Factoring Non-monic Polynomials Represented by Black Boxes

Tian Chen and Michael Monagan
Department of Mathematics, Simon Fraser University, Canada

Factoring Multivariate Polynomials

The black box representation of a polynomial is one of the most space efficient implicit representations [3]. Given a polynomial $a \in \mathbb{Z}[x_1, \cdots, x_n]$ represented by a black box, we aim to compute its factors in the sparse representation. Figure 1 shows three ways to compute them. Method 0 first interpolates the sparse representation of a and then factors it using a sparse Hensel lifting algorithm, e.g. algorithm CMSHL [1]. Method I is Kaltofen and Trager's method [3] which first constructs black boxes for the factors then applies sparse polynomial interpolation to them. Method II contributed by the authors in [2] computes the factors in the sparse representation directly by a modified CMSHL algorithm. It works for the square-free and monic case. Method II is the most efficient of the three and it outperforms Kaltofen and Trager's algorithm as it requires less number of probes to the black box [2].

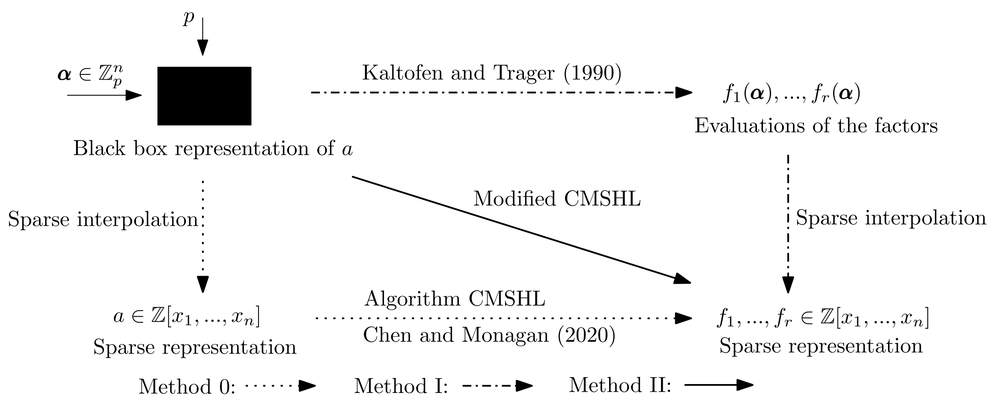

Figure 1: Factoring Multivariate Polynomials Represented by Black Boxes

In this work, we complete the black box factorization problem with a new algorithm that handles non-monic input a. Our new algorithm accepts all cases of input $a \in \mathbb{Z}[x_1, \cdots, x_n]$ including the non-square-free and the non-primitive cases.

The j^{th} Hensel lifting step for our new algorithm is shown in Algorithm CMSHL. The key is that we reduce each Hensel lifting step to many bivariate Hensel lifts by evaluation and sparse interpolation [2]. Then for the non-monic case, we use the square-free part of the bivariate images of a (sqf(A_k) in step 12) to compute all bivariate Hensel lifts. We have modified the algorithm in [4] to make the bivariate Hensel lifts work for the non-monic case. At the end of each bivariate Hensel lift, the leading coefficients are multiplied by a scalar to match the input factors $\hat{f}_{\rho_{j-1}}(x_1, Y_k)$.

Square-free Part of $a \in \mathbb{Z}[x_1, \cdots, x_n]$

We define the square-free part of a, sqf(a), as follows. Let $a = h f_1^{e_1} f_2^{e_2} \cdots f_r^{e_r}$ be the irreducible factorization of $a \in \mathbb{Z}[x_1, \cdots, x_n]$ over $\mathbb{Z}$, where $h \in \mathbb{Z}[x_2, \cdots, x_n]$ is the content of a in x_1. Then,

$$\text{sqf}(a) := f_1 f_2 \cdots f_r = a / \gcd(a, \partial a/\partial x_1).$$

Note: cont(sqf(a), x_1) $= 1$.

Number of Probes to the Black Box

Theorem.

Let $a \in \mathbb{Z}[x_1, \cdots, x_n]$ be represented by a black box **B**. Let $d_j = \deg(a, x_j)$ for $1 \le j \le n$. Let s_j be he number s defined in step 7 of Algorithm CMSHL for the j^{th} Hensel lifting step. The total number of probes to **B** for algorithm CMSHL is

$$\sum_{j=2}^{n} s_j (d_1 + 1)(d_j + 1) \in \mathcal{O}(n d_1 d_{\max} s_{\max})$$

where $d_{\max} = \max_{2 \le j \le n} d_j$ and $s_{\max} = \max_{3 \le j \le n} s_j$.

While in Kaltofen and Trager's method [3], the total number of probes to the black box **B** is $\mathcal{O}(n d_1 d_{\max} \# f_{\max})$, where $\# f_{\max} = \max_{1 \le \rho \le r} \# f_\rho$. $s < \# f_{\max}$ [2].

Benchmark Problem

The benchmark shows CPU times for factoring determinants which have four factors where the size of the factors is much smaller than the size of the determinant. The matrices are generated as follows. We first create two different $n \times n$ symmetric Toeplitz matrices T_1 and T_2 with multivariate polynomial entries. Then we create the $N \times N$ block diagonal matrix $B = [\mathbf{c_1}, \mathbf{c_2}]$ where $\mathbf{c_1} = [T_1, 0]^T$ and $\mathbf{c_2} = [0, T_2]^T$.

Then we create an upper triangular matrix P_u with diagonal entries 1 and entries above the diagonal chosen from $\{0, 1\}$ at random and also a lower triangular matrix P_l with diagonal entries 1 and entries below the diagonal chosen at random from $\{0, 1\}$. Now we create the input matrix $A = P_l B P_u$ so that

$$\det A = \det P_l \det B \det P_u = \det B = \det T_1 \det T_2.$$

Algorithm CMSHL

1: **Input:** $\alpha_j \in \mathbb{Z}_p$, the black box **B**, $\hat{f}_{\rho,j-1} \in \mathbb{Z}_p[x_1, \cdots, x_{j-1}]$ for $1 \le \rho \le r$ s.t. sqf($a_j(x_j = \alpha_j)$) $= \prod_{\rho=1}^{r} \lambda_\rho \prod_{\rho=1}^{r} \hat{f}_{\rho,j-1}$ with $j > 2$, d_i (degrees of a in x_i) for $1 \le i \le n$.
2: **Output:** $\hat{f}_{\rho,j} \in \mathbb{Z}_p[x_1, \cdots, x_j]$ for $1 \le \rho \le r$ s.t. sqf(a_j) $= \prod_{\rho=1}^{r} \lambda_\rho \prod_{\rho=1}^{r} \hat{f}_{\rho,j}$ where $\hat{f}_{\rho,j}(x_j = \alpha_j) = \hat{f}_{\rho,j-1}$ for $1 \le \rho \le r$; Otherwise, **return** FAIL.
3: Let $\hat{f}_{\rho,j-1} = \sum_{i=0}^{df_\rho} \sigma_{\rho,i}(x_2, ..., x_{j-1}) x_1^i$ where $\sigma_{\rho,i} = \sum_{k=1}^{s_{\rho,i}} c_{\rho,ik} M_{\rho,ik}$ with $M_{\rho,ik}$ the monomials in $\sigma_{\rho,i}$ and $df_\rho = \deg(\hat{f}_{\rho,j-1}, x_1)$ for $1 \le \rho \le r$.
4: Pick $\beta = (\beta_2, \cdots, \beta_{j-1}) \in \mathbb{Z}_p^{j-2}$ at random.
5: Evaluate: $\{\mathcal{S}_\rho = \{S_{\rho,i} = \{m_{\rho,ik} = M_{\rho,ik}(\beta), 1 \le k \le s_{\rho,i}\}, 0 \le i \le df_\rho\}, 1 \le \rho \le r\}$.
6: **if** any $|S_{\rho,i}| \ne s_{\rho,i}$ **then return** FAIL **end if**
7: Let s be the maximum of $s_{\rho,i}$.
8: **for** k from 1 to s **do**
9: Let $Y_k = (x_2 = \beta_2^k, \cdots, x_{j-1} = \beta_{j-1}^k)$.
10: $A_k \leftarrow a_j(x_1, Y_k, x_j)$. // via probes to **B** and interpolation ..... $\mathcal{O}(s d_1 d_j \cdot C(\text{probe } \mathbf{B}))$
11: $g_k \leftarrow \gcd(A_k, \frac{\partial A_k}{\partial x_1}) \bmod p$. **if** $\deg(g_k, x_1) \ne d_1 - \sum_{\rho=1}^{r} df_\rho$ **then return** FAIL **end if**
12: $A_k \leftarrow \text{quo}(A_k, g_k) \bmod p$. // to get sqf($A_k$) mod p, no content in x_1.
13: $F_{\rho,k} \leftarrow \hat{f}_{\rho,j-1}(x_1, Y_k)$ for $1 \le \rho \le r$. ..... $\mathcal{O}(s(\#f_1 + \cdots + \#f_r))$
14: **if** any $\deg(F_{\rho,k}) < df_\rho$ for $1 \le \rho \le r$ **then return** FAIL **end if**
15: **if** $\gcd(F_{\rho,k}, F_{\phi,k}) \ne 1$ for any $\rho \ne \phi$ $(1 \le \rho, \phi \le r)$ **then return** FAIL **end if**
16: $\hat{f}_{\rho,k} \leftarrow \text{BivariateHenselLift}(A_k(x_1, x_j), F_{\rho,k}(x_1), \alpha_j, p)$. ..... $\mathcal{O}(s(d_1 d_j^2 + d_1^2 d_j))$
17: **end for**
18: Let $\hat{f}_{\rho,k} = \sum_{l=1}^{t_\rho} \alpha_{\rho,kl} \tilde{M}_{\rho,l}(x_1, x_j)$ for $1 \le k \le s$ where $t_\rho = \#\hat{f}_{\rho,k}$, for $1 \le \rho \le r$.
19: **for** ρ from 1 to r **do**
20: **for** l from 1 to t_ρ **do**
21: $i \leftarrow \deg(\tilde{M}_{\rho,l}, x_1)$.
22: Solve the linear system for $c_{\rho,lk}$: $\left\{\sum_{k=1}^{s_{\rho,i}} m_{\rho,ik}^n c_{\rho,lk} = \alpha_{\rho,nl} \text{ for } 1 \le n \le s_{\rho,i}\right\}$.
23: **end for** ..... $\mathcal{O}(s d_j (\#f_1 + \cdots + \#f_r))$
24: Construct $\hat{f}_{\rho,j} \leftarrow \sum_{l=1}^{t_\rho} \left(\sum_{k=1}^{s_{\rho,i}} c_{\rho,lk} M_{\rho,ik}(x_2, ..., x_{j-1})\right) \tilde{M}_{\rho,l}(x_1, x_j)$.
25: **end for**
26: Pick $\beta = (\beta_2, \cdots, \beta_j) \in \mathbb{Z}_p^{j-1}$ at random.
27: $A_\beta \leftarrow \text{sqf}(a_j(x_1, \beta)) \bmod p$ // via probes to **B**, interpolation, and square-free computation.
28: **if** $\hat{f}_{\rho,j}(x_1, \beta) \mid A_\beta$ **and** $\deg(\hat{f}_{\rho,j}(x_1, \beta)) = df_\rho$ for $1 \le \rho \le r$ **then return** $\hat{f}_{\rho,j}$ for $1 \le \rho \le r$
else return FAIL **end if**

Implementation Results

We have implemented our new algorithm in Maple with major subroutines coded in C. The first table shows the CPU timings (in seconds) for our new algorithm, compared with Maple and Magma's current best determinant and factorization algorithms. Our algorithm is much faster.

n	4	5	6	7	8	9
$N = 2n$	8	10	12	14	16	20
$\#f_i$	7,7,7,7	12,7,12,7	32,32,32,32	56,30,56,30	167,167,167,167	153,294,153,294
$\#\det(A)$	120	701	5162	79740	1716810	7490224
CMSHL total	0.092	0.257	0.972	3.618	19.677	40.219
total probes	721	2112	6453	19584	85189	145065
Maple det	0.057	0.455	7.880	382.80	$>$ 64 gigs	-
Maple factor	0.140	0.109	0.326	1.270	-	-
Maple total	0.197	0.564	8.206	384.07	-	-
Magma det	0.140	1.680	6.290	594.60	$>$ 3h	-
Magma factor	0.800	0.120	0.480	33.140	-	-
Magma total	0.940	1.800	6.770	627.74	-	-

The second table is a breakdown of timings for the 4 major subroutines, namely, (1) the probes to the black box to interpolate the bivariate images $A_k(x_1, x_j)$ of a in step 10 (BB total), (2) evaluating the factors $\hat{f}_{\rho,j-1}$ for $1 \le \rho \le r$ in step 13 (Eval $\hat{f}_{\rho_j-1}$), (3) the non-monic bivariate Hensel lifts in step 16 (BHL), and (4) solving the Vandermonde systems in step 22 (VSolve).

n	4	5	6	7	8	9
$N = 2n$	8	10	12	14	16	20
H.L. x_n total	0.041	0.088	0.385	0.868	5.931	12.163
s (H.L. x_n)	5	9	25	31	131	201
BB total	0.010	0.028	0.138	0.429	3.389	7.678
BB eval	0.005	0.017	0.083	0.322	2.639	5.936
BB det	0.004	0.004	0.038	0.078	0.450	1.015
Eval $\hat{f}_{\rho_j-1}$	0.002	0.001	0.014	0.015	0.074	0.128
BHL	0.001	0.003	0.012	0.018	0.050	0.082
VSolve	0.002	0.001	0.007	0.007	0.016	0.020

References

[1] T. Chen and M. Monagan. The complexity and parallel implementation of two sparse multivariate Hensel lifting algorithms for polynomial factorization. In *Proc. of CASC '20*, pages 150–169. Springer, 2020.

[2] T. Chen and M. Monagan. Factoring multivariate polynomials represented by black boxes – A Maple + C implementation. To appear in *Post Conference Proc. of CASC '21*, 2022.

[3] E. Kaltofen and B. M. Trager. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. Symb. Cmpt.* 9(3):301–320. Elsevier, 1990.

[4] G. Paluck and M. Monagan. New bivariate Hensel lifting algorithm for n factors. *ACM Communications in Computer Algebra*, 53(3):142–145, 2019.