ISSN: 2424-8460(O)
2251-2608(P)

Original Research Article

A faster square-free factorization algorithm for multivariable polynomials

Weile Huang*
BASIS International School Guangzhou , Guangzhou, Guangdong, 510700, China

Abstract: This paper provides a faster algorithm for multivariate square-free decomposition, using random primes as coefficients of univariate substituting variables instead of p-adic construction, and returning to the multivariate expression using the unique prime factorization property of integers.
Keywords: multivariate polynomial; square-free decomposition; sparse polynomial

1. Introduction

Square-free decomposition of polynomials is an indispensable device with many uses in polynomial factorization, partial fraction decomposition of rational functions, and rational function integrations. Existing univariate square-free decomposition methods have reduced the cost of computation to finding the GCD of P and P' [1,2]. However, existing multivariate square-free decomposition algorithms commonly rely on Hensel or p-adic construction to convert the problem to a univariate one, which becomes increasingly difficult when the number of variables becomes large. This paper provides a faster algorithm for multivariate square-free decomposition, keeping the idea of substitution to convert multivariate polynomials into univariate ones, but with a faster method that introduces prime coefficients to the substituting variables, and recovering into multivariate expressions using simple integer division.

2. Preliminary knowledge

Definition 2.1. A polynomial is called square-free if it does not contain any factor of multiplicity ≥2 . The general form of a square-free decomposition of polynomial P is $P = f_1 f_2^2 ... f_\ell^\ell$, where each f_i is square-free and gcd(f_i,f_j)=1, $\forall\ i\neq j$

Definition 2.2. We call a polynomial f $\in \mathbb{Z}[x_1,\ldots,x_n,y]$ is separable w.r.t. y if f has the form as $f=c_1M_1y^{d1}+c_2M_2y^{d2}+\ldots+c_tM_ty^{dt}$, where $c_i(\neq 0) \in \mathbb{Z}$, M_i,i=1,…,t are monomials in $\mathbb{Z}[x_1,\ldots,x_n]$ and d_i,i=1,…,t are different exponents.

Definition 2.3. A term is called the monomial content of $f=c_1m_1+c_2m_2+\ldots+c_tm_t \in \mathbb{Z}[X]$, if $c=gcd(c_1,\ldots,c_t)$ and m=gcd(m_1,…,m_t), where mi's are different monomials and ci's are non-zero coefficients. Denote cm as MoCont(f) and we call f/MoCont(f) the monomial primitive part of f and denoted it as MoPrim(f).

Definition 2.4. Let $F_1 = \sum_{i=0}^{d} a_i y^i$ and $F_2 = \sum_{i=0}^{\ell} b_i y^i$ be non-zero. The Sylvester matrix of F_1, F_2, is the d+ℓ by d+ℓ matrix

$$\begin{pmatrix} a_d & a_{d-1} & \ldots & a_1 & a_0 & & \\ & a_d & a_{d-1} & \ldots & a_1 & a_0 & \\ & & \ldots & \ldots & \ldots & \ldots & \\ & & & a_d & \ldots & \ldots & a_0 \\ b_\ell & b_{\ell-1} & \ldots & b_1 & b_0 & & \\ & b_\ell & b_{\ell-1} & \ldots & b_1 & b_0 & \\ & & \ldots & \ldots & \ldots & \ldots & \\ & & & b_\ell & \ldots & \ldots & b_0 \end{pmatrix} \tag{2.1}$$

ISSN: 2424-8460(O)
2251-2608(P)

where the upper part of the matrix consists of ℓ rows of coefficients of F_1, the lower part consists of rows of coefficients of F_2. The resultant of F_1 and F_2 is the determinant of the Sylvester matrix of F_1, F_2, written as $\text{res}_y(F_1,F_2)$.

The following are some facts. Denote $\text{LC}_y(F_1)$ as the leading coefficient of F_1 w.r.t. y.

Lemma 2.5. [4, Lemma 4] Let $F_1,F_2 \in \mathbb{Z}[x_1,\ldots,x_n,y]$ with $d=deg_y(F_1)>0$ and $\ell=deg_yF_2>0$. Let $a_d=LC_y(F_1)$,b$\ell=LC_y(F_2)$,$R=res_y(F_1,F_2)$,$\vec{\alpha} \in \mathbb{Z}^n$. Then

(i) a_d,b_ℓ,R are polynomials in $\mathbb{Z}$[X]

(ii) $res_y(F_1(y, \vec{\alpha}), F_2(y,\vec{\alpha}))=R\ \vec{\alpha})$ and degygcd $(F_1(y, \vec{\alpha}), F_2(y, \vec{\alpha}))>0 \Leftrightarrow$ resy$(F_1(y, \vec{\alpha}),(F_2(y, \vec{\alpha}))=0$;

This theorem will be useful when choosing the coefficients $p_1,\ldots,p_n$ to substitute in the multivariate polynomial $P(x_1y^{s1},\ldots,x_ny^{sn})$, where $s_i \in \mathbb{N}$. [3].

Our proof will make extensive use of the Schwartz-Zippel Lemma.

Lemma 2.6. Let F be a field and $A \in F\ [x_1,\ldots,x_n]$ be non-zero with total degree D and let S $\subset F$ be a finite set. If $\vec{\beta}$ is chosen at random from S^n then $\text{Prob}[A(\vec{\beta}) = 0] \leq \frac{D}{|S|}$.

3. A faster square-free algorithm for multivariable polynomials

3.1. An example for our new algorithm

Consider the multivariate polynomial

$$P = f_1 \cdot f_2^2 \in \mathbb{Z}[x_1, ..., x_n]$$

,

where

$$f_1 = 5x_1^2x_2^4 + 7x_3^6$$

and

$$f_2 = 10x_1x_2x_3^5 + 6x_1 + 3x_2$$

.

It is easy to check that f_1 and f_2 are square-free and

$$\begin{aligned}P = {} & 500x_1^4x_2^6x_3^{10} + 700x_1^2x_2^2x_3^{16} + 600x_1^4x_2^5x_3^5 + 300x_1^3x_2^6x_3^5 + 840x_1^2x_2x_3^{11} \\ & +420x_1x_2^2x_3^{11} + 180x_1^4x_2^4 + 180x_1^3x_2^5 + 45x_1^2x_2^6 + 252x_1^2x_3^6 + 252x_1x_2x_3^6 + 63x_2^2x_3^6.\end{aligned}$$

We want to compute the square-free factorization of P. The main idea involves using the substitution method to simplify polynomial P into a univariate polynomial. Subsequently, we apply univariate square-free factorization to compute its square-free decomposition. Finally, we reconstruct the multivariate factors from the univariate factors obtained.

A simple substitution method, known as the Kronecker substitution, can be utilized here because the polynomial has a maximum degree of 18. Specifically, we can choose the substitution $(x_1, x_2, x_3)=(y, y^{19}, y^{192})$ to transform $P(x_1, x_2, x_3)$ into $P(y, y^{19}, y^{192})$. However, it's important to note that with this type of substitution, as the number of variables increases, the degree of the resulting polynomial may become excessively high, leading to a significant increase in computation time.

Therefore, we propose a new method for choosing the substituting variables: randomly selecting three prime numbers as the coefficients, and randomly selecting three integers in a certain range as the degrees. The range we

ISSN: 2424-8460(O)
2251-2608(P)

propose is [0,9], where $9=3^2$ and 3 is the upper bound of the number of terms in f_1 and f_2. We will later explain why this is chosen as the bound.

Now, randomly choose (7,2,5) and let $(x_1, x_2, x_3)=(y^7, y^2, y^5)$. We reduce P into an univariate polynomial $P(y^7, y^2, y^5)$.

Denote $H_1(y):P(y^7, y^2, y^5)$. Now perform square-free decomposition again on $H_1(y)=P(y^7, y^2, y^5)$, and we get

$$H_1=y^{26}\cdot(7y^8+5)\cdot(10y^{32}+6y^5+3)^2.$$

We know that $7y^8+5$ is corresponding to f_1 and $10y^{32}+6y^5+3$ is corresponding to f_2. We can read all coefficients 7, 5 and 10, 6, 3 of f_1 and f_2 from $7y^8+5$ and $10y^{32}+6y^5+3$. But we can not recover the exponents of f_1 and f_2.

To recover the exponents, we have the following remark.

Remark 3.1. $f(x_1, ..., x_n) = cx_1^{e_1}...x_n^{e_n}$ and $p_1,\ldots,p_n$ be different primes. Let $s_i \in \mathbb{N}$, $i=1,\ldots,n$. Then

$$f(y^{s1},\ldots,y^{sn})=c^{yd}$$

and

$$f(p_1y^{s_1}, ..., p_ny^{s_n}) = cp_1^{e_1}...p_n^{e_n}y^d$$

The only difference between the two univariate polynomials is the coefficients. One is C and the other is $cp_1^{e_1}\cdots p_n^{e_n}$. So we can compute $p_1^{e_1}\cdots p_n^{e_n}$ and C by division, and compute $e_1,\ldots,e_n$ by sequentially dividing $p_1^{e_1}\cdots p_n^{e_n}$ by $p_1,\ldots,p_n$. Thus we recover the term $cx_1^{e_1}\cdots x_n^{e_n}$.

Now we randomly choose $(p_1,p_2,p_3)=(13,17,19)$ and replace it with $H_2(y):=P(13y^7,17y^2,19y^5)$, and perform square-free factorization on H_2 to get

$$H_2=y^{26}\cdot(329321167y8+70575245)\cdot(5472178790y^{32}+78y^5+51)^2.$$

Notice that the degrees of H_1 and H_2 are the same, and only the coefficients differ. To retrieve the degrees of each variable in H_1, H_2, we divide the coefficients of H_1 and H_2 for each corresponding term, and factorize in terms of 13, 17 and 19:

$$329321167/7 = 19^6,$$

corresponding to the term $7x_3^6$ in f_1.

$$70575245/5 = 13^2 \cdot 17^4,$$

corresponding to the term $5x_1^2x_2^4$ in f_1.

$$5472178790/10 = 13 \cdot 17 \cdot 19^5,$$

corresponding to the term $10x_1x_2x_3^5$ in f_2.

$$78/6=13,$$

corresponding to the term $6x_1$ in f_2.

Finally,

$$51/3 = 17,$$

ISSN: 2424-8460(O)
2251-2608(P)

corresponding to the term $3x_2$ in f_2.

Thus, we have the result:

$$P = \left(5x_1^2x_2^4 + 7x_3^6\right) \cdot \left(10x_1x_2x_3^5 + 6x_1 + 3x_2\right)^2,$$

successfully performing a square-free factorization on P.

Now we generalize the method to any multivariate polynomial: suppose a multivariate polynomial $P(x_1,\ldots, x_n) \in \mathbb{Z}[x_1,\ldots,x_n]$. The degree s_i of each substitution variable $(x_1,\ldots, x_n)=(y^{s1},\ldots,y^{sn})$ can then be chosen from a range of integers $[0, T^2]$, where T=terms(f_1)+terms(f_2)+…+terms(f_ℓ) and terms(f_i) denotes the number of terms in square-free polynomial f_i. We can then substitute two sets of variables into $P(x_1,\ldots, x_n)$: one with the prime coefficients, $H_2=P(P_1y^{s1},\ldots,p_ny^{sn})$, and another without the coefficients, $H_1=P(y^{s1},\ldots,y^{sn})$.

After substitution, we would get two univariate polynomials and can perform square-free decomposition on each of them. If successful, the two square-free polynomials would have the same number of terms and degrees for each square-free component, but with different coefficients. We can then divide the corresponding terms in one such component of the two square-free polynomials and get an integer, which can easily factorized as we know its factors only consist of $(p_1,\ldots,p_n)$. Performing the division and factorization on each square-free component, we can retrieve the degree for each variable in the original polynomial $P(x_1,\ldots, x_n)$. Thus, the multivariate square-free decomposition algorithm can be realized without resorting to p-adic construction.

However, there is a possibility that if H_1 and H_2 do not have the same number of terms and degrees for each square-free component, our algorithm will fail. To improve the success rate, we compute $H_3 = P(p_1^2y^{s1}, \ldots, p_n^2y^{sn})$ instead of computing H_1.

We have the following remark.

Remark 3.2. Let $f(x_1, \ldots, x_n) = cx_1^{e1}\ldots x_n^{en}$ and $p_1,\ldots,p_n$ be different primes. Let $s_i \in \mathbb{N}$, i=1,…,n.

Then

$$f(p_1y^{s1}, \ldots, p_ny^{sn}) = cp_1^{e1}\ldots p_n^{en}y^d$$

and

$$f\left(p_1^2y^{s1}, \ldots, p_n^2y^{sn}\right) = cp_1^{2e1}\ldots p_n^{2en}y^d$$

The coefficients in the two univariate polynomials is $cp_1^{e1}\cdots p_n^{en}$ and the other is $cp_1^{2e1}\cdots p_n^{2en}$. We also can compute C from the division $\frac{(cp_1^{e1}\cdots p_n^{en})^2}{cp_1^{2e1}\cdots p_n^{2en}}$ and compute $e_1,\ldots,e_n$ by sequentially dividing $p_1^{e1}\cdots p_n^{en}$ by $p_1,\ldots,p_n$. Thus we recover the term $cx_1^{e1}\cdots x_n^{en}$.

3.2. Algorithm

As the monomial content of P can be computed easily, in the following algorithm, we directly assume P is monomial primitive.

Algorithm 1: Square-free Factorization for Multivariate Polynomials

Input: Monomial primitive polynomial $P \in \mathbb{Z}[x_1,\ldots,x_n]$; a terms bound

$$T \geq \text{terms}(f_1) + \ldots + \text{terms}(f_\ell) \text{ where } P = f_1f_2^2\ldots f_\ell^\ell.$$

Output: $f_1,f_2,\ldots,f_\ell$ with high probability.

1. Randomly choose different $p_1,p_2,\ldots,p_n$ in $(N,2N]$, where N=poly(T,D,n).

2. Randomly choose an integer s_i between $[0,S]$, where $S=T^2$.

ISSN: 2424-8460(O)
2251-2608(P)

3. Evaluate P on two sets of variables:

$$H_2 \leftarrow P(p_1 \cdot y^{s_1}, p_2 \cdot y^{s_2}, ..., p_n \cdot y^{s_n}),$$
$$H_3 \leftarrow P\left(p_1^2 \cdot y^{s_1}, p_2^2 \cdot y^{s_2}, ..., p_n^2 \cdot y^{s_n}\right).$$

4. Perform univariate square-free factorization on H_2 and H_3:

$$H_2 \leftarrow g_1 \cdot g_2^2 \cdot g_3^3 \cdot ... \cdot g_\ell^\ell$$
$$H_3 \leftarrow q_1 \cdot q_2^2 \cdot q_3^3 \cdot ... \cdot q_\ell^\ell$$

5. Let f_i:0,i=1,…,ℓ.

6. for k←1 to 1 do

$$g_k \leftarrow a_1 \cdot y^{d_1} + a_2 \cdot y^{d_2} + ... + a_t \cdot y^{d_t}$$
$$q_k \leftarrow b_1 \cdot y^{d_1} + b_2 \cdot y^{d_2} + ... + b_t \cdot y^{d_t}$$

7. for i←1 to t do

8. Factor a_i and b_i as $cp_1^{e_1}...p_n^{e_n}$ and $c(p_1^{e_1}...p_n^{e_n})^2$

9. Let $f_i \leftarrow f_i + cx_1^{e_1} \cdots x_n^{e_n}$.

10. end

11. end

12. Check if the calculation is correct by expanding the square-free factorization back to the original expression.

13. If $P \neq f_1 f_2^2 \cdots f_\ell^\ell$, then return "Failure". Else Return $f_1, f_2, \ldots, f_\ell$.

Denote r_i:$f_i(x_1y^{s1},\ldots,x_ny^{sn})$ for i=1,…,ℓ. To ensure the correctness of Algorithm 1, we should ensure that g_i and q_i in Step 4 correspond one-to-one to f_i.

It is easy to see that if the following three situations are met, it is sufficient to achieve it.

1. r_i is separate with respect to y;

2. when replacing x_1=p_1,…,x_n=p_n or $x_1 = p_1^2, ..., x_n = p_n^2$ in r_i, r_i remains be square-free up to a trivial factor y^k.

3. when replacing x_1=p_1,…,x_n=p_n or $x_1 = p_1^2, ..., x_n = p_n^2$ in all r_i's, the GCD of r_i, r_i is $y^{ei,j}$ for $i \neq j$ and some $e_{i,j} \in \mathbb{N}$.

3.2.1. Condition 1

In Condition 1, we need to ensure each r_i is separate with respect to y. Let $P = f_1^1 f_2^2 ... f_\ell^\ell$. And without loss of generality, we only consider r_1. Assume

$$f_1 = c_1 x_1^{d_{11}} x_2^{d_{12}} ... x_n^{d_{1n}} + c_2 x_1^{d_{21}} x_2^{d_{22}} ... x_n^{d_{2n}} + ... + c_t x_1^{d_{t1}} x_2^{d_{t2}} ... x_n^{d_{tn}}.$$

After replacement, we have $r_1 = f_1(x_1 y^{s_1}, ..., x_n y^{s_n})$ and

$$r_1 = c_1 x_1^{d_{11}} x_2^{d_{12}} ... x_n^{d_{1n}} y^{s_1 d_{11} + s_2 d_{12} + ... + s_n d_{1n}} + ... + c_t x_1^{d_{t1}} x_2^{d_{t2}} ... x_n^{d_{tn}} y^{s_1 d_{t1} + s_2 d_{t2} + ... + s_n d_{tn}}.$$

Let $D_i = s_1 d_{i1} + s_2 d_{i2} + ... + s_n d_{in}, i = 1, ..., t$. We need to ensure that $\forall\ i \neq \mathbf{j}$, $D_i \neq D_j$. This means $D_i - D_j \neq 0$, which implies

$$s_1(d_{i1} - d_{j1}) + s_2(d_{i2} - d_{j2}) + ... + s_n(d_{in} - d_{jn}) \neq 0.$$

Therefore, we need to choose degrees $\{s_1,\ldots,s_n\}$ such that

ISSN: 2424-8460(O)
2251-2608(P)

$$\Omega_1(s_1, ..., s_n) = \prod_{i\neq j;1\leq i,j\leq t} \left[s_1(d_{i1} - d_{j1}) + s_2(d_{i2} - d_{j2}) + ... + s_n(d_{in} - d_{jn})\right] \neq 0.$$

We know that $\deg \Omega_1 \leq \binom{t}{2} \leq t^2/2$. Similarly, for $f_2,\ldots,f_k$, we have $\Omega_2,\ldots,\Omega_\ell$. Let

$$\Omega := \prod_{i=1}^{\ell} \Omega_i .$$

If $\Omega\neq 0$, then $r_1,\ldots,r_\ell$ are all separated w.r.t y. And

$$\deg\Omega \leq (t_1^2 + ... + t_\ell^2)/2 \leq (t_1 + ... + t_\ell)^2/2 \leq T^2/2.$$

Using the Schwartz-Zippel Lemma (Lemma 2.6), if $s_1,\ldots,s_n \in [0,T^2]$, then $\text{Prob}[\Omega(s_1, ..., s_n) = 0] \leq \frac{T^2/2}{T^2} = 1/2$, which means that we have a chance higher than 1/2 such that $r_1,\ldots,r_\ell$ are all separate w.r.t y. Therefore, in step 2, choosing $\{s_1,\ldots,s_n\}$ between $[0,T^2]$ is a reasonable construction.

3.2.2. Condition 2

In step 4, we also need to ensure that after substituting, the new univariate polynomial $H_2(y)$ or $H_3(y)$ does not contain terms of some degree that did not exist in the original polynomial .

For example, for the polynomial $P(x_1,x_2)=(x_1-2)(x_2-3)$, if we substitute in $H_2(2y,3y)=(2y-2)(3y-3)=6(y-1)^2$, this will create a square term originally not present in $P(x_1,x_2)$. To avoid this situation, we need to randomly select $p_1,\ldots,p_n$ and input r_i, to make both $f_i(p_1y^{s1},\ldots,p_ny^{sn})$ and $f_i(p_1^2y^{s_1}, ..., p_n^2y^{s_n})$ square-free up to a factor y^k

That is

$$\gcd\left(f_i(p_1y^{s_1}, ..., p_ny^{s_n}), f_i(p_1y^{s_1}, ..., p_ny^{s_n})'\right) = y^{k-1},$$
$$\gcd\left(f_i(p_1^2y^{s_1}, ..., p_n^2y^{s_n}), f_i(p_1^2y^{s_1}, ..., p_n^2y^{s_n})'\right) = y^{k-1},$$

where 'is the derivative w.r.t y

By Lemma 2.5, this means that $\text{res}_y(r_i/y^k, r_i/y^k)\neq 0$ when replacing with $x_i=p_i$ or $x_i=p_i^2$ for $i=1,\ldots,n$.

Solving for the resultant ultimately results in a polynomial with degree related to T,D,n,ℓ, where T is the maximal term number among f_i, D is the degree of $P(x_1,\ldots,x_n)$, n is the number of variables of h, and ℓ is the number of square-free factors after the square-free decomposition P. Since ℓ cannot exceed D, we assume ℓ to be D. Therefore, choosing random prime coefficients $p_1,\ldots,p_n$, where N=poly(T,D,n) (that is a polynomial in variables T,D,n), ensures that no new square-free factor arises when substituting.

3.2.3. Condition 3

First, we know that $\gcd(r_i,r_j)=y^{i,j}$ for $i\neq j$. So we have $\gcd(r_i/y^{i,j},r_j/y^{i,j})=1$.

By Lemma 2.5, we have $\text{res}_y(r_i/y^{i,j},r_j/y^{i,j})\neq 0$.

Thus if $x_i=p_i$ for $i=1,\ldots,n$ is substituted (or $x_i=p_i^2$ for $i=1,\ldots,n$) in $\text{res}_y(r_i/y^{i,j},r_j/y^{i,j})$, then it doesn't vanish with high probability by Lemma 2.6. Then Condition 3 is satisfied with high probability.

4. Cost analysis

The main cost for this algorithm is on the two univariate square-free factorization in step 4. According to [1], the cost of univariate square-free algorithm is approximately $O(\ell^4(c^2d+cd^2))$, for a decomposed polynomial $f=f_1f^2\ldots$

ISSN: 2424-8460(O)
2251-2608(P)

f^{ℓ}_{ℓ}, where ℓ is the number of square-free factors, c=log$K$ where K is the largest coefficient in f_i $\forall$ i, and deg(f_i)=d. For H_1=P(y^{s1}, y^{s2},…,y^{sn}), C is the same as that of the original polynomial H, while d becomes dT^2 since T^2 is the upper bound for selecting the degree s_i for substitution. Hence the cost for calculating H_1 is O(ℓ^4(c^2dT^2+cd^2T^4)).

For H_2=P(p_1·y^{s1},p_2·y^{s2},…,p_n·y^{sn}), c=log$K$ becomes c=log(Cp_1^{e1}...p_n^{en}), where C is the original coefficient and e_1,…,e_n are the degrees for each variable x_1,…,x_n in the original expression. This can be approximated to n·log(P_m^{axe})+log(C), and we suppose e is as large as it can get, which is just d, the degree of f_i and P_{max}:=max(p_1,…,p_n). Thus is approximately ndlog$P$, and the cost of calculating H_2 is O(ℓ^4d^3[n^2(log$P_{\max}$log$C$)$^2T^2$+$n$log$P_{\max}$logCT^4].

The case for H_3 is the same as H_2.

5. Conclusions

This paper presents a novel probabilistic algorithm for multivariate polynomial square-free decomposition that fundamentally improves upon traditional p -adic construction methods. By strategically substituting variables with prime-coefficient univariate terms (x_i=p_iy^{si}) and leveraging the unique factorization property of integers, our approach achieves polynomial-time complexity while avoiding the exponential growth patterns seen in Hensel lifting-based methods.

Our key innovations include:

A dual substitution framework using H_2=P(p_iy^{si}) and H_3=P($p_i^2y^{si}$) that enables coefficient comparison through prime factorization patterns, eliminating the need for complex -adic reconstructions.

A probabilistic guarantee framework underpinned by Schwartz-Zippel lemma analysis, demonstrating success probability per iteration when choosing substitution degrees $s_i \in [0,T^2]$.

An efficient exponent recovery mechanism through sequential prime division of term coefficients in substituted univariate factors.

As demonstrated in our 3-variable case study, the algorithm successfully handles polynomials where traditional Kronecker substitution would require degrees up to y^{19^2}, while our method maintains manageable y^{32} degrees through intelligent prime selection. Complexity analysis reveals our approach reduces dependency on variable count n from exponential to polynomial terms, particularly advantageous for sparse polynomials with many variables but limited term counts. Future directions include extending the prime selection heuristic for finite field polynomials, developing hybrid approaches combining our substitution method with modular arithmetic techniques, and optimizing the parallel recovery of exponents through distributed factorization of substituted coefficients.

References

[1] Yun, D. (1976). On square-free decomposition algorithms. In Proceedings of the third ACM symposium on Symbolic and Algebraic Computation (pp. 26-35).

ISSN: 2424-8460(O)
2251-2608(P)

[2] Thiemann, R., Yamada, A. (2016). Polynomial Factorization. Archive of Formal Proofs.

[3] Gathen, J. von zur, Gerhard, J. (2013). Modern computer algebra. Cambridge University Press.

[4] J. Hu and M. Monagan. A fast parallel sparse polynomial GCD algorithm. Journal of Symbolic Computation, 105:28-63, 2021.