Computing All Factorizations in $\mathbb{Z}_N[x]$

Howard Cheng*
Department of Computer Science
University of Waterloo
Waterloo, Canada
hchcheng@scg.math.uwaterloo.ca

George Labahn
Department of Computer Science
University of Waterloo
Waterloo, Canada
glabahn@scg.math.uwaterloo.ca

ABSTRACT

We present a new algorithm for determining *all* factorizations of a polynomial f in the domain $\mathbb{Z}_N[x]$, a non-unique factorization domain, given in terms of parameters. From the prime factorization of N, the problem is reduced to factorization in $\mathbb{Z}_{p^k}[x]$ where p is a prime and $k \geq 1$. If p^k does not divide the discriminant of f and one factorization is given, our algorithm determines all factorizations with complexity $O(n^3 M(k \log p))$ where n denotes the degree of the input polynomial and $M(t)$ denotes the complexity of multiplication of two t-bit numbers. Our algorithm improves on the method of von zur Gathen and Hartlieb, which has complexity $O(n^7 k(k \log p + \log n)^2)$. The improvement is achieved by processing all factors at the same time instead of one at a time and by computing the kernels and determinants of matrices over $\mathbb{Z}_{p^k}$ in an efficient manner.

Categories and Subject Descriptors

I.1.2 [**Symbolic and Algebraic Manipulation**]: Algorithms

Keywords

Polynomial factorization, structured matrices

1. INTRODUCTION

The factorization of a polynomial f in $\mathbb{Z}_N[x]$ is well understood in the case when N is a prime. Methods for computing such a factorization include Berlekamp's algorithm, the algorithm of Cantor and Zassenhaus, and others [8]. However, when N is not a prime the situation is considerably more complex. Because the domain $\mathbb{Z}_N[x]$ is not a unique factorization domain, it is not sufficient to obtain a single factorization. Rather, we wish to compute all factorizations of a polynomial. The result of such a factorization is given in terms of parameters. For example, one can show that

$$x^2 + 18 \equiv (x + 3 + 9\alpha)\ (x + 24 + 18\alpha) \mod 3^3$$

with $0 \leq \alpha < 3$ gives all factorizations into monic irreducible factors for the polynomial $x^2 + 18$ in $\mathbb{Z}_{3^3}[x]$.

In [6, 7] von zur Gathen and Hartlieb present an algorithm which produces all factorizations of f into irreducible factors in $\mathbb{Z}_{p^k}[x]$, provided that p^k does not divide the discriminant of f and one factorization of f is known. When the prime factorization of N is known, their algorithm can be used to obtain all factorizations of f in $\mathbb{Z}_N[x]$ using the Chinese Remainder Theorem. Their algorithm finds all factorizations of f in $\mathbb{Z}_{p^k}[x]$ by finding parameterized solutions for the kernels of one or more Sylvester matrices in $\mathbb{Z}_{p^r}$ where $1 \leq r \leq k$. The parameterized solution for a kernel is obtained from a Smith Normal Form computation over $\mathbb{Z}$. If f is factored into $m > 2$ irreducible factors, their algorithm examines one factor at a time, computing the kernels of $m-1$ Sylvester matrices. The bit complexity of their algorithm is $O(n^7 k(k \log p + \log n)^2)$. The complexity remains the same even if a faster Smith Normal Form algorithm is used. Although the upper bound may not be tight [6], several important improvements can be made.

In this paper we present a new algorithm for finding all factorizations of a polynomial in $\mathbb{Z}_{p^k}[x]$ from one factorization of f. As in [6, 7], we solve the problem by reducing it to a problem in linear algebra, and require that p^k does not divide the discriminant of f. By processing all factors of a factorization at the same time, the linear algebra problem becomes one of finding parameterized solutions of a kernel of a single striped Sylvester matrix (when there are only two factors then such a matrix is the same as a Sylvester matrix). This approach leads to a faster and simpler algorithm, and also a simplified proof of its correctness. We note that our algorithm does not eliminate the requirement to have a prime factorization of N nor one factorization of f, which are the main bottlenecks in factoring polynomials in $\mathbb{Z}_N[x]$ [7].

In order to compute the parameterized solutions for the kernel of a striped Sylvester matrix, we use matrix triangularization algorithms such as those of Buchmann and Neis [3] and Storjohann [10], instead of the Smith Normal Form computation used in [6, 7]. This improves the complexity of the kernel computation. Finally, both the algorithm in [6, 7] and our algorithm require the computation of the determi-

*Supported by a Natural Sciences and Engineering Research Council Postgraduate Scholarship.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ISSAC 2001, UWO, Canada
©2001 ACM 1-58113-218-2/ 00/ 0008 $5.00

nants of striped Sylvester matrices. By taking advantage of the matrix structure, we improve such computation using a variant of the fraction-free elimination algorithm of Beckermann and Labahn [1].

The remainder of the paper proceeds as follows. Section 2 gives an overview of the problem as well as the algorithm of von zur Gathen and Hartlieb [6, 7]. A generalization of Sylvester matrices for more than two polynomials is given in Section 3. In Section 4, we explain how all the factors of a factorization of f can be processed at the same time, leading to a simplified algorithm that is given and analyzed in Section 5. In Section 6, we discuss how to improve the computation of the kernel and the determinants in our algorithm. The final section includes a conclusion and topics for future research.

2. PRELIMINARIES

In this section, we define the problem of factorization in $\mathbb{Z}_N[x]$ and provide a description of the approach taken by von zur Gathen and Hartlieb. We follow [6, 7] in the presentation of this section. We also define notation used in the remainder of this paper.

Let $f \in \mathbb{Z}_N[x]$ be a polynomial of degree n. We wish to compute all factorizations of f into irreducible factors in $\mathbb{Z}_N[x]$. That is,

$$f \equiv \prod_{i=1}^{m} u_i \mod N,$$

where $u_i \in \mathbb{Z}_N[x]$ are irreducible. We may assume that f and u_i $(1 \le i \le m)$ are monic [6, 7]. If $N = \prod_{1 \le i \le l} p_i^{k_i}$ is the prime factorization of N, we can reduce the problem to factoring f over $\mathbb{Z}_{p_i^{k_i}}$ for $1 \le i \le l$ by the Chinese Remainder Theorem. It is necessary to assume that the factorization of N is known, as Shamir [9] has shown that the problem of factoring N is polynomial-time reducible to the problem of factoring polynomials in $\mathbb{Z}_N[x]$.

Given an integer t, we define its p-adic valuation, $v_p(t)$, to be the largest power of p dividing t $(v_p(0) = \infty)$. We also define $\mathbb{Z}_{(p)}$ to be the p-adic integers [2]. For two polynomials $a = \sum_{i=0}^{n_a} a_i x^i$ and $b = \sum_{i=0}^{n_b} b_i x^i$, with $\deg a = n_a$ and $\deg b = n_b$, we define the Sylvester matrix of a and b to be the following square matrix of order $n_a + n_b$.

$$\mathbf{S}(a,b) = \underbrace{\left[\begin{array}{cccc|cccc} a_0 & & & & b_0 & & & \\ \vdots & \ddots & & & \vdots & \ddots & & \\ a_{n_a} & & \ddots & & \vdots & & b_0 & \\ & \ddots & & a_0 & b_{n_b} & & \vdots & \\ & & \ddots & \vdots & & \ddots & \vdots & \\ & & & a_{n_a} & & & b_{n_b} & \end{array}\right]}_{n_b \quad\quad\quad\quad n_a}$$

The resultant of a and b is $\operatorname{res}(a,b) = \det \mathbf{S}(a,b)$. We also define $r(a,b) = v_p(\operatorname{res}(a,b))$.

Let p be a prime, and suppose that $p^k \nmid \operatorname{disc}(f)$, where $\operatorname{disc}(f)$ denotes the discriminant of f. Let

$$f \equiv \prod_{i=1}^{m} g_i \mod p^k$$

be one factorization of f in $\mathbb{Z}_{p^k}[x]$ into irreducible factors that are pairwise relatively prime, with $n_i = \deg g_i$. As in [6, 7], we will assume that g_i is monic for the remainder of this paper. Such a factorization of f can be obtained by some other means, such as Chistov's algorithm [4, 5]. The algorithm in [6, 7] can now be described.

ALGORITHM 2.1. *Algorithm to compute all factorizations in* $\mathbb{Z}_{p^k}[x]$.

Input: *a monic polynomial* $f \in \mathbb{Z}[x]$, k *such that* $k > v_p(disc(f))$, *and pairwise relatively prime, monic irreducible polynomials* $g_1, \ldots, g_m$ *such that* $f \equiv \prod_{i=1}^m g_i \mod p^k$.

Output: *all factorizations of* f *over* $\mathbb{Z}_{p^k}$ *into monic irreducible factors.*

1. *If* $m = 1$, *then output "f is irreducible" and stop.*
2. *If* $v_p(disc(f)) = 0$, *then output "$f \equiv \prod_{i=1}^m g_i \mod p^k$" and stop.*
3. $w_1 \leftarrow f$.
4. *For* $l = 1, \ldots, m-1$ *do*
   (a) *Let* $h_l = \prod_{l<j\le m} g_j$, *and* $r_l = r(g_l, h_l)$.
   (b) *Lift the factorization* $w_l \equiv g_l h_l \mod p^{k-\sum_{1\le j<l} r_j}$ *to a factorization* $w_l \equiv a_l b_l \mod p^k$, *where* $a_l \equiv g_l$, $b_l \equiv h_l \mod p^{k-\sum_{1\le j\le l} r_j}$. *Note that there may be parameters in* w_l, a_l, b_l.
   (c) *Compute the kernel of* $\mathbf{S}(g_l, h_l)$ *over* $\mathbb{Z}_{p^{r_l}}$, *which gives all factorizations* $w_l \equiv u_l w_{l+1} \mod p^k$ *such that*
   $$u_l \equiv a_l \mod p^{k-r_l}$$
   $$w_{l+1} \equiv b_l \mod p^{k-r_l},$$
   where u_l *and* w_{l+1} *may have parameters.*
5. $u_m \leftarrow w_m$. *Output "$f \equiv \prod_{i=1}^m u_i \mod p^k$" along with the ranges of the parameters in* u_i $(1 \le i \le m)$. □

The algorithm uses the fact that to find all factorizations of $\prod_{i=l}^m g_i$ in $\mathbb{Z}_{p^k}[x]$, it is equivalent to find all factorizations $\prod_{i=l}^m g_i \equiv ab \mod p^k$ such that $a \equiv g_l$ and $b \equiv h_l \mod p^{k-r_l}$, and then process b recursively. Finding a and b is equivalent to solving for $\phi, \psi \in \mathbb{Z}_{p^k}[x]$ such that

$$g_l h_l \equiv (g_l + p^{k-r_l}\phi)(h_l + p^{k-r_l}\psi) \mod p^k,$$

with $\deg\phi < \deg g_l$ and $\deg\psi < \deg h_l$. Equating coefficients of like powers, this reduces to solving for the kernel of $\mathbf{S}(g_l, h_l)$ over $\mathbb{Z}_{p^{r_l}}$. We remark that the kernel is a submodule, not a vector space, of a $\mathbb{Z}_{p^{r_l}}$-module.

In [6, 7], the kernel is obtained by computing the Smith Normal Form of $\mathbf{S}(g_l, h_l)$ over $\mathbb{Z}$. This gives unimodular matrices $\mathbf{P}, \mathbf{Q}$ over $\mathbb{Z}$ such that

$$\mathbf{P} \cdot \mathbf{S}(g_l, h_l) \cdot \mathbf{Q} = \operatorname{diag}(d_1, \ldots, d_{n_l + \deg h_l}) = \mathbf{D},$$

where d_i divides d_{i+1} for $1 \le i < n_l + \deg h_l$. A basis for the kernel of $\mathbf{S}(g_l, h_l)$ is then $\mathbf{Q} \cdot \mathbf{B}$ where $\mathbf{B}$ is a basis of the kernel of $\mathbf{D}$ over $\mathbb{Z}_{p^{r_l}}$. A basis for the kernel of $\mathbf{D}$ can be obtained by examining $v_p(d_i)$ for $1 \le i \le n_l + \deg h_l$.

3. STRIPED SYLVESTER MATRICES

In this section, we recall the notion of striped Sylvester matrices of m polynomials, which generalizes Sylvester matrices of two polynomials. Such matrices will be used to improve the running time of Algorithm 2.1 by removing the loop in step 4. The main result in this section is given in Lemma 3.3, which relates the determinant of a striped Sylvester matrix and the discriminant of the input polynomial. This result is used in proving Theorem 4.4, which characterizes all the factorizations of the input polynomial in $\mathbb{Z}_{p^k}[x]$.

Let $h_i = (\prod_{j=1}^m g_j)/g_i = \sum_{j=0}^{\hat{n}_i} h_j^{(i)} x^j$ for $1 \le i \le m$, where $\hat{n}_i = \deg h_i = n - n_i$. The striped Sylvester matrix of $h_1, \ldots, h_m$ is the following $n \times n$ matrix:

$\mathbf{K}(h_1, \ldots, h_m) =$

$$\underbrace{\left[\begin{array}{ccc} h_0^{(1)} & & \\ \vdots & \ddots & \\ h_{\hat{n}_1}^{(1)} & & h_0^{(1)} \\ & \ddots & \vdots \\ & & h_{\hat{n}_1}^{(1)} \end{array}\right.}_{n_1} \cdots \underbrace{\left.\begin{array}{ccc} h_0^{(m)} & & \\ \vdots & \ddots & \\ h_{\hat{n}_m}^{(m)} & & h_0^{(m)} \\ & \ddots & \vdots \\ & & h_{\hat{n}_m}^{(m)} \end{array}\right]}_{n_m}.$$

When $m = 2$, we have $\mathbf{K}(h_1, h_2) = \mathbf{S}(g_2, g_1)$. We also define $\mathbf{r}(h_1, \ldots, h_m) = v_p(\det \mathbf{K}(h_1, \ldots, h_m))$. If the polynomials $h_1, \ldots, h_m$ are understood, we will use the abbreviation $\mathbf{K} = \mathbf{K}(h_1, \ldots, h_m)$ and $\mathbf{r} = \mathbf{r}(h_1, \ldots, h_m)$. The striped Sylvester matrix $\mathbf{K}$ defined here is the striped Krylov matrix of $h_1, \ldots, h_m$ with multi-index $\vec{n} = (n_1, \ldots, n_m)$ [1].

We now state several properties of $\mathbf{K}$. These results are generalizations of similar properties of Sylvester matrices (see, for example, [11]). We are not aware of these results in the literature, although we remark that they are not difficult to derive.

Lemma 3.1. $\det \mathbf{K} = 0$ *if and only if* $\deg(\gcd(g_a, g_b)) > 0$ *for some* $1 \le a < b \le m$.

Proof. First, note that $\det \mathbf{K} = \sum_{i=1}^m \phi_i h_i$ for some $\phi_i \in \mathbb{Z}[x]$ (not all zero) such that $\deg \phi_i < n_i$ (Theorem 7.1 of [8]).

Suppose $\det \mathbf{K} = 0$, and $\deg(\gcd(g_a, g_b)) = 0$ for all $1 \le a < b \le m$. Then for any $1 \le a \le n$, we have $-\phi_a h_a = \sum_{i \ne a} \phi_i h_i$. Since $g_a \mid h_i$ for $i \ne a$, it follows that $-\phi_a h_a = g_a l$ for some $l \in \mathbb{Z}[x]$. But g_a does not have a nontrivial common factor with g_b for $j \ne a$, so $g_a \mid \phi_a$. By the degree constraint, it follows that $\phi_a = 0$. Thus, $\phi_a = 0$ for all $1 \le a \le n$, a contradiction.

Conversely, if g_a and g_b have a nontrivial common factor φ, then $\varphi \mid h_i$ for all $1 \le i \le m$ because at least one of g_a and g_b divides h_i. Thus, $\varphi \mid \det \mathbf{K}$. If $\det \mathbf{K} \ne 0$, then $\deg \det \mathbf{K} = 0$, contradicting that $\deg \varphi > 0$. □

Lemma 3.2. *Let* $\alpha_{i,j}$ *be indeterminates, and* $g_i = \prod_{j=1}^{n_i} (x - \alpha_{i,j})$. *Then*

$$\det \mathbf{K} = \prod_{i=1}^{m-1} \prod_{j=i+1}^{m} \prod_{k=1}^{n_i} \prod_{l=1}^{n_j} (\alpha_{i,k} - \alpha_{j,l}). \tag{1}$$

Proof. Each coefficient of h_i is an elementary symmetric function of $\alpha_{j,l}$ for $j \ne i$. Now, $\det \mathbf{K}$ is a homogeneous polynomial of the coefficients of h_i (of degree n_i), so it must be a symmetric function of $\alpha_{i,j}$. Therefore each $\alpha_{i,j}$ has degree at most $\sum_{k=1}^m n_k - n_i$ in $\det \mathbf{K}$.

By Lemma 3.1, $\det \mathbf{K} = 0$ if and only if g_a, g_b have a nontrivial common factor. That is, they have a common root. Thus, $\det \mathbf{K}$ is divisible by $(\alpha_{i,k} - \alpha_{j,l})$ for $1 \le i < j \le m$, $1 \le k \le n_i$, $1 \le l \le n_j$. Hence, it is also divisible by the RHS of (1) since the factors are relatively prime. Since the RHS of (1) has degree $\sum_{k=1}^m n_k - n_i$ for each $\alpha_{i,j}$, it follows that the degree of each $\alpha_{i,j}$ on the LHS is also $\sum_{k=1}^m n_k - n_i$. Thus, the two sides differ by at most a constant factor.

To see that both sides are indeed equal, we note that both sides have the term

$$\prod_{i=1}^{m-1} \prod_{k=1}^{n_i} \alpha_{i,k}^{\sum_{j=i+1}^m n_j}.$$

This can be seen on the RHS of (1) by expanding the product. For the LHS, we note that the coefficients of h_i are elementary symmetric functions of $\alpha_{j,l}$. We examine the term in the expansion of $\det \mathbf{K}$ with entries of degree $\sum_{i<k} n_i$ (in $\alpha_{i,j}$) for the stripe of h_k. The sign in the expansion is $(-1)^{(\sum_{i=1}^m n_i)^2 - \sum_{i=1}^m n_i^2} = 1$. Furthermore, this is the only possible way to obtain this term in the expansion for the determinant. In the first n_m rows, we must choose entries from the last n_m columns. For the next n_{m-1} rows, again, we must choose the second last stripe of n_{m-1} rows. Continuing this way, we see that this is the only way to produce this term, so the coefficient of the term must be 1. □

Lemma 3.3. *If* $f = \prod_{i=1}^m g_i$, *then*

$$disc(f) = (\det \mathbf{K})^2 \prod_{i=1}^m disc(g_i).$$

Proof. By Lemma 3.2,

$$(\det \mathbf{K})^2 = \prod_{i=1}^{m-1} \prod_{j=i+1}^{m} \prod_{k=1}^{n_i} \prod_{l=1}^{n_j} (\alpha_{i,k} - \alpha_{j,l})^2,$$

which is the product of squared differences of roots of g_i and g_j for $i \ne j$. Since the product of squared differences of roots of g_i is $\mathrm{disc}(g_i)$, the product of squared differences of the roots in g_i, g_j for $1 \le i \le j \le m$ is $(\det \mathbf{K})^2 \prod_{i=1}^m \mathrm{disc}(g_i)$. The result now follows from the fact that the roots of f are the roots of g_i for $1 \le i \le m$. □

4. PROCESSING ALL FACTORS OF A FACTORIZATION AT THE SAME TIME

When we examine Algorithm 2.1, we see that one factor is processed in each loop iteration of step 4, and the kernels of

$m-1$ Sylvester matrices are computed. We now show how to process all factors of a factorization at the same time and solve for the kernel of a matrix over $\mathbb{Z}_{p^{\mathbf{r}}}$ only once. The main tool we use is the striped Sylvester matrix described in the previous section. The results given here are similar to those given in [6, 7].

The next result relates the problem of linear diophantine equations to a linear algebra problem.

PROPOSITION 4.1. *Suppose* $\det \mathbf{K} \neq 0$. *Let* $l \in \mathbb{Z}[x]$ *with* $\deg l < n$. *Then there exist unique polynomials* $\phi_i \in \mathbb{Z}[x]$ *with* $\deg \phi_i < n_i$ $(1 \leq i \leq m)$ *such that*

$$(\det \mathbf{K})\ l = \sum_{i=1}^{m} \phi_i h_i. \tag{2}$$

Proof. We write $l = \sum_{j=0}^{n-1} l_j x^j$ where $l_j \in \mathbb{Z}$, and $\phi_i = \sum_{j=0}^{n_i-1} \phi_{i,j} x^j$. Then (2) is equivalent to

$$\mathbf{K}\vec{\phi} = (\det \mathbf{K})\ \vec{l}, \tag{3}$$

where $\vec{l} = [l_0 \cdots l_{n-1}]^T$ and $\vec{\phi} = [\phi_{1,0} \cdots \phi_{1,n_1-1} \cdots \phi_{m,0} \cdots \phi_{m,n_m-1}]^T$. Since $\det \mathbf{K} \neq 0$, we can write

$$\vec{\phi} = (\det \mathbf{K})\ \mathbf{K}^{-1}\vec{l}. \tag{4}$$

The entries of $(\det \mathbf{K})\ \mathbf{K}^{-1}$ are in $\mathbb{Z}$, so the unique solution to (2) is $\phi_i \in \mathbb{Z}[x]$ for $1 \leq i \leq m$. □

Suppose that $\det \mathbf{K} = p^{\mathbf{r}} b$ for some $b \in \mathbb{Z}$ such that $p \nmid b$. Applying the above proposition modulo $p^{\mathbf{r}+1}$ and noting that b is invertible $\mod p^{\mathbf{r}+1}$, we get the following corollary.

COROLLARY 4.2. *In order to compute* $\phi_i \in \mathbb{Z}[x]$ *such that* $\deg \phi_i < n_i$ *and* $p^{\mathbf{r}} l \equiv \sum_{i=1}^{m} \phi_i h_i \mod p^{\mathbf{r}+1}$, *it suffices to determine* $\mathbf{K} \bmod p^{\mathbf{r}+1}$.

The following is a technical lemma that is required in the proof of our algorithm.

LEMMA 4.3. *Let* $g_i, u_i \in \mathbb{Z}[x]$ $(1 \leq i \leq m)$ *be monic polynomials such that* $\det \mathbf{K} \neq 0$ *and* $g_i \equiv u_i \mod p^{\mathbf{r}+1}$. *Then* $\mathbf{r} = \mathbf{r}(\hat{u}_1, \ldots, \hat{u}_m)$, *where* $\hat{u}_i = (\prod_{j=1}^{m} u_j)/u_i$.

Proof. Since $\det \mathbf{K}(\hat{u}_1, \ldots, \hat{u}_m) = \det \mathbf{K} + p^{\mathbf{r}+1} l$ for some $l \in \mathbb{Z}$, we have $\det \mathbf{K}(\hat{u}_1, \ldots, \hat{u}_m) = p^{\mathbf{r}}(b + pl)$ for some $b \in \mathbb{Z}$ with $p \nmid b$. It follows that $p \nmid (b+pl)$ and hence $\det \mathbf{K}(\hat{u}_1, \ldots, \hat{u}_m) = p^{\mathbf{r}} B$ where $p \nmid B$. Thus, $\mathbf{r} = \mathbf{r}(\hat{u}_1, \ldots, \hat{u}_m)$. □

The following theorem provides a characterization of all factorizations of f in $\mathbb{Z}_{p^k}[x]$, leading directly to our algorithm.

THEOREM 4.4. *Let* $p \in \mathbb{Z}$ *be prime,* $k \in \mathbb{N}$, *and* $f, g_1, \ldots, g_m$ *be monic polynomials in* $\mathbb{Z}[x]$. *Suppose that:*

1. $f \equiv \prod_{i=1}^{m} g_i \mod p^k$;
2. $\det \mathbf{K} \neq 0$;
3. $k > v_p(disc(f))$.

Then there exist monic $u_1, \ldots, u_m \in \mathbb{Z}_{(p)}[x]$ *such that* $f = \prod_{i=1}^{m} u_i$ *in* $\mathbb{Z}_{(p)}[x]$, $u_i \equiv g_i \mod p^{k-\mathbf{r}}$.

Proof. First, Lemma 3.3 remains true for factorizations modulo p^k, so

$$\operatorname{disc}(f) = (\det \mathbf{K})^2 \prod_{i=1}^{m} \operatorname{disc}(g_i) + p^k l$$

for some $l \in \mathbb{Z}$. Now, $k > v_p(\operatorname{disc}(f))$ and hence $k > v_p((\det \mathbf{K})^2 \prod_{i=1}^{m} \operatorname{disc}(g_i))$. But $2\mathbf{r} = v_p((\det \mathbf{K})^2)$, so $k > 2\mathbf{r}$.

Now, we show by induction on $i \geq 1$ how to construct $\phi_{j,i} \in \mathbb{Z}[x]$ with $\deg \phi_{j,i} < n_j$, such that if

$$f \equiv \prod_{j=1}^{m} A_j \mod p^{k+i-1} \tag{5}$$

with $A_j \in \mathbb{Z}[x]$, $A_j \equiv g_j \mod p^{k-\mathbf{r}}$, and A_j is monic, then

$$f \equiv \prod_{j=1}^{m} (A_j + p^{k-\mathbf{r}+i-1}\phi_{j,i}) \mod p^{k+i}.$$

Note that $A_j + p^{k-\mathbf{r}+i-1}\phi_{j,i}$ is monic since $\deg \phi_{j,i} < n_j$.

From (5), $f = \prod_{j=1}^{m} A_j + p^{k+i-1} l$ where $l \in \mathbb{Z}[x]$ and $\deg l < \sum_{j=1}^{m} n_j = n$ (f and $\prod_{i=1}^{m} A_j$ are monic). Since $A_j \equiv g_j \mod p^{k-\mathbf{r}}$ and $k > 2\mathbf{r}$, we have $\mathbf{r} = \mathbf{r}(\hat{A}_1, \ldots, \hat{A}_m)$ by Lemma 4.3. By Corollary 4.2, there exist $\phi_{j,i} \in \mathbb{Z}[x]$ with $\deg \phi_{j,i} < n_j$ such that

$$p^{\mathbf{r}} l \equiv \sum_{j=1}^{m} \phi_{j,i} \prod_{t \neq j} A_t \mod p^{\mathbf{r}+1}.$$

Then

$$\begin{aligned}
& f - \prod_{j=1}^{m} (A_j + p^{k-r+i-1}\phi_{j,i}) \\
& \equiv f - \prod_{j=1}^{m} A_j - p^{k-r+i-1} \sum_{j=1}^{m} \phi_{j,i} \prod_{t \neq j} A_t \mod p^{k+i} \\
& \equiv p^{k-r+i-1} \left(p^r l - \sum_{j=1}^{m} \phi_{j,i} \prod_{t \neq j} A_t \right) \mod p^{k+i} \\
& \equiv 0 \mod p^{k+i}.
\end{aligned}$$

Thus, $u_j = g_j + \sum_{i \geq 1} p^{k-r+i-1}\phi_{j,i} \in \mathbb{Z}_p[x]$ have the desired properties. □

By Theorem 4.4, given a factorization $f \equiv \prod_{i=1}^{m} g_i \mod p^k$ into monic irreducible factors, all factorizations into monic irreducible factors must have the form

$$f \equiv \prod_{i=1}^{m} (g_i + p^{k-\mathbf{r}}\phi_i) \mod p^k. \tag{6}$$

To find all solutions ϕ_i of (6), we see that

$$f \equiv \prod_{i=1}^{m} g_i + p^{k-\mathbf{r}} \left(\sum_{i=1}^{m} \phi_i h_i \right) \mod p^k$$

$$\Leftrightarrow 0 \equiv p^{k-\mathbf{r}} \left(\sum_{i=1}^{m} \phi_i h_i \right) \mod p^k$$

$$\Leftrightarrow 0 \equiv \left(\sum_{i=1}^{m} \phi_i h_i \right) \mod p^{\mathbf{r}},$$

which is equivalent to computing the kernel of $\mathbf{K}$ modulo $p^{\mathbf{r}}$. Now, $p^{\mathbf{r}}$ is smaller than p^k, and is bounded by $p^{\lceil k/2 \rceil}$. Therefore, we only need to compute $\mathbf{K} \bmod p^{\lceil k/2 \rceil}$.

5. ALGORITHM

We now state the algorithm to handle all factors at the same time.

ALGORITHM 5.1. *Algorithm to compute all factorizations in* $\mathbb{Z}_{p^k}[x]$.

Input: *a monic polynomial* $f \in \mathbb{Z}[x]$, k *such that* $k > v_p(disc(f))$, *and pairwise relatively prime, monic irreducible polynomials* $g_1, \ldots, g_m$ *such that* $f \equiv \prod_{i=1}^{m} g_i \bmod p^k$.

Output: *all factorizations of* f *in* $\mathbb{Z}_{p^k}[x]$ *into monic irreducible factors.*

1. *If* $m = 1$, *then output "f is irreducible" and stop.*
2. *If* $v_p(disc(f)) = 0$, *then output "$f \equiv \prod_{i=1}^{m} g_i \bmod p^k$" and stop.*
3. *Compute* $h_i = f/g_i$ *for* $1 \leq i \leq m$, *and form the striped Sylvester matrix* $\mathbf{K} \bmod p^{\lceil k/2 \rceil}$.
4. *Compute* $\mathbf{r} = v_p(\det \mathbf{K})$.
5. *Compute the kernel of* $\mathbf{K}$ *over* $\mathbb{Z}_{p^{\mathbf{r}}}$, *which gives all factorizations*

$$f \equiv \prod_{i=1}^{m} u_i^{(\alpha_1, \ldots, \alpha_j)} \mod p^k$$

such that

$$u_i^{(\alpha_1, \ldots, \alpha_j)} \equiv g_i \mod p^{k-\mathbf{r}},$$

along with the ranges of the parameters $\alpha_1, \ldots, \alpha_j$. □

Our algorithm is much simpler than Algorithm 2.1. In Algorithm 2.1, the lifting step dominates the running time of each iteration. By processing all factors at the same time, the lifting step is eliminated and we need to solve for the kernel only once. The size of the matrix is the same as that formed in the first iteration of Algorithm 2.1. Therefore, our approach requires no more work than one iteration of Algorithm 2.1.

EXAMPLE 5.2. *Let us consider the example from [6, 7]. Let* $f = x^5 + 9x^4 + 15x^3 + 54x^2 + 36x + 81$, $p = 3$, *and* $k = 15$. *One factorization of* f *in* $\mathbb{Z}_{p^k}[x]$ *is* $f \equiv g_1 g_2 g_3 \bmod p^k$ *where*

$$g_1 = x^2 + 3, \quad g_2 = x + 6006780,$$
$$g_3 = x^2 + 8342136x + 6483495.$$

We get

$$h_1 \equiv x^3 + 9x^2 + 12x + 27,$$
$$h_2 \equiv x^4 + 3105x^3 + 1230x^2 + 2754x + 3681,$$
$$h_3 \equiv x^3 + 3465x^2 + 3x + 3834,$$

and so

$$\mathbf{K} = \begin{bmatrix} 27 & 0 & 3681 & 3834 & 0 \\ 12 & 27 & 2754 & 3 & 3834 \\ 9 & 12 & 1230 & 3465 & 3 \\ 1 & 9 & 3105 & 1 & 3465 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix},$$

all modulo $p^{\lceil k/2 \rceil}$. *We find that* $\mathbf{r} = 6$, *and a basis of the kernel of* $\mathbf{K}$ *over* $Z_{p^{\mathbf{r}}}$ *is seen to be* $\{3^5 \cdot [0,2,1,0,0]^T, 3^4 \cdot [8,0,0,1,0]^T, 3^3 \cdot [0,4,24,0,26]^T\}$. *Thus, we can write all 729 factorizations of* f *as* $f \equiv \prod_{i=1}^{m} u_i \bmod p^k$ *where*

$$u_1 = g_1 + 3^9((2 \cdot 3^5 \alpha_1 + 4 \cdot 3^3 \cdot \alpha_3)x + 8 \cdot 3^4 \alpha_2),$$
$$u_1 = g_2 + 3^9(3^5 \alpha_1 + 24 \cdot 3^3 \alpha_3),$$
$$u_1 = g_3 + 3^9(26 \cdot 3^3 \alpha_3 x + 3^4 \alpha_2),$$

and $0 \leq \alpha_1 < 3$, $0 \leq \alpha_2 < 9$, $0 \leq \alpha_3 < 27$. □

The problem of finding all factorizations of f in $\mathbb{Z}_{p^k}[x]$ reduces to the problem of computing the kernel of $\mathbf{K}$ over $\mathbb{Z}_{p^{\mathbf{r}}}$. The complexity of the reduction is simply the complexity of forming $\mathbf{K} \bmod p^{\lceil k/2 \rceil}$. To simplify our complexity results, we define $M(t)$ to be the complexity of multiplication of two t-bit integers.

THEOREM 5.3. *The entries of* $\mathbf{K} \bmod p^{\lceil k/2 \rceil}$ *can be computed in* $O(n^2 M(k \log p))$ *bit operations.*

Proof. For $1 \leq i \leq m$, we compute $h_i = f/g_i$ in $O(nn_i)$ multiplications, each involving coefficients of size $O(k \log p)$. Thus, to compute all the coefficients in $\mathbf{K} \bmod p^{\lceil k/2 \rceil}$, we need to perform a total of $O(n^2 M(k \log p))$ bit operations. □

6. TRIANGULARIZATION OF STRIPED SYLVESTER MATRICES OVER $\mathbb{Z}_{p^k}$

It is possible to find parameterized solutions of the kernel of $\mathbf{K}$ modulo $p^{\mathbf{r}}$ by using the Smith Normal Form as done in [6, 7]. The fastest Smith Normal Form algorithm that we are aware of can be used to compute the Smith Normal Form of $\mathbf{K}$ in

$$O(n^\omega \log n \log \beta + n^2 \log n \log \beta M(\log \beta))$$

bit operations, where $\beta = (\sqrt{n} p^{\mathbf{r}})^n$ and $O(n^\omega)$ is the arithmetic complexity of multiplying two $n \times n$ matrices [10]. As such, we can find a basis for the kernel in the same number of operations, and the complexity of Algorithm 5.1 becomes

$$O(n^\omega \log n \log \beta + n^2 \log n \log \beta M(\log \beta)),$$

which is already a significant improvement over Algorithm 2.1 of von zur Gathen and Hartlieb.

In fact, the Smith Normal Form provides more information about the matrix than is required for our kernel computation problem. Instead, we can apply the algorithms of Buchmann and Neis [3] or Storjohann (Chapter 4) [10] to triangularize matrices into special forms over $\mathbb{Z}_{p^{\mathbf{r}}}$ in $O(n^3 M(\mathbf{r}\log p))$ and $O(n^\omega M(\mathbf{r}\log p))$ bit operations, respectively. Using these algorithms, we can obtain the kernel of a matrix over $\mathbb{Z}_{p^{\mathbf{r}}}$ in the same time. Thus, we can further reduce the complexity of Algorithm 5.1 to $O(n^\omega M(k\log p))$ because $\mathbf{r} < k$.

The dominant cost of our algorithm is no longer attributed only to the kernel computation. It is now important to also consider the cost of computing the discriminant and $\det \mathbf{K}$, both of which are determinants of structured matrices. The determinant of a matrix over $\mathbb{Z}_{p^k}$ can also be obtained once a triangular form over $\mathbb{Z}_{p^k}$ is determined. Thus, the problems of computing the kernel of $\mathbf{K}$, computing $\det \mathbf{K}$, and determining whether $k > v_p(\mathrm{disc}(f)) = v_p(\det \mathbf{K}(f, f'))$ have the same complexity as triangularizing matrices over $\mathbb{Z}_{p^k}$. Using the matrix triangularization algorithms mentioned above, the overall complexity of Algorithm 5.1 is $O(n^\omega M(k\log p))$.

6.1 Computing Determinants of Striped Sylvester Matrices

The triangularization algorithms mentioned above work on general matrices and do not take advantage of the structure present in the striped Sylvester matrix. We now present an algorithm which takes advantage of the special structure to compute the determinants of striped Sylvester matrices over $\mathbb{Z}_{p^k}$. As such, the algorithm is more efficient than algorithms for general matrices. However, the computation of the kernel is still the dominant cost of the algorithm.

If we relate the columns of the ith stripe of $\mathbf{K}$ to the polynomials $x^j h_i$ $(0 \le j < n_i)$ and the rows of $\mathbf{K}$ to the coefficients of x^i, then the kernel of $\mathbf{K}$ is the same as the solutions $\vec{u} = (u_1, \ldots, u_m) \in \mathbb{Z}_{p^k}[x]^m$ to the polynomial equation

$$u_1 h_1 + \cdots + u_m h_m = x^n r \tag{7}$$

for some power series r, where $\deg u_i < n_i$. The right-hand side of (7) denotes a power series whose coefficients for $1, x, \ldots, x^{n-1}$ are all zero. More generally, we have the polynomial equation

$$u_1 h_1 + \cdots + u_m h_m = x^\sigma r \tag{8}$$

for some power series r, where $\deg u_i < n_i$ and $0 \le \sigma \le n$.

If the polynomials in (8) have coefficients over the quotient field $\mathbb{F}$ of an integral domain, the FFFG (Fraction-Free Fast Gaussian) elimination algorithm of Beckermann and Labahn [1] solves the equation by computing a module basis of size m for the solutions of (8). That is, it computes a set of polynomial vectors

$$\left\{ \vec{u}_i = (u_i^{(1)}, \ldots, u_i^{(m)}) : \vec{u}_i \text{ satisfies } (8), 1 \le i \le m \right\},$$

such that all solutions $\vec{u}$ of (8) can be expressed as

$$\vec{u} = \sum_{i=1}^{m} c_i \vec{u}_i$$

with $c_i \in \mathbb{F}[x]$. The module basis for order σ computed by the FFFG algorithm is given by the columns of a matrix polynomial $\mathbf{M}_\sigma$. We can view the solutions to (8), and hence the columns of $\mathbf{M}_\sigma$, as the column operations required to eliminate the columns for the first σ rows. By increasing the value of σ, we can view (8) as a matrix triangularization problem.

We now give a brief overview of a simplified version of the FFFG elimination algorithm modified to work over $\mathbb{Z}_{p^s}$ for $s \ge 1$ (Algorithm 6.1). As noted above, the algorithm can be viewed as a matrix triangularization algorithm. Instead of performing the elimination in a fraction-free way, we choose as pivots the elements with the least p-adic valuation to ensure that all operations can be performed in $\mathbb{Z}_{p^s}$ and are invertible. The algorithm requires us to specify a multi-index $\vec{n}$, whose ith component indicates the number of columns in the ith stripe of $\mathbf{K}$. In our case, we set $\vec{n} = (n_1, \ldots, n_m)$ where $n_i = \deg g_i$. The algorithm also maintains a second multi-index $\vec{v}_\sigma$, whose ith component, $\vec{v}_\sigma^{(i)}$, records the number of columns from the ith stripe that have been used as pivots in the elimination process after σ steps. At each step, the degree of the (i,j)th entry of $\mathbf{M}_\sigma$ is bounded by $\vec{v}_\sigma^{(i)} - 1 + \delta_{ij}$, where δ_{ij} is the Kronecker delta.

The algorithm maintains only one column in each stripe to improve efficiency. The column maintained for stripe i is the one corresponding to $x^{\vec{v}_\sigma^{(i)}} h_i$, so the columns currently maintained are indicated by $\vec{v}_\sigma$. Once a column has been used as a pivot, it is no longer needed and the next column in the stripe can be computed. The next column is obtained efficiently by multiplying the pivot column by x and then subtracting a linear combination of the current columns from the remaining stripes to satisfy the degree constraints on the entries of $\mathbf{M}_\sigma$. The degree constraints allow us to view the ith column of $\mathbf{M}_\sigma$ as the linear combination of the previous pivot columns to eliminate the first σ rows of the current column in the ith stripe. To simplify the algorithm, we do not maintain the matrix $\mathbf{K}$ itself, but rather the matrices $\mathbf{M}_\sigma$ which transform $\mathbf{K}$ into triangular form as in [1]. We also compute only $\pm\det \mathbf{K}$ as we are only interested in the p-adic valuation of $\det \mathbf{K}$.

In steps 1–5, a non-zero element with the least p-adic valuation is chosen among the current columns of each stripe as the pivot. The determinant of the matrix is simply the product of all the entries used as pivots during the elimination process. In step 8, row σ of the current column in each non-pivot stripe is eliminated, and in step 9, the next column in the pivot stripe is eliminated and updated to become the current column in that stripe. We refer the reader to [1] for more details on the FFFG elimination algorithm, including its correctness and complexity.

ALGORITHM 6.1. *Modified FFFG elimination over* $\mathbb{Z}_{p^s}$.

Input: *a vector of polynomials* $\vec{h} = (h_1, \ldots, h_m)$, *a multi-index* $\vec{n} = (n_1, \ldots, n_m)$, $n = \deg f$, *and* $s \ge 1$.

Output: $d \equiv \pm\det \mathbf{K} \mod p^s$.

Initialization: $\mathbf{M}_0 \leftarrow \mathbf{I}_m$, $d \leftarrow 1$, $\vec{\nu}_0 \leftarrow \vec{0}$.

Iterative Step: *for* $\sigma = 0, 1, 2, \ldots, n-1$:

1. *Calculate for* $\ell = 1, \ldots, m$:

$$r^{(\ell)} \leftarrow \mathrm{coefficient}(\vec{h} \cdot \mathbf{M}_\sigma^{(\cdot,\ell)}, x^\sigma).$$

2. *Let* $t = \min_{1 \le \ell \le m} v_p(r^{(\ell)})$.
3. *Define set* $\Lambda = \Lambda_\sigma = \{\ell \in \{1, \dots, m\} : r^{(\ell)} \neq 0,\ v_p(r^{(\ell)}) = t,\ \vec{v}^{(\ell)} < \vec{n}^{(\ell)}\}$.
4. *If* $\Lambda = \{\}$ *then* $d \leftarrow 0$, *return.*
5. *Otherwise, let* $\vec{\nu}_{\sigma+1} \leftarrow \vec{\nu}_\sigma + \vec{e}_\pi$, *where* π *is any element in* Λ *and* $\vec{e}_\pi$ *is the* π*th unit vector.*
6. *Update* d: $d \leftarrow d \cdot r^{(\pi)} \mod p^s$.
7. *Calculate for* $\ell = 1, \dots, m$, $\ell \neq \pi$, *the leading coefficients:*
$$P^{(\ell)} \leftarrow \text{coefficient}(\mathbf{M}_\sigma^{(\ell,\pi)}, x^{\vec{\nu}_\sigma^{(\ell)}-1}).$$
8. *Increase order for* $\ell = 1, \dots, m$, $\ell \neq \pi$:
$$\mathbf{M}_{\sigma+1}^{(\cdot,\ell)} \leftarrow \mathbf{M}_\sigma^{(\cdot,\ell)} - \mathbf{M}_\sigma^{(\cdot,\pi)} \cdot (r^{(\ell)}/p^t) \cdot (r^{(\pi)}/p^t)^{-1} \mod p^s.$$
9. *Increase order for* $\ell = \pi$ *and adjust degrees:*
$$\mathbf{M}_{\sigma+1}^{(\cdot,\pi)} \leftarrow x \cdot \mathbf{M}_\sigma^{(\cdot,\pi)} - \sum_{\ell \neq \pi} \mathbf{M}_{\sigma+1}^{(\cdot,\ell)} \cdot P^{(\ell)} \mod p^s.$$

Return: *d.* □

EXAMPLE 6.2. *Consider the polynomials in Example 5.2:*

$$h_1 \equiv x^3 + 9x^2 + 12x + 27,$$
$$h_2 \equiv x^4 + 3105x^3 + 1230x^2 + 2754x + 3681,$$
$$h_3 \equiv x^3 + 3465x^2 + 3x + 3834,$$

modulo $p^{\lceil k/2 \rceil} = 3^8$. *We wish to compute* $\mathbf{r}$, *so we compute* $\pm\det \mathbf{K}$ *over* $\mathbb{Z}_{3^8}$. *We run Algorithm 6.1 with* $\vec{n} = (2, 1, 2)$ *(the degrees of* g_i *in Example 5.2) and* $n = 5$.

When $\sigma = 0$, *we have*

$$\vec{h} \cdot \mathbf{M}_0 = (h_1, h_2, h_3).$$

We now eliminate the coefficients corresponding to $x^\sigma = 1$. *We see that* h_2 *has the least* p*-adic valuation in its constant coefficient, so that* $t = v_p(3681) = 2$ *and* $\pi = 2$. *Using the constant coefficient of* h_2 *as a pivot, we eliminate the constant coefficients of* h_1 *and* h_3 *to get*

$$\mathbf{M}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 5406 & x & 15 \\ 0 & 0 & 1 \end{bmatrix}.$$

Then

$$\vec{h} \cdot \mathbf{M}_1 = (5406x^4 + 2593x^3 + 3096x^2 + 1227x,$$
$$x^5 + 3105x^4 + 1230x^3 + 2754x^2 + 3681x,$$
$$15x^4 + 649x^3 + 2232x^2 + 1947x),$$

and $d = 3681$.

For the next iteration, $\sigma = 1$, *we see that* $v_p(1227) = v_p(1947) = 1$, *so 1227 and 1947 have the least* p*-adic valuation in the coefficient of* x. *We can choose* $\pi = 1$ *to break the tie in step 5 of the algorithm. Using the first element of* $\vec{h} \cdot \mathbf{M}_1$ *to eliminate the coefficient of* x *in the remaining elements, we obtain*

$$\mathbf{M}_2 = \begin{bmatrix} x + 3096 & 6558 & 6014 \\ 6426 & x + 3465 & 1944 \\ 0 & 0 & 1 \end{bmatrix},$$

where now

$$\vec{h} \cdot \mathbf{M}_2 = (6427x^4 + 3834x^3 + 6168x^2,$$
$$x^5 + 9x^4 + 12x^3 + 27x^2,$$
$$1944x^4 + 6015x^3 + 1458x^2),$$

and $d = 810$.

The intermediate results obtained for the remaining values of σ *are:*

$\sigma = 2$: $\vec{h} \cdot \mathbf{M}_2 = (6427x^4 + 3834x^3 + \boxed{6168}x^2, x^5 + 9x^4 + 12x^3 + 27x^2, 1944x^4 + 6015x^3 + 1458x^2)$, $\pi = 1$, $d = 810$,

$\sigma = 3$: $\vec{h} \cdot \mathbf{M}_3 = (x^5 + 3105x^4 + 1227x^3, x^5 + 3105x^4 + 1227x^3, 5103x^4 + \boxed{6015}x^3)$, $\pi = 3$, $d = 3888$.

$\sigma = 4$: $\vec{h} \cdot \mathbf{M}_4 = (x^5 + 189x^4, x^5 + 189x^4, \boxed{6015}x^4)$, $\pi = 3$, $d = 2916$.

Thus, $\mathbf{r} = v_p(d) = 6$.

We can also view the elimination process in terms of matrix triangularization. After the elimination step for $\sigma = 0$, *we obtain the matrix*

$$\begin{bmatrix} 0 & 0 & \boxed{3681} & 0 & 0 \\ 1227 & 27 & 2754 & 1947 & 3834 \\ 3096 & 12 & 1230 & 2232 & 3 \\ 2593 & 9 & 3105 & 649 & 3465 \\ 5406 & 1 & 1 & 15 & 1 \end{bmatrix},$$

where the boxed element is the pivot chosen during the elimination process. After the elimination step for $\sigma = 1$, *we obtain the matrix*

$$\begin{bmatrix} 0 & 0 & \boxed{3681} & 0 & 0 \\ \boxed{1227} & 0 & 2754 & 0 & 0 \\ 3096 & 6168 & 1230 & 1458 & 1542 \\ 2593 & 3834 & 3105 & 6015 & 2052 \\ 5406 & 6427 & 1 & 1944 & 514 \end{bmatrix}.$$

Continuing on, the final triangular matrix obtained by the algorithm is

$$\begin{bmatrix} 0 & 0 & \boxed{3681} & 0 & 0 \\ \boxed{1227} & 0 & 2754 & 0 & 0 \\ 3096 & \boxed{6168} & 1230 & 0 & 0 \\ 2593 & 3834 & 3105 & \boxed{6015} & 0 \\ 5406 & 6427 & 1 & 5103 & \boxed{6015} \end{bmatrix}.$$

□

The bit complexity of the FFFG elimination algorithm is $O(mn^2 M(s \log p))$ [1], which is better than the triangularization algorithms for general matrices when m is small. This should not be surprising—$\mathbf{K}$ has more structure when m is small relative to n. Therefore, we should use this algorithm to compute the determinant when m is small.

THEOREM 6.3. *The bit complexity of computing whether* $k > v_p(disc(f))$ *is* $O(n^2 M(k \log p))$ *and the complexity of computing* $\mathbf{r}$ *is* $O(mn^2 M(k \log p))$.

Proof. Each of these quantities can be obtained by computing the determinant of a striped Sylvester matrix over $\mathbb{Z}_{p^k}$. Applying Algorithm 6.1 gives the bit complexity $O(mn^2 M(k \log p))$. To compute $k > v_p(\mathrm{disc}(f))$, we only have to compute $\det \mathbf{K}(f, f')$ over $\mathbb{Z}_{p^k}$, so $m = 2$ is a constant. $\square$

THEOREM 6.4. *All factorizations of f in $\mathbb{Z}_{p^k}[x]$ can be computed by Algorithm 5.1 in $O((n^\omega + n^2 m)M(k \log p))$ bit operations.*

Proof. The computation of the kernel of $\mathbf{K}$ over $\mathbb{Z}_{p^\mathbf{r}}$ can be computed by Storjohann's algorithm (Chapter 4, [10]) in $O(n^\omega M(\mathbf{r} \log p))$ bit operations. The result now follows from $\mathbf{r} < k$ and Theorems 5.3 and 6.3. $\square$

We note that if we take $\omega = 3$, the cost of the kernel computation is still the dominant cost, so the overall complexity of our factorization algorithm is $O(n^3 M(k \log p))$ in this case.

7. CONCLUSIONS AND FUTURE WORK

We have described an algorithm for factoring a polynomial f in $\mathbb{Z}_{p^k}[x]$, which can be used to factor polynomials in $\mathbb{Z}_N[x]$. Our algorithm improves on that of von zur Gathen and Hartlieb [6, 7] by processing all factors of a factorization at the same time and computing the kernel and determinants of matrices over $\mathbb{Z}_{p^k}$ more efficiently. Although our algorithms have been stated for polynomials over $\mathbb{Z}_N[x]$, they can easily be adapted to any coefficient ring R with the same properties as those stated in [6, 7]. For example, we can have $R = \mathbb{F}_q[y]$, the univariate polynomials over a finite field. The complexity results will be different, but our algorithm is still more efficient than that in [6, 7] because our algorithm performs no more operations than the first iteration of their algorithm.

One of the computational bottlenecks of our algorithm is the computation of the kernel of a structured matrix over $\mathbb{Z}_{p^\mathbf{r}}$. It would be of interest to compute the kernel by taking advantage of the matrix structure. Unfortunately, the triangular matrix obtained by the modified FFFG elimination algorithm does not appear to help in computing the kernel of $\mathbf{K}$ over $\mathbb{Z}_{p^\mathbf{r}}$, except when $\mathbf{r} = 1$ so that $\mathbb{Z}_{p^\mathbf{r}}$ is a field. A difficulty in taking advantage of the structure of $\mathbf{K}$ in kernel computation algorithms is that the order in which the columns are used as pivots is typically dictated by some variant of the extended Euclidean algorithm, and the order is not easy to predict in advance.

We would also like to investigate algorithms to obtain all factorizations in other non-unique factorization domains. In particular, we are interested in efficiently finding all factorizations over a differential polynomial domain.

8. ACKNOWLEDGEMENTS

The authors would like to thank the referees for their helpful suggestions and corrections.

9. REFERENCES

[1] Beckermann, B., and Labahn, G. Fraction-free computation of matrix rational interpolants and matrix GCDs. *SIAM J. Matrix Analysis and Applications 22*, 1 (2000), 114–144.

[2] Borevich, Z. I., and Shafarevich, I. R. *Number Theory*. Academic Press, 1966.

[3] Buchmann, J., and Neis, S. Algorithms for linear algebra problems over principal ideal rings. Tech. rep., Technische Hochschule Darmstadt, 1996.

[4] Chistov, A. L. Efficient factorization of polynomials over local fields. *Soviet Mathematics, Doklady 35*, 2 (1987), 430–433.

[5] Chistov, A. L. Algorithm of polynomial complexity for factoring polynomials over local fields. *Journal of Mathematical Sciences 70*, 4 (1994), 1912–1933.

[6] von zur Gathen, J., and Hartlieb, S. Factoring modular polynomials. In *International Symposium on Symbolic and Algebraic Computation* (1996), pp. 10–17.

[7] von zur Gathen, J., and Hartlieb, S. Factoring modular polynomials. *Journal of Symbolic Computation 26*, 5 (1998), 583–606.

[8] Geddes, K. O., Czapor, S. R., and Labahn, G. *Algorithms for computer algebra*. Kluwer Academic Publishers, 1992.

[9] Shamir, A. On the generation of polynomials which are hard to factor. In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing* (1993), pp. 796–804.

[10] Storjohann, A. *Algorithms for Matrix Canonical Forms*. PhD thesis, Department of Computer Science, Swiss Federal Institute of Technology—ETH, 2000.

[11] van der Waerden, B. L. *Algebra*, vol. I. Springer-Verlag, 1970.