \bar{f}_2 \times \cdots \times \bar{f}_n \bmod p$, B and $\{l_1, l_2, \ldots, l_n\}$. If this fails, set $B := B^2$ and go to step 11 otherwise let $f'_1, f'_2, \ldots, f'_n$ be the output s.t. $\tilde{f} = \hat{l} \times f'_1 \times \cdots \times f'_n$.
21: If the variables $x_2, \ldots, x_v$ were shifted in Step 12, shift them back in $f'_1, \ldots, f'_n$.
22: **return** $\mathrm{lc}_{x_1,\ldots,x_v}(f) \times \mathrm{monic}(f'_1) \times \cdots \times \mathrm{monic}(f'_n)$

Algorithm 2: Sparse p-adic lifting

Input: $\tilde{f}, \tilde{f}_1, \ldots, \tilde{f}_n \in L[x_1, \ldots, x_v]$ s.t. $\tilde{f} - \tilde{f}_1 \times \tilde{f}_2 \times \cdots \times \tilde{f}_n \equiv 0 \bmod p$. The numerical bound B and $\{l_1, \ldots, l_n\}$

the set of the leading coefficients of the factors.

Output: Either FAIL, if the evaluation point is unlucky or $h_1, h_2, \ldots, h_n$ s.t. $\tilde{f} = h_1 \times h_2 \times \cdots \times h_n$.

1: Let h_i be $\tilde{f}_i$ with its leading coefficient replaced by l_i.
2: Let $e = \tilde{f} - h_1 \times \cdots \times h_n$. ($\deg_{x_1}(e) < \deg_{x_1}(\tilde{f})$)
3: Let $P = p$.
4: Suppose $\tilde{f}_i = \sum_{j=1}^{T_i} a_{ij} M_{ij}$ with $a_{ij} \in \mathbb{Z}_p$ and M_{ij} monomials.
5: Let $\sigma_i = \sum_{j=1}^{T_i} A_{ij} M_{ij}$ where A_{ij} is an unknown coefficient.
6: **while** $e \neq 0$ and $P < 2B$ **do**
7: $e' = e/P$ (*exact division*)
8: Let $p_z = e' - \sum_{i=1}^{n} \sigma_i \frac{\prod_{j=1}^{n} h_j}{h_i}$.
9: Solve for A_{ij}s by collecting and equating coefficients of p_z in $x_1, \ldots, x_v$, $t_1, \ldots, t_k$ and $z_1, \ldots, z_r$ to zero modulo P.
10: If the system of linear equations is inconsistent then return FAIL. (*Missing term in the form due to the choice of the modulus*)
11: Update $h_i := h_i + \sigma_i \times P$.
12: Set $P := P^2$
13: Set $e = \tilde{f} - h_1 \times \cdots \times h_n$.
14: **end while**
15: If $e = 0$ then return $h_1, h_2, \ldots, h_n$ else return FAIL.

Algorithm 3: Univariate factorization

Input: Square-free $f \in L[x_1]$ and $\mathbb{D}$ the defect of L.

Output: Unique factorization of $f = \mathrm{lc}_{x_1}(f) \times f_1 \times f_2 \times \cdots \times f_n$ over L s.t. f_i is monic in x_1.

1: Compute $\tilde{f}$ (See Definition 1) and Let $\bar{l} = \mathrm{lc}_{x_1}(\tilde{f})$.
2: Choose the bound B of a modest size (*Heuristic numerical bound*).
3: Let $T = \max_{i=1}^{k} (\deg_{t_i}(\tilde{f}) + \sum_{j=1}^{r} \deg_{t_i} \tilde{m}_j)$ (*Heuristic bound on the degree of $\tilde{f}$ in any parameter. See Conjecture 1*).
4: Factor $\gamma = \mathbb{D} \times \bar{l} \in \mathbb{Z}[t_1, \ldots, t_k]$ over $\mathbb{Z}$ to obtain $\gamma = \Omega \times \beta_1^{c_1} \times \cdots \times \beta_{k'}^{c_{k'}}$.
5: **Main Loop:** Choose a new good evaluation point. $\alpha = (t_1 = \alpha_1, \ldots, t_k = \alpha_k)$ that satisfies the requirements in Section 3.1.
6: Using Trager's algorithm, factor $h = \tilde{f}(\alpha) = \bar{l}(\alpha) \times h_1 \times h_2 \times \cdots \times h_n$ over the algebraic number field. Note that $\mathrm{lc}_{x_1}(h_i) = 1$.
7: Compute $\tilde{h}_i$ and let $\bar{d}_i = \mathrm{lc}_{x_1}(h_i) \in \mathbb{Z}$. Find the biggest e_{ij} s.t. $\beta_i^{e_{ij}} \mid \bar{d}_j$. Let $l_i = \beta_1^{e_{1i}} \times \cdots \times \beta_{k'}^{e_{k'i}}$. Distribute $\Omega \in \mathbb{Z}$ to l_i's and if needed, update $\tilde{f}$ and $\tilde{h}_i$. At this point we have $l_i = \mathrm{lc}_{x_1}(\tilde{f}_i)$.
8: Compute $\delta, \hat{l} \in \mathbb{Z}[t_1, \ldots, t_k]$ s.t. $\delta \times \bar{l} = \hat{l} \times \prod_{i=1}^{n} l_i$. ($\delta \mid \mathbb{D}^c$ *for some* $c \in \mathbb{Z}$ *and* $\hat{l}$ *is a factor of* $\mathrm{lc}_{x_1}(\tilde{f})$ *that is not in* $l_1, \ldots, l_n$)
9: Let $\hat{f} = \delta \tilde{f}$ ($\hat{f}(\alpha) = \hat{l}(\alpha) \times \tilde{h}_1 \times \tilde{h}_2 \times \cdots \times \tilde{h}_n$).
10: Choose a prime p s.t. $p > 2B$. satisfying $\deg_{x_1}(\tilde{f} \bmod p) = \deg_{x_1}(\tilde{f})$ and $\mathrm{lc}_{z_i}(\tilde{m}_i(\alpha)) \bmod p \neq 0$.
11: Lift the parameters $\{t_1, \ldots, t_k\}$ in $\hat{f}(\alpha) - \hat{l} \times \tilde{h}_1 \times \tilde{h}_2 \times \cdots \times \tilde{h}_n \equiv 0 \bmod p$ using Hensel lifting with $l_i \in \mathbb{Z}[t_1, \ldots, t_k]$ as the true leading coefficient of $\tilde{h}_i$ and T as the degree bound. If this fails, set $T := 2 \times T$ and go to step 5 (*unlucky evaluation point*).
12: Call algorithm 2 on inputs $\hat{f} = \hat{l} \times \tilde{h}_1 \times \tilde{h}_2 \times \cdots \times \tilde{h}_n \bmod p$, $\{l_1, \ldots, l_n\}$ and B. If this fails, set $B := B^2$ and go to step 5 (main loop) otherwise let $f_1', f_2', \ldots, f_n'$ be the output s.t. $\hat{f} = \hat{l} \times f_1' \times \cdots \times f_n'$.
13: **return** $\mathrm{lc}_{x_1}(f) \times \mathrm{monic}(f_1') \times \cdots \times \mathrm{monic}(f_n')$.

Algorithm 4: Distinct prime divisors

Input: A set $\{a_1, a_2, \ldots, a_n\}$ where $a_i \in \mathbb{Z}$.

Output: Either FAIL or a set of divisors $\{d_1, d_2, \ldots, d_n\}$ s.t. $d_i \neq 1$ and $d_i \mid a_i$ and $\forall j \neq i : \gcd(d_i, d_j) = 1$.

for i from 1 to n **do**
  Let $d_i = a_i$.
  for j from 1 to $i - 1$ **do**
    Let $g = \gcd(d_i, d_j)$.
    Set $d_i := d_i/g$ and $d_j := d_j/g$.
    Let $g_1 = \gcd(g, d_i)$ and $g_2 = \gcd(g, d_j)$. (*Either $g_1 = 1$ or $g_2 = 1$*)
    while $g_1 \neq 1$ **do**
      Let $g_1 = \gcd(d_i, g_1)$.
      Set $d_i := d_i/g_1$.
    end while
    while $g_2 \neq 1$ **do**
      Let $g_2 = \gcd(d_j, g_2)$.
      Set $d_j := d_j/g_2$.
    end while
    if $d_i = 1$ or $d_j = 1$ **then**
      return FAIL.
    end if
  end for
end for
return $\{d_1, \ldots, d_n\}$.

Algorithm 5: Distributing leading coefficients

Input: $\tilde{f}$ and $U = \{u_1, u_2, \ldots, u_n\}$, the set of monic univariate factors where $u_i \in L(\alpha)[x_1]$. $l = \gamma \times l_1^{e_1} \times l_2^{e_2} \times \cdots \times l_m^{e_m}$ the *non-monic* factorization of $l = \mathrm{lc}_{x_1}(\tilde{f})$ where $\gamma \in \mathbb{Z}[t_1, \ldots, t_k]$. The evaluation point α and $\mathbb{D}$ the defect of the algebraic field. $\{D_1, \ldots, D_m\}$ where $D_i = \mathrm{den}(l_i(\alpha)^{-1})$.

Output: Either FAIL, if the leading coefficient is unlucky or $\{\hat{l}_1, \hat{l}_2, \ldots, \hat{l}_n\}$ where $\hat{l}_i \in L[x_2, \ldots, x_v]$ is the true leading coefficient of u_i in x_1 together with possibly updated $\tilde{f}$.

1: Let $\beta = \mathbb{D} \times \gamma = \Omega \times \beta_1^{c_1} \times \beta_2^{c_2} \times \cdots \times \beta_{k'}^{c_{k'}}$ where $\Omega \in \mathbb{Z}$.
2: Let $d_i = \mathrm{den}(u_i)$ and $\mu_i = \beta_i(\alpha)$.
3: Let $\{p_1, \ldots, p_m, q_1, \ldots, q_{k'}\}$ be the output of algorithm 4 on input $\{D_1, \ldots, D_m, \mu_1, \ldots, \mu_{k'}\}$. If this fails, return FAIL.
4: For all $1 \leq i \leq m$, let $g_i = \gcd(\Omega, p_i)$ and Set $p_i := p_i/g_i$. If $p_i = 1$ then return FAIL.
5: For all $1 \leq i \leq k'$, let $g_i' = \gcd(\Omega, q_i)$ and Set $q_i := q_i/g_i'$. If $q_i = 1$ then return FAIL.
6: **for** each d_j **do**
7: **for** i from 1 to m **do**
8: Let $g_1 = \gcd(d_j, p_i)$.
9: Set $e'_{ji} = 0$.
10: **while** $g_1 \neq 1$ **do**
11: Set $e'_{ji} := e'_{ji} + 1$.
12: Set $d_j = d_j/g_1$.
13: Set $g_1 = \gcd(d_j, g_1)$.
14: **end while**
15: **end for**
16: **for** i from 1 to k' **do**
17: Let $g_2 = \gcd(d_j, q_i)$.
18: Set $c'_{ji} = 0$.

19: **while** $g_2 \neq 1$ **do**
20: Set $c'_{ji} := c'_{ji} + 1$.
21: Set $d_j = d_j/g_2$.
22: Set $g_2 = \gcd(d_j, g_2)$.
23: **end while**
24: **end for**
25: **end for**
26: **for** i from 1 to m **do**
27: if $\sum_{j=1}^{n} e'_{ji} \neq e_i$ then return FAIL.
28: **end for**
29: Let $\hat{l}_i = \beta_1^{c_{i1}} \beta_2^{c_{i2}} \dots \beta_{k'}^{c_{ik'}} l_1^{e_{i1}} l_2^{e_{i2}} \dots l_m^{e_{im}}$. Distribute $\Omega \in \mathbb{Z}$ to $\hat{l}_i$s and if needed update $\tilde{f}$.
30: **return** $\{\hat{l}_1, \hat{l}_2, \dots, \hat{l}_n\}$.

Remark 6 In our implementation of algorithm 1, we first choose an evaluation point and compute a univariate factorization then factor $\mathrm{lc}_{x_1}(\tilde{f})$. This is because if $\tilde{f}$ is irreducible, then we do not bother factoring the leading coefficient which might be a big polynomial.

Description of Algorithm 2: In algorithm 2, we have

$$\tilde{f} - \tilde{f}_1 \times \tilde{f}_2 \times \cdots \times \tilde{f}_n \equiv 0 \bmod p.$$

Let $e_1 = \tilde{f} - \tilde{f}_1 \times \cdots \times \tilde{f}_n$. We know that $p \mid e_1$. If $e_1 = 0$ then we are done. We want to find polynomials $\sigma_1, \dots, \sigma_n$ s.t.

$$\tilde{f} - (\tilde{f}_1 + \sigma_1 p)(\tilde{f}_2 + \sigma_2 p) \dots (\tilde{f}_n + \sigma_n p) \equiv 0 \bmod p^2.$$

Expanding the above expression results in $g \equiv 0 \bmod p$ where

$$g = \sigma_1 \tilde{f}_2 \tilde{f}_3 \dots \tilde{f}_n + \cdots + \sigma_n \tilde{f}_1 \tilde{f}_2 \dots \tilde{f}_{n-1} - \frac{e}{p}.$$

We assume that the terms in σ_i are the same as the terms in $\tilde{f}_i$ with the integer coefficient replaced by an unknown. We compute the polynomial g and equate each coefficient in $z_1, \dots, z_r, t_1, \dots, t_k, x_1, \dots, x_v$ to zero. This gives us a linear system which will not be underdetermined because we already know the exact leading coefficient in the main variable of each factor $\tilde{f}_i$ and the uniqueness is guaranteed by Hensel lemma. After solving this system we will obtain the correction polynomials $\sigma_1, \dots, \sigma_n$. We update each factor $\tilde{f}_i := \tilde{f}_i + \sigma_i p$. Now we have

$$\tilde{f} - \tilde{f}_1 \times \tilde{f}_2 \times \cdots \times \tilde{f}_n \equiv 0 \bmod p^2.$$

We repeat this non-linear lifting algorithm until $p^{2^k} > 2|B|$ where B is the biggest integer coefficient in the factors of $\tilde{f}$. Thus if there are no extraneous factors and no missing terms caused by the choice of primes and evaluation points, the algorithm will not depend on a bound on the size of the coefficients in the factor of $\tilde{f}$ which could be big.

Remark 7 In Step 8 of algorithm 3 and Step 15 of algorithm 1, we compute $\hat{l} \in \mathbb{Z}[t_1, \dots, t_k]$ which is the factor of the leading coefficient of $\tilde{f}$ in all the variables which does not show up in the leading coefficient of any factors of $\tilde{f}$.

Example 10 *Let $m = z^2 - \frac{1}{t}$ and $f = x^2 - \frac{1}{t}$. We have $\breve{m} = tz^2 - 1$ and $\tilde{f} = tx^2 - 1$. The factorization of $\tilde{f}$ is*

$$\tilde{f} = t(x - z)(x + z).$$

Here $l_1 = l_2 = 1$ and $\hat{l} = t$. We have $\hat{l} \mid lc_x(\tilde{f})$ but $\hat{l} \nmid l_i$.

A Failed Experiment

The bottleneck of Hensel lifting algorithm is solving the Diophantine equations. One can solve these Diophantine equations using sparse interpolation with a similar technique as in algorithm 2. Here is an example.

Example 11 *Let $\breve{m} = z^2 - (t-1)^3$ and*

$$\tilde{f} = \left(t^3 - t - t^2 + 1\right) x^2 - x\left(2\,t + 1\right) z - t^4 + t^2.$$

Suppose we choose the evaluation point to be $t = 4$. We compute the univariate factors using Trager's algorithm and after computing and attaching the leading coefficients of the factors we have

$$\begin{aligned} \hat{f} &= (t-1)^2 \tilde{f}, \\ \tilde{f}_1 &= \left(t^3 - t - t^2 + 1\right) x + 16\, z, \\ \tilde{f}_2 &= \left(t^2 - 2\,t + 1\right) x - 5\, z, \end{aligned}$$

where $\hat{f} - \tilde{f}_1 \tilde{f}_2 \equiv 0 \bmod (t-4)$. Now we start Hensel lifting. The first error polynomial is $e_1 = \hat{f} - \tilde{f}_1 \tilde{f}_2$. We have

$$\frac{e_1}{t-4} = \left(3\,t^2 z - 6\,tz + 3\,z\right) x - t^5 - 2\,t^4 - 8\,t^3 + 46\,t^2 - 55\,t + 20.$$

Now we need to find two polynomials σ_1 and σ_2 s.t.

$$\sigma_2 \tilde{f}_1 + \sigma_1 \tilde{f}_2 - \frac{e_1}{t-4} \equiv 0 \bmod (t-4). \tag{1}$$

Similar to algorithm 2, we can assume that σ_1 and σ_2 have the same monomials as $\tilde{f}_1$ and $\tilde{f}_2$ respectively and since we know that the leading coefficient of $\tilde{f}_1$ and $\tilde{f}_2$ are correct, the forms for σ_1 and σ_2 are $\sigma_1 = Az$ and $\sigma_2 = Bz$. Using these forms and Equation 1 we construct and solve a linear system to obtain $A = 8, B = -1$. We update $\tilde{f}_1 := \tilde{f}_1 + \sigma_1 \times (t-4)$ and $\tilde{f}_2 := \tilde{f}_2 + \sigma_2 \times (t-4)$ to get

$$\begin{aligned} \tilde{f}_1 &= (t^3 - t^2 - t + 1)x + 16z + 8(t-4)z, \\ \tilde{f}_2 &= (t^2 - 2t + 1)x - (t-4)z - 5z. \end{aligned}$$

This time the new error polynomial is $e_2 = \hat{f} - \tilde{f}_1 \tilde{f}_2$ and we have

$$\frac{e_2}{(t-4)^2} = \left(t^2 z - 2\,tz + z\right) x - t^4 + 2\,t^3 - 2\,t + 1$$

and

$$\sigma_2 \tilde{f}_1 + \sigma_1 \tilde{f}_2 - \frac{e_2}{(t-4)^2} \equiv 0 \bmod (t-4)^2. \tag{2}$$

The new assumed forms are

$$\begin{aligned} \sigma_1 &= Az + Bz(t-4), \\ \sigma_2 &= Czt + Dz. \end{aligned}$$

Again we construct a system of linear equations using Equation 2 and after solving this system we have $A = 1, B = 0, C = 0, D = 0$. We update $\tilde{f}_1$ and $\tilde{f}_2$ and to obtain

$$\begin{aligned} \tilde{f}_1 &= \left(t^3 - t^2 - t + 1\right) x + zt^2, \\ \tilde{f}_2 &= \left(t^2 - 2\,t + 1\right) x - zt - z. \end{aligned}$$

The new error polynomial $e_3 = \hat{f} - \tilde{f}_1 \tilde{f}_2$ is zero so $\tilde{f} = lc_x(\tilde{f}) \times monic(\tilde{f}_1) \times monic(\tilde{f}_2)$ and we are done.

We do not use this method in our new algorithm for lifting parameters and variables. This is because it is slower than solving the Diophantine equations using the traditional method for two main reasons:

1. The systems of linear equations in each step can be very big if the factors are dense.

   Example 12 *Suppose $\tilde{f}_1$, $\tilde{f}_2$ and $\tilde{f}_1 \times \tilde{f}_2$ have N_1, N_2 and N terms respectively. Then the system of linear equation has N equations and as many as $N_1 - 1 + N_2 - 1$ unknowns.*

2. As Hensel lifting algorithm progresses, usually, the error term gets smaller and smaller, so solving the Diophantine equation is usually easier at the next step. But using sparse interpolation, as the Hensel lifting algorithm proceeds, each factor $\tilde{f}_i$ usually gets bigger and bigger, because we add new terms, so the system of linear equations gets bigger which means Hensel lifting will be slower.

We do not have the second problem above for sparse interpolation in algorithm 2, when we lift integer coefficients, mainly because the forms of the σ polynomials do not change due to the fact that only integer coefficients of factors of $\tilde{f}$ are being updated.

5. BENCHMARKS

We have compared Maple 13 implementations of our new algorithm (efactor) with Maple's implementation of Trager's algorithm modified to use SparseModGcd (See [**?**]) for computing GCDs over L on the eight problems in the Appendix.

The timings are given in Table 1. All timings are in CPU seconds and were obtained on Maple 13 on a 64 bit AMD Opteron CPU @ 2.4 GHz, running Linux. In the table, n is the number of variables, r is the number of field extensions, k is the number of parameters, d is the total degree of f, $\#f$ is the number of terms in f and $\#\tilde{f}$ is the number of terms in $\tilde{f}$. In all the following problems, f factors into two irreducible factors f_1 and f_2.

For each problem we used $p = 3037000453$, a 31.5 bit prime, for Hensel lifting. For problems 3,4,5 and 7, p is big enough so that there is no need to lift the integer coefficients using sparse p-adic lifting algorithm. However, for problems 1,2,6 and 8, p is not big enough. In problems 1,2 and 6, the number of lifting steps is one, i.e., $p^2 > 2||f_i||$. For the last problem, the number of lifting steps is three, i.e. $p^8 > 2||f_i||$.

The last column in Table 1 is the time for computing

$$\gcd(f_1 f_2, f_1(f_2 + 1))$$

using our SparseModGcd algorithm in [**?**]. One can see that our factorization algorithm is often as fast as the GCD algorithm on a problem of comparable size, except for problem 6. In problem 6, almost all (99%) of the time was factoring the univariate polynomial over $\mathbb{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \sqrt{7}, \sqrt{11})$ using Trager's algorithm.

For the sixth problem, we have multiplied the polynomial f from Section 1 by one of its conjugates. Table 1 illustrates that Trager's algorithm did not finish in a reasonable time. In fact Maple did not succeed to compute even the norm of the input polynomial after 50000 seconds.

The percentages of timings for different parts of our new algorithm for these problems are presented in Table 2. In this table, the second column is the percentage of time spent on univariate factorization over $L(\alpha)$ using Trager's algorithm. The numbers in the third column correspond to the time spent on lifting variables and integer coefficients respectively. And finally, numbers in the last column are the percentages of time spent on doing square-free factorizations of the inputs.

#	$(n, r, k, d, \#f, \#\tilde{f})$	Trager	efactor	GCD
1	(2,2,1,17,191,6408)	5500	259.91	47.47
2	(2,2,1,22,228,12008)	37800	296.74	56.90
3	(2,2,2,10,34,34)	120	0.22	0.16
4	(2,2,2,12,34,34)	571	0.31	0.19
5	(3,2,2,10,69,69)	5953	0.27	0.29
6	(6,5,0,4,46,46)	> 50000	88.43	1.93
7	(5,2,1,10,15489,17052)	> 50000	58.41	57.75
8	(1,1,2,102,426,928)	16427	72.10	7.71

Table 1: Timings (in CPU seconds)

#	Univariate	Lifting	Sqr-free
1	0.30%	(4.99%,90.1%)	4.01%
2	0.80%	(7.82%,84.42%)	6.45%
3	51.61%	(17.05%,0%)	19.35%
4	57.23%	(22.03%,0%)	12.50%
5	42.86%	(35.53%,0%)	19.41%
6	99.47%	(0.31%,0.52%)	0.14%
7	0.80%	(28,289%,0%)	67.41%
8	2.06 %	(91.68%,5.47%)	0.70 %

Table 2: Timing (percentile) for different parts of efactor

Remark 8 The bottleneck of our new algorithm for the first two problems is the sparse p-adic lifting algorithm. This is because of the large number of terms in $\tilde{f}$.

6. CONCLUDING REMARKS

We gave a new algorithm for factorization over an algebraic function field. We generalized Wang's algorithm for leading coefficient predetermination for the algebraic field L. We gave a sparse p-adic lifting algorithm for lifting the integer coefficient of the factors of the input polynomial. Here we list some of the inefficiencies of our algorithm.

1. One of the bottlenecks of the sparse p-adic lifting algorithm is multiplying the polynomial $\tilde{f}_i$ by the forms of σ polynomials, i.e.

   $$\tilde{f}_i \times \sigma_1 \times \cdots \times \sigma_{i-1} \times \sigma_{i+1} \times \ldots \sigma_n,$$

   and reducing it modulo the minimal polynomials. This is expensive when factors of $\tilde{f}$ are dense. Since all the coefficients in σ polynomials are distinct unknown variables, one may be able to devise a more efficient multiplication algorithm for this task.

2. We use Trager's algorithm for univariate factorization of $\tilde{f}(\alpha)$ over the algebraic number field $L(\alpha)$. As illustrated in problem six of Section 5, this algorithm can be expensive when there are several extensions and $\tilde{f}$ has several variables, resulting in big coefficients in $\text{norm}(\tilde{f}(\alpha))$. To solve this problem, one could use a modular algorithm similar to the algorithm for univariate factorization over $\mathbb{Z}$.

3. We need to choose the evaluation point α in algorithms 1 and 3 at random from a large set of points to avoid

any missing terms in the factors of $\tilde{f}$ and also to get a set of distinct prime divisors for the inverses of the factors of the leading coefficient. This results in large integer coefficients in $\tilde{f}(\alpha)$ which makes both Trager's algorithm and Hensel lifting slower. If there are any missing terms due to the choice of α (or the prime p), the system of linear equations in Step 10 of algorithm 2 will not have a unique solution with high probability.

Appendix

Here we give the polynomials (in factored form) for the eight problems presented in Section 5. Here f_1 and f_2 are the factors and $f = f_1 \times f_2$. Problems 3-5 are from [**?**]. Problems 1 and 2 have large leading coefficients in the main variable x. Problem 6 is from [**?**]. Problems 7 and 8 have large number of variables and large integer coefficients respectively. In all problems s, t are parameters and u, v, w, x, y are polynomial variables.

1. Problem 1:

$$f_1 = 63x^2yt + 16x^2t^2 + 7xz_1{}^3 - 43y^2t^2 - 34yz_1{}^2z_2 -$$
$$20xyz_1{}^2z_2 - 35y^2z_2x^2z_1 + 29y^2x^2t^3z_2 - 27y^2x^2tz_1{}^3 +$$
$$78y^2x^2tz_1z_2{}^3 + 25y^2x^2tz_2{}^4 + 30y^2x^2z_2{}^5,$$
$$f_2 = -27x - 99yz_1 - 81xy^2t - 42t^3z_2 + 30xyz_1{}^3 -$$
$$21yz_1{}^4 - 85y^2z_2x + 50y^2xz_1{}^2z_2{}^2 - 55y^2xz_2{}^4 -$$
$$64y^2xz_1t^2z_2{}^2 - 75y^2xtz_1z_2{}^3 + 90y^2xz_1{}^3z_2{}^2$$

2. Problem 2:

$$f_1 = -51z_1{}^2 + 77xz_1z_2{}^2 + 95x^4z_2 + x^3z_1z_2 +$$
$$55xtz_2{}^3 + 53x^4y^2z_1z_2 - 28x^4y^2z_2{}^2 + 5x^4y^2t^2z_1 +$$
$$xy^2tz_2 + 13x^4y^2tz_1z_2 - 10x^4y^2t^2z_1{}^2z_2 - 82x^4y^2z_1{}^4z_2,$$
$$f_2 = -15xy^3 - 59xy^2t - 96t^2z_1z_2 + 72x^4z_1 - 87ytz_2{}^3 +$$
$$98x^4y^3z_1{}^3 - 48x^4y^3t^3z_1z_2 - 19x^4y^3t^2z_1{}^2z_2 +$$
$$47t^2z_1{}^2z_2 + 62x^4y^3t^2z_2{}^3 + 37x^4y^3z_1{}^4z_2 + 5x^4y^3z_1z_2{}^4$$

3. Problem 3:

$$f_1 = x^2 + \left(3 + z_2z_1y + t^3\right)x + y^2 + z_2{}^2s,$$
$$f_2 = 2tx^2 + \left(-z_2s^3y + z_2{}^2s\right)x + 5z_1t^3 - 3sy^2.$$

4. Problem 4:

$$f_1 = x^3 + y^3 + z_2z_1xy^2 + t^3x^2 + x + z_2{}^2s,$$
$$f_2 = 2tx^3 - 3sy^3 - z_2s^2txy^2 + z_2{}^2sx + 5z_1t^3.$$

5. Problem 5:

$$f_1 = x^2 + (t^3 + 3z_2z_1y + wt)x + 3swz_2 + y^2 + w^2 + z_2{}^2s,$$
$$f_2 = 2tx^2 + (z_2{}^2s - z_2s^3y)x + 5z_1t^3 - 3sy^2 - 2wyz_2z_1 + stw^2.$$

6. Problem 6:

$$f_1 = \frac{19}{2}c_4^2 - \sqrt{11}\sqrt{5}\sqrt{2}c_5c_4 - 2\sqrt{5}c_1c_2 - 6\sqrt{2}c_3c_4 + \frac{3}{2}c_0^2 + \frac{23}{2}c_5^2$$
$$+ \frac{7}{2}c_1^2 - \sqrt{7}\sqrt{3}\sqrt{2}c_3c_2 + \frac{11}{2}c_2^2 - \sqrt{3}\sqrt{2}c_0c_1 + \frac{15}{2}c_3^2 - \frac{10681741}{1985},$$
$$f_2 = \frac{19}{2}c_4^2 - \sqrt{11}\sqrt{5}\sqrt{2}c_5c_4 - 2\sqrt{5}c_1c_2 + 6\sqrt{2}c_3c_4 + \frac{3}{2}c_0^2 + \frac{23}{2}c_5^2$$
$$+ \frac{7}{2}c_1^2 + \sqrt{7}\sqrt{3}\sqrt{2}c_3c_2 + \frac{11}{2}c_2^2 + \sqrt{3}\sqrt{2}c_0c_1 + \frac{15}{2}c_3^2 - \frac{10681741}{1985}$$

7. Problem 7:

```
>f1 := randpoly([x,y,u,v,w,t,z1,z2], terms = 200);
>f2 := randpoly([x,y,u,v,w,t,z1,z2], terms = 200);
```

8. Problem 8:

```
>f1 := s*x^50 + randpoly([x,z,t,s], degree = 50,
coeffs = rand(10^50)) + 1;
>f2 := t*x^50 + randpoly([x,z,t,s], degree = 50,
coeffs = rand(10^50)) + 1;
```

Also the minimal polynomials are

1. Problems 1,2 and 7:

$$m_1 = z_1{}^2 - t,$$
$$m_2 = z_2{}^3 - z_1z_2{}^2 - t^2z_2 + 7.$$

2. Problems 3 to 5:

$$m_1 = z_1{}^2 - 2,$$
$$m_2 = z_2{}^3 + tz_2{}^2 + s.$$

3. Problem 8:

$$m_1 = z^2 - tz - s.$$