Fast parallel computation of the Smith normal form of polynomial matrices

GILLES VILLARD
Institut IMAG – Laboratoire LMC
46, av. F. Viallet, F38031 Grenoble Cedex
Gilles.Villard@imag.fr

Abstract

We establish that the Smith normal form of a polynomial matrix in $F[x]^{n\times n}$, where F is an arbitrary commutative field, can be computed in $\mathcal{NC}_F$.

1 Introduction

Any matrix A over a principal ideal domain may be brought into the *Smith normal form* (SNF); the form is entirely computed within the domain of the entries and consists in a diagonalization of A. From a theoretical point of view [7] the form is well known and has many applications, both in the integer case [22] and in the polynomial case [7, 12].

From an algorithmic point of view, polynomial time algorithms are known for a sequential computation of the SNF with integer entries [17] or polynomial ones [16, 14, 23]. In parallel, finding a fast algorithm is still a difficult question in the integer case, since the problem relies on integer gcd computations. In the polynomial case, fast algorithms may be found in [14, 15, 10, 11] but they use random choices, so that the problem is only known to be in $\mathcal{RNC}^2$. However, the algorithm in [15] also computes unimodular transformation matrices for the form and does not suffer from exponential intermediate coefficient growth. Indeed, the authors have established, over "concrete fields" *e.g.* over the rational numbers, that there exist transformation matrices for the Smith form, whose entries have coefficients of length polynomial in the dimensions and coefficient lengths of the input matrices.

Concerning a closely related problem, *matrix similarity testing*, an ingenious solution is given in [24] and leads to a fast parallel deterministic algorithm showing that both similarity and non-similarity can be decided in $\mathcal{NC}^2$.

The main result of this paper avoids the random choices of the above algorithms and establishes a fast parallel deterministic algorithm for computing the Smith normal form of matrices which entries are in $F[x]$ where F is a commutative field. After some basic reminders in section 2, elements on matrix pencils (*i.e.* matrix pairs) lead us to consider the SNF as an application of the *Frobenius* and *Jordan normal forms*; using the results of [20, 21] concerning these latter problems, we give an algorithm in $\mathcal{NC}_F$ for computing the Smith normal form.

Our approach depends on the field F in the sense that we give a general algorithm running over any field but if the field satisfy some restrictive assumptions we propose a much simpler method. However both solutions are in $\mathcal{NC}_F$.

More precisely, when F is perfect and p-th roots can be taken, this is true for fields of characteristic zero and for finite fields, the matrix entries and the polynomial coefficients we manipulate are elements of F: this is presented in section 3. The problem of computing the Smith form reduces to the problem of computing the Frobenius form, the algorithm in [20] that gives a solution for this latter problem is directly applied.

If F is an arbitrary field, some intermediate steps are calculations over an algebraic extension of F. Using a parallel arithmetic on algebraic numbers as in [21] we show in section 4 that these computations reduce to computations in F. This parallel arithmetic is greatly inspired by the one proposed for the sequential *D5 system* [4, 5]. This system is implemented in AXIOM and some of its applications, especially form matrix normal forms may be found in [13]. From there we extend the algorithms in [21] to compute the Frobenius form over any field then to compute the Smith form.

We refer to [3] for the definitions of the boolean complexity classes $\mathcal{NC}$ and $\mathcal{RNC}$ of problems deterministically and probabilistically solvable by boolean circuits. In analogy with these classes von zur Gathen [9] has defined the classes $\mathcal{NC}_F$ and $\mathcal{RNC}_F$ of problems solvable by arithmetic circuits.

This paper focuses on non singular square matrices. They are no great difficulties to generalize our approach to the general case.

2 Basic concepts

We recall here some well known definitions and results concerning the Smith normal form. In the following, $A(x)$ is a non singular matrix of dimension n in $F[x]^{n\times n}$, with the degrees of the entries bounded by n, F being a commutative field. A matrix in $F[x]^{n\times n}$ is called *unimodular* if its determinant is a non zero element of F.

Definition 1 *[7]. A non singular matrix $S(x)$ in $F[x]^{n\times n}$ is in Smith normal form if it is diagonal, its diagonal entries are monic, each one dividing the next.*

ISAAC 94 - 7/94 Oxford England UK
© 1994 ACM 0-89791-638-7/94/0007..$3.50

Theorem 1 *[7]. Every non singular matrix $A(x)$ of $F[x]^{n\times n}$ is equivalent to a unique matrix $S(x)$ which is in Smith normal form: $S(x) = U(x)A(x)V(x)$, $U(x)$ and $V(x)$ unimodular. The non unity diagonal entries $s_i(x)$, $1 \le i \le \sigma$, of $S(x)$ (in the reverse order) are called the invariant factors of $A(x)$.*

In most sequential and parallel algorithms, the SNF is computed by repeated triangularization of the input matrix (by computing the *Hermite normal form*), but this usual approach is not appropriate to derive a fast parallel deterministic algorithm. Instead we will use the strong links that exist between the Smith form of special matrices and the Frobenius and Jordan forms.

Theorem 2 *[7]. Every matrix B of $F^{n\times n}$ is similar to a unique matrix C which is in Frobenius normal form (quasi-companion): $C = P_C^{-1}BP_C$. The polynomials associated to the companion blocks of C are the invariant factors of $A(x) = B - xI$ or equivalently of B.*

In other words, if the input matrix $A(x) \in F[x]^{n\times n}$ is such that there exists a constant matrix $B \in F^{n\times n}$ satisfying $A(x) = B - xI$, the computation of the Smith form of $A(x)$ reduces to computation of the Frobenius form of B. The main idea of our new algorithm is to use this result to compute the Smith form in the general case. The algorithm will bring $A(x)$ into an intermediate form for which such a matrix B does exist.

3 SNF over $F[x]$: F of characteristic zero or finite

When the field F is perfect and p-th roots can be taken, we have shown in [20] that the Frobenius form of a matrix can be computed in $\mathcal{NC}^2_F$. Since this algorithm makes use of the squarefree decomposition of polynomials in $F[x]$, the restrictive assumptions on F are necessary.

We proceed in two steps: at lemma 1, following [12], we first linearize $A(x)$ which entries are of degree n and reduce the study to a degree 1 matrix $\bar{A}_1x + \bar{A}_0$. Then, in theorem 3, we apply the study in [7] on matrix pencils, *i.e.* pairs $(\bar{A}_1, \bar{A}_0)$ of matrices, and compute a block-diagonal form for such pencils that gives the Smith form of $A(x)$. This second step deals with matrices of the type $B - xI$ on which we can use algorithms for the Frobenius form as announced previously.

Lemma 1 *Let $A(x)$ be a non singular matrix polynomial of degree n, $A(x) = \sum_{i=0}^{n} A_i x^i$ in $F[x]^{n\times n}$. The linearization of A into a degree 1 polynomial matrix, $\bar{A}(x) = \bar{A}_1x + \bar{A}_0$, with A_1 and A_0 constant matrices in $F^{n^2\times n^2}$ given by*

$$\bar{A} = \begin{bmatrix} I & 0 & \dots & 0 \\ 0 & \ddots & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & I & 0 \\ 0 & \dots & 0 & A_n \end{bmatrix} x + \begin{bmatrix} 0 & -I & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -I \\ A_0 & A_1 & \dots & A_{n-1} \end{bmatrix}$$

is such that the invariant factors of $\bar{A}$ are equal to the invariant factors of A.

Proof. Let $S(x)$ be the Smith form of $A(x)$. It can easily be shown that $\bar{A}(x)$ is equivalent to $diag(I, I, \dots, I, A(x))$ then to $\bar{S}(x) = diag(I, I, \dots, I, S(x))$. Since the diagonal entries of $\bar{S}(x)$ satisfy the divisibility property and from the unicity of the Smith form, $\bar{S}(x)$ is the Smith form of $\bar{A}(x)$. □

Theorem 3 *Let F be a commutative field of characteristic zero or a finite field, and $A(x)$ be a non singular matrix in $F[x]^{n\times n}$ with the degrees of the entries bounded by n. If F contains at least $n^2 + 1$ elements then the problem of computing the Smith normal form of $A(x)$ is in $\mathcal{NC}^2_F$, otherwise the problem is in $\mathcal{NC}^3_F$.*

Proof. For more details on the construction of normal forms for matrix pencils we refer to [7]. Applying lemma 1, the problems reduces to the problem of computing the Smith form of a n^2 by n^2 matrix $\bar{A}(x)$ of degree 1: $\bar{A}(x) = \bar{A}_1x+\bar{A}_0$. The purpose of the proof is to show that this latter problem reduces to Frobenius normal form computations.

Since $\bar{A}(x)$ is non singular, if $\#F > n^2$ there exist c in F so that $\det(\bar{A}_1c + \bar{A}_0) \neq 0$. If this is not the case, $q = \#F \le n^2$, we work in a field extension K of F containing $n^2 + 1$ elements, in particular, a convenient c is found in K. Let $U = \bar{A}_1c + \bar{A}_0$. Multiplying $\bar{A}(x)$ by U^{-1} on the left we obtain:

$$A^{(1)}(x) = U^{-1}\bar{A}(x) = (x - c)U^{-1}\bar{A}_1 + I.$$

Using similarity transformations, $U^{-1}\bar{A}_1$ can be brought into

$$C = \begin{bmatrix} C_0 & 0 \\ 0 & C_1 \end{bmatrix}$$

where C_0 and C_1 are quasi-companion matrices, C_0 corresponding to the null eigenvalues of $U^{-1}\bar{A}_1$ and C_1 corresponding to the non zero eigenvalues. The transformation is done by first computing the Frobenius form of $U^{-1}\bar{A}_1$, which gives the invariant factors $s_1(x), s_2(x), \dots, s_\sigma(x)$ of $U^{-1}\bar{A}_1$, and next, by computing the polynomials $s_i^*(x) = s_i(x)/x^{\sigma_i}$ where σ_i is the highest degree so that x^{σ_i} divides $s_i(x)$, for $1 \le i \le \sigma$. The matrix C_0 is quasi-companion of dimension $\sum_{i=1}^{\sigma} \sigma_i$, it consists of the companion blocks associated with the x^{σ_i}. In the same way, C_1 is the quasi-companion inversible matrix which blocks are the companion matrices associated to the $s_i^*(x)$. Then, if P is such that $P^{-1}(U^{-1}\bar{A}_1)P = C$, let

$$A^{(2)}(x) = P^{-1}A^{(1)}P = \begin{bmatrix} I + (x - c)C_0 & 0 \\ 0 & I + (x - c)C_1 \end{bmatrix}.$$

Notice that $I + (x - c)C_0$ is unimodular. Multiplying by C_1^{-1}, we finally get:

$$\begin{aligned} A^{(3)}(x) &= \begin{bmatrix} I & 0 \\ 0 & C_1^{-1} \end{bmatrix} A^{(2)}(x) \\ &= \begin{bmatrix} I + (x - c)C_0 & 0 \\ 0 & xI + C_1^{-1} - cI \end{bmatrix} \\ &= \begin{bmatrix} V(x) & 0 \\ 0 & xI - W \end{bmatrix}. \end{aligned}$$

Since only constant matrices have been used for the transformation of $\bar{A}(x)$, $\bar{A}(x)$ and $A^{(3)}(x)$ are equivalent matrices: they have the same Smith normal form. Furthermore, since $V(x)$ is unimodular, the Smith form of $A^{(3)}(x)$ is the Smith form of $xI - W$: applying theorem 2, the problem reduces to the computation of a Frobenius normal form. The Frobenius form of W gives the invariant factors and the Smith form of $\bar{A}(x)$ and of $A(x)$.

It remains to establish that the computation can be done in $\mathcal{NC}^2_K$. Once W is computed, we know from [20] that its Frobenius form so the Smith form of $A(x)$ can be obtained in $\mathcal{NC}^2_K$. To finish with making the proof, it is easy to see that

W can also be computed in $\mathcal{NC}^2_K$. The convenient value for c may be found by evaluating simultaneously $\det(A_1 c + A_0)$ for $c = 0, \ldots, n^2$. The matrix C is fast computed also from [20], and for the matrix inverses U^{-1} and C_1^{-1} we use the algorithm in [2].

Now if $\#F > n^2$ then $K = F$ and the problem is in $\mathcal{NC}^2_F$.

Otherwise, if $q = \#F \leq n^2$, let F be of characteristic p, the elements of the algebraic extension K are represented as polynomials of degree less than $\lceil \log_q(n^2+1) \rceil$ over F. To apply the $\mathcal{NC}^2_K$ algorithm in [20] we need to compute multiplications, divisions and p-th roots in K (the algorithm relies on the $\mathcal{NC}^2_K$ squarefree decomposition of polynomials over K given in [8]). Each arithmetic operation in K requires $O(\log\log_q n)$ operations in F, and from [6] a p-th root can be computed in $O(\log^2(\log_q n)+\log(q))$ operations in F. Hence the total parallel time is $O(log^2(n)(\log^2(\log_q n) + \log(q)))$ over F. As announced the problem is in $\mathcal{NC}^3_F$. □

Since it appears to be quite huge, we will now spend only a few words on the *processor demand* of our algorithm. If $O(M(n))$ and $O(P(n))$ operations are sufficient to respectively multiply two $n \times n$ matrices and two polynomials of degree n over F, the overall demand of the algorithm in [20] for the Frobenius form is $O(n^5 M(n))$. The computation of the Smith form consequently requires $O(n^{10} M(n^2))$ processors if $\#F > n^2$ and $O(n^{10} M(n^2) P(\log n))$ otherwise.

The results we have given so far are all in terms of counting operations in F. In addition, for "concrete fields", our algorithm does not suffer from exponential coefficient-size blow-up. Let us assume for example that the input matrix $A(x)$ has polynomial entries with the degrees bounded by n and which coefficients are rational numbers with lengths bounded by n. We have seen that computing the Smith form of $A(x)$ requires to compute matrix inverses and Frobenius forms of matrices with rational entries: their sizes clearly depend polynomially on n. Since the algorithm in [20] for the Frobenius form runs in three steps involving only operations (characteristic polynomials, squarefree decompositions and gcd) on polynomials which coefficients are entries of the input matrix, we deduce that our computation of the Smith form involves only elements of lengths polynomial in n.

As said previously, the algorithm of this section has a main drawback: since it relies on the algorithm in [20], it makes use of squarefree decomposition of polynomials and requires assumptions on the ground field F. Next section intends to show how this can be avoided.

4 SNF over $F[x]$: F any field

The Smith form of a matrix $A(x)$ will be computed as above from the Frobenius form of constant matrices. For the Frobenius form itself, no fast parallel algorithm running over any field was known. We are going to show that the algorithm in [21], which directly computes the Jordan form, may be used: from the entries of the Jordan form that lie in an algebraic extension of F, the entries of the Frobenius form in F can be recovered. Two main difficulties have to be bypassed:

(i). We are going to deal with algebraic numbers represented as roots of polynomials over F. Unfortunately since squarefree decomposition is not available, some expressions involving such numbers may be hard to simplify. This will occur for the computation of the invariant factors if a naïve method is used.

(ii). The Frobenius form of a matrix over F has its entries in F. These entries will be obtained from the Jordan form as expressions involving algebraic numbers over F. We will have to prove that their representations in F can be recovered from their representations in the given algebraic extension of F.

We begin in section 4.1 with some reminders about the Jordan form, and we explain how it produces, in a natural way, the invariant factors and the Frobenius form. Then we illustrate the point (i) above and explain why, from a practical point of view, a more elaborated approach has to be found. The method we propose is described in section 4.2, it leads to a $\mathcal{NC}^2_F$ algorithm for the Frobenius form that is presented in section 4.3. In particular a solution is given for the problem (ii) above. The final result, concerning the Smith normal form is given in section 4.4.

4.1 The Jordan form and the invariant factors

Any matrix B in $F^{n\times n}$ is similar to a unique (up to permutation) block-diagonal matrix J in Jordan form *i.e.* which diagonal blocks are matrices of the form [7]:

$$J_k(\lambda_j) = \begin{bmatrix} \lambda_j & 1 & \ldots & 0 \\ 0 & \lambda_j & \ddots & 0 \\ \vdots & & \ddots & 1 \\ 0 & \ldots & 0 & \lambda_j \end{bmatrix} \in F^{k\times k},$$

where λ_j is an eigenvalue of B; J_k is a $k \times k$ banded matrix, which is called a k-Jordan block associated with λ_j. Each block $J_k(\lambda_j)$ corresponds to an *elementary divisor* $(x-\lambda_j)^k$ of B.

From the Jordan form J and from a theoretical point of view, the $s_i(x)$ (the invariant factors in the reverse order) of B are easily computed: if the Jordan blocks associated with any eigenvalue λ_j are numbered from 1 to n_j by increasing dimensions, then the elementary divisor $(x-\lambda_j)^{\gamma_{j,i}}$ is a divisor of $s_i(x)$ if and only if $\gamma_{j,i}$ is the dimension of the i-th block associated with λ_j.

From a computational point of view, the situation is more complex. Indeed, when the field F is not algebraically closed, the eigenvalues of B lie in an algebraic extension of F, and in general, the standard Jordan form cannot be computed. But we may always compute a *symbolic Jordan form* $\tilde{J}$: this form gives the structure of J using symbols that take the place of the eigenvalues [15, 9]. Each symbol $\tilde{\lambda}_j$ is a *generalized eigenvalue*, it is associated with a polynomial $\Lambda(\lambda)$ in $F[\lambda]$, with the understanding that Λ is a representation of λ_j, *i.e.* $\Lambda(\lambda_j) = 0$. This representation is a factor of the characteristic polynomial of B.

For a given matrix, many choices are consistent with this. The symbolic form coincides with the "true" Jordan form if the eigenvalues are known, *i.e.* if the representations are linear factors $(\lambda - \lambda_i)$. The representations could be chosen irreducible but this would lead to restrictive assumptions since we do not know how to factor polynomials fast in parallel [9]. The representations could be chosen squarefree but again it is not a reasonable choice in order to work over arbitrary fields. However such assumptions are not necessary to compute the structure of the Jordan form and the Smith form: in the following our representations will simply be products of elementary divisors of fixed exponent.

Now, if we apply the previous construction to compute the invariant factors from the symbolic Jordan form, we will obtain only symbolic representations of them. Since we work over any field and since squarefree decomposition of polynomials is not available, these representations will not allow to find the invariant factors in $F[x]$ as in [20] for fields of characteristic zero or finite fields. A simple illustration of this fact is given by example 1.

Example 1. let $\tilde{J}$ be a symbolic Jordan form:

$$\tilde{J} = \begin{bmatrix} \lambda_1 & 1 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & \lambda_2 & 1 \\ 0 & 0 & 0 & \lambda_2 \end{bmatrix}, \quad \Lambda(\lambda) = (\lambda^2+1)^3.$$

In other words, $\tilde{J}$ corresponds to a block with two eigenvalues λ_1 and λ_2 represented by the generalized eigenvalue $(\lambda^2+1)^3 = 0$.

Since each eigenvalue is associated with only one Jordan block, $\tilde{J}$ gives rise to only one invariant factor: $s_1(x) = (x^2+1)^2$. From the dimensions of the Jordan blocks, it is easy to give a symbolic representation of $s_1(x)$: $s_1(x) = ((x-\lambda_1)(x-\lambda_2))^2$ with $\Lambda(\lambda_1) = 0$ and $\Lambda(\lambda_2) = 0$. But there is no simple way to simplify this latter expression if the squarefree decomposition of polynomials is not available and especially if roots of $(\lambda^2+1)^3$ cannot be taken.

The solution we propose is to explicitly construct a transformation P_C for the Frobenius form C (*i.e.* $C = P_C^{-1} B P_C$) from a transformation $P_{\tilde{J}}$ for the symbolic Jordan form (*i.e.* $\tilde{J} = P_{\tilde{J}}^{-1} B P_{\tilde{J}}$).

4.2 From the Jordan form to the Frobenius form

The construction we use to obtain a transformation for the Frobenius form from a transformation for the Jordan form may be found in [7]. Let B be a matrix in $F^{n\times n}$ and J its Jordan form. Any transformation P_J for the Jordan form has a well known structure.

Definition 2 *Let P_J be a transformation for the Jordan form* i.e. *P_J is such that $J = P_J^{-1} B P_J$. Let $[j_1, j_2, \ldots, j_k]$ be k consecutive columns of J corresponding to a k-Jordan block associated with any eigenvalue λ_j of B. The corresponding columns $[p_1, p_2, \ldots, p_k]$ of P_J constitute a length k* Jordan chain *associated with λ_j. It satisfies:*

$$(B-\lambda_j I)p_1 = 0, \quad (B-\lambda_j I)p_l = p_{l-1}, \ 2 \le l \le k.$$

The vector p_k is called end of the Jordan chain, *it is such that:*

$$\begin{cases} p_k \in Ker(B-\lambda I)^k, \ p_k \notin Ker(B-\lambda I)^{k-1}, \\ p_k \notin Range(B-\lambda I), \ 1 \le l \le \delta. \end{cases}$$

The two first relations ensure that starting from p_k, the chain is exactly of length k, the Range condition gives that the chain is not included in a longer chain. In addition, the minimal polynomial of p_k is an elementary divisor $(x-\lambda_j)^k$.

Next lemma is based on classic results, we omit its proof.

Lemma 2 *For any invariant factor $s_i(x)$ of B, if the vectors $p_1^{(i)}, p_2^{(i)}, \ldots, p_\delta^{(i)}$ are ends of Jordan chains associated with the Jordan blocks corresponding to $s_i(x)$, then $s_i(x)$ is the product of the minimal polynomials of the $p_l^{(i)}$, $1 \le l \le \delta$.*

This lemma consequently yields a desired transformation P_C from the matrix B to its Frobenius normal form. Any companion block of the Frobenius form corresponds to an $s_i(x)$ (theorem 2). The associated columns of the transformation P_C are built by iterating a cyclic vector $p^{(i)}$ *i.e.* a vector which minimal polynomial is $s_i(x)$. Applying lemma 2, such a vector can be obtained from the corresponding ends of chains, taking for instance $p^{(i)} = p_1^{(i)} + p_2^{(i)} + \ldots + p_\delta^{(i)}$. If $s_i(x)$ is of degree d_i, this gives d_i columns of P_C: $[p^{(i)}, Bp^{(i)}, B^2p^{(i)}, \ldots, B^{d_i-1}p^{(i)}]$.

In the next section, we extend these arguments to the computation of the Frobenius form from a symbolic Jordan form.

4.3 A fast parallel algorithm for the Frobenius form

From ([21], theorem 6) a symbolic Jordan form $\tilde{J}$ of a matrix B in $F^{n\times n}$ and a corresponding transformation are computed in $\mathcal{NC}_F^2$.

The symbolic form gives us the structure of the Jordan form, *i.e.* the number of blocks and their dimensions, and gives representations for the generalized eigenvalues. Indeed, as said previously, since in general the eigenvalues cannot be computed, we deal with algebraic numbers given by generalized eigenvalues, *i.e.* eigenvalues belonging to the same factors in a partial factorization of the characteristic polynomial of B. Following [4], these generalized eigenvalues are represented as polynomials in $F[\lambda]$, the polynomials $J_\delta^{(k)}(\lambda)$ below. In particular, unlike in ([21], theorem 7), these polynomials need not to be squarefree or even relatively prime, and we do not have to pay for a corresponding additional cost of $\log n$. Our representations are products of elementary divisors of fixed exponent.

More precisely, applying ([21], theorem 6), for any fixed δ and k, $1 \le \delta, k \le n$, we compute:

- a polynomial $J_\delta^{(k)}(\lambda)$ in $F[\lambda]$ which roots λ_j are the eigenvalues of B associated with exactly δ Jordan blocks of dimension k: associated with δ elementary divisors $(\lambda-\lambda_j)^k$.

Since we know that the problem of computing a nullspace basis over generalized eigenvalues is in $\mathcal{NC}_F^2$ ([21], proposition 4), we also easily compute the ends of the Jordan chain giving a transformation matrix for the symbolic Jordan form. For any fixed δ and k as above we get:

- δ ends of Jordan chains $p_1^{(k,\delta)}(\lambda), p_2^{(k,\delta)}(\lambda), \ldots, p_\delta^{(k,\delta)}(\lambda)$.

These vectors are computed using:

$$\begin{cases} p_l^{(k,\delta)} \in Ker(B-\lambda I)^k, \ p_l^{(k,\delta)} \notin Ker(B-\lambda I)^{k-1}, \\ p_l^{(k,\delta)} \notin Range(B-\lambda I), \ 1 \le l \le \delta, \ with \ J_\delta^{(k)}(\lambda) = 0. \end{cases}$$

Given k and δ, for all l, the $p_l^{(k,\delta)}(\lambda)$ are vectors which entries are combinations of the eigenvalues represented by the polynomial $J_\delta^{(k)}(\lambda)$; they are represented as polynomials in $F[\lambda]/(J_\delta^{(k)}(\lambda))$.

From here we are going to use an additional variable x to avoid any confusion between the invariant factors and the representations of the eigenvalues. To compute the Frobenius form of B, we proceed in two steps: first, for each $s_i(x)$, we isolate the associated ends of chains; then, applying lemma 2, we construct an appropriate transformation.

Lemma 3 *Let B be a matrix in $F^{n\times n}$. Let the structure of the Jordan form of B be given by the polynomials $J_\delta^{(k)}(\lambda)$, $1 \le \delta, k \le n$, and let $p_l^{(k,\delta)}(\lambda)$, $1 \le i \le \delta$, be the corresponding ends of Jordan chains. The ends of chains associated with each $s_i(x)$ can be computed in $\mathcal{NC}_F^2$ (the $s_i(x)$ are the invariant factors in the reverse order).*

Proof. In section 4.1 we have seen that if the Jordan blocks are numbered by increasing dimensions, then $s_i(x)$ is computed from the i-th blocks. From [21], as the $J_\delta^{(k)}(\lambda)$, polynomials $B_j^{(k)}(\lambda)$ which roots are the eigenvalues of B that are associated with exactly j blocks of dimensions strictly lower than k can be computed in $\mathcal{NC}_F^2$. We take $B_j^{(k)}(\lambda) = 0$ if there is no block of dimension strictly lower than k. Now, let

$$I_i^{(k,\delta)}(\lambda) = \gcd\left(B_{i-1}^{(k)}(\lambda), J_\delta^{(k)}(\lambda)\right), \quad 1 \le i,k,\delta \le n.$$

By construction, the roots of $I_i^{(k,\delta)}(\lambda)$ have $i-1$ blocks of dimensions strictly lower than k and δ blocks of dimension k. In other words the roots of $I_i^{(k,\delta)}(\lambda)$ are the eigenvalues λ_j for which the elementary divisor $(x-\lambda_j)^k$ is a divisor of $s_i(x), s_{i+1}(x), \ldots, s_{i+\delta-1}(x)$. From there, the corresponding ends of chains are simply the vectors $p_l^{(k,\delta)}(\lambda) \bmod I_i^{(k,\delta)}(\lambda)$. We conclude the proof by noticing that the $I_i^{(k,\delta)}(\lambda)$ are computed simultaneously in $\mathcal{NC}_F^2$ using the algorithm in [2] for the gcd of polynomials. □

The Frobenius form is now directly computed by applying lemma 2 in this new context.

Lemma 4 *For F a commutative field and a matrix B in $F^{n\times n}$, to compute the Frobenius form of B is in $\mathcal{NC}_F^2$.*

Proof. Since the symbolic Jordan form and the Jordan chains are computed over an algebraic extension of F, the main point of the proof is to verify that the Frobenius form or equivalently the invariant factors, that are quantities over F, can be recovered.

For any target invariant factor $s_i(x)$ of B, let the associated ends of chains be $p_1^{(i)}, p_2^{(i)}, \ldots, p_\delta^{(i)}$. Let the entries of $p_l^{(i)}$, $1 \le l \le \delta$, be polynomials in a generalized eigenvalue represented by $\Lambda_l(\lambda)$ in $F[\lambda]$. They are computed by lemma 3. Then define

$$p^{(i)}(\lambda) = \left(\prod_{l\neq 1}\Lambda_l(\lambda)\right) p_1^{(i)}(\lambda) + \ldots + \left(\prod_{l\neq k}\Lambda_l(\lambda)\right) p_\delta^{(i)}(\lambda).$$

By construction, for any eigenvalue λ_j root of $s_i(x)$, the vector $p^{(i)}(\lambda_j)$ is an end of a Jordan chain giving the Jordan block associated to λ_j which corresponds to $s_i(x)$. From lemma 2, we know that $s_i(x)$ is the product of the minimal polynomials of the $p^{(i)}(\lambda_j)$ for λ_j root of $s_i(x)$.

Now, we view the input matrix B as a matrix over the field $F(\lambda)$ of the polynomial fractions in λ over F. The Frobenius form so the invariants factors of B remain the same. Over $F(\lambda)$, we may consider the minimal polynomial $\pi_i(x)$ of the vector $p^{(i)}(\lambda)$. From standard properties of the minimal polynomial of a vector, see [7] for instance, $\pi_i(x)$ is a multiple of the minimal polynomials of the $p^{(i)}(\lambda_j)$ for all λ_j root of $s_i(x)$. Consequently, using lemma 2 as noticed above, $\pi_i(x)$ is a multiple of $s_i(x)$.

Following section 4.2, we may now construct a transformation $P_{\bar{C}}$ over $F(\lambda)$ to bring B into a form $\bar{C}(\lambda)$, very close to the Frobenius form. If $s_i(x)$ is of degree d_i, $p^{(i)}(\lambda)$ considered as a cyclic vector gives us d_i columns of $P_{\bar{C}}(\lambda)$ corresponding to the companion block associated with $s_i(x)$: $[p^{(i)}, Bp^{(i)}, B^2p^{(i)}, \ldots, B^{d_i-1}p^{(i)}]$. Since the minimal polynomial of $p^{(i)}(\lambda)$ is a multiple of $s_i(x)$, we have $\bar{C}(\lambda) = P_{\bar{C}}^{-1} B P_{\bar{C}}$:

$$\bar{C}(\lambda) = \begin{bmatrix} C_{s_1(x)} & B_1^{(2)}(\lambda) & B_1^{(3)}(\lambda) & \ldots & B_1^{(\sigma)}(\lambda) \\ 0 & C_{s_2(x)} & B_2^{(3)}(\lambda) & \ldots & B_2^{(\sigma)}(\lambda) \\ 0 & 0 & C_{s_3(x)} & \ldots & B_3^{(\sigma)}(\lambda) \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & \ldots & C_{s_\sigma(x)} \end{bmatrix},$$

where $C_{s_i(x)}$ is the companion matrix associated with $s_i(x)$ (see example 2 below). By construction, $\bar{C}$ is in polycyclic form (see [18, 19]), *i.e.* each matrix $B_j^{(l)}$ is zero except its last column. From such a form, a transformation toward the Frobenius form could be computed [18, 11], however, this is not currently our purpose since the form itself is already obtained.

It remains to get convinced that the computations can be done in $\mathcal{NC}_F^2$.

This is clear for the simultaneous computations of the $p^{(i)}(\lambda)$ and of $P_{\bar{C}}$ using elementary operations on polynomials and matrices. Using the algorithm in [1] for the inversion of a matrix in one indeterminate, we invert $P_{\bar{C}}$. Then $\bar{C}$ and C the Frobenius form of B are easily calculated. □

Example 2. Consider a matrix

$$B = \begin{bmatrix} 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

whose a symbolic Jordan form is

$$\tilde{J} = \begin{bmatrix} \lambda_1 & 1 & 0 & 0 \\ 0 & \lambda_1 & 0 & 0 \\ 0 & 0 & \lambda_2 & 1 \\ 0 & 0 & 0 & \lambda_2 \end{bmatrix}, \quad \Lambda(\lambda) = (\lambda^2+1)^2.$$

As above, only one invariant factor $s_1(x)$ is involved, associated with it we may take

$$p_1^{(1)}(\lambda) = \begin{bmatrix} -\lambda^3 \\ 1 \\ \lambda \\ -\lambda^2 \end{bmatrix}$$

as end of Jordan chain. The entries of $p_1^{(1)}(\lambda)$ are polynomials in the generalized eigenvalue $(\lambda^2+1)^2 = 0$.

To compute the Frobenius form of B it now suffices to construct an appropriate transformation by iterating $p_1^{(1)}(\lambda)$ as a cyclic vector: $P_{\bar{C}}(\lambda) = [p_1^{(1)}, Bp_1^{(1)}, B^2p_1^{(1)}, B^3p_1^{(1)}]$,

$$P_{\bar{C}}(\lambda) = \begin{bmatrix} -\lambda^3 & 1+2\lambda^2 & \lambda+2\lambda^3 & -3\lambda^2-2 \\ 1 & \lambda & \lambda^2 & \lambda^3 \\ \lambda & \lambda^2 & \lambda^3 & -1-2\lambda^2 \\ -\lambda^2 & -\lambda^3 & 1+2\lambda^2 & \lambda+2\lambda^3 \end{bmatrix}.$$

And the polycyclic form $\bar{C} = P_{\bar{C}}^{-1}(\lambda)BP_{\bar{C}}(\lambda)$ is directly obtained, here it coincides with the Frobenius form:

$$C = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Observe that this last step involves only standard computations over the polynomial fractions in λ and leads to constant companion matrices.

4.4 A fast parallel algorithm for the Smith form

From lemma 4 and from arguments similar to those used for theorem 3, our final result follows immediately.

Theorem 4 *For F a commutative field and a non singular matrix $A(x)$ in $F[x]^{n\times n}$ with the degrees of the entries bounded by n, the problem of computing the Smith normal form of $A(x)$ is in $\mathcal{NC}_F^2$ if F contains at least n^2+1 elements, and is in $\mathcal{NC}_F^3$ otherwise.*

5 Conclusion

We have have provided a parallel algorithm to compute the Smith normal form of polynomial matrices over an arbitrary fiel F. Our algorithm is of main interest since it shows that the problem can be solved deterministically in $\mathcal{NC}_F$. Two questions remain opened: how to compute the unimodular transformation matrices and how to avoid intermediate calculus in an algebraic extension of F ?

References

[1] A. Borodin, S.A. Cook, and N. Pippenger. Parallel computation for well-endowed rings and space bounded probalistic machines. *Information and Control*, 58:113–136, 1983.

[2] A. Borodin, J. von zur Gathen, and J. Hopcroft. Fast parallel matrix and gcd computations. *Information and Control*, 52:241–256, 1982.

[3] S.A. Cook. A taxonomy of problems with fast parallel algorithms. *Inf. Control*, 64:2–22, 1985.

[4] J. Della Dora, C. Dicrescenzo, and D. Duval. About a new method for computing in algebraic number fields. In *Proc. EUROCAL'85*, LNCS 204, Springer Verlag, pages 289–290, 1985.

[5] C. Dicrescenzo and D. Duval. Algebraic extensions and algebraic closure in Scratchpad II. In *Proc. ISSAC'88*, LNCS 358, Springer Verlag, pages 440–446, 1988.

[6] F.E. Fich and M. Tompa. The parallel complexity of exponentiating polynomials over finite fields. In *Proc. 17th Annual ACM Symp. Theory Comp.*, 1985.

[7] F.R. Gantmacher. *Théorie des matrices.* Dunod, Paris, France, 1966.

[8] J. von zur Gathen. Parallel algorithms for algebraic problems. *SIAM J. Comp.*, 13:802–824, 1984.

[9] J. von zur Gathen. Parallel arithmetic computations: a survey. In *Proc. 12th Int. Symp. Math. Found. Comput. Sci., Bratislava*, pages 93–112. LNCS 233, Springer Verlag, 1986.

[10] M. Giesbrecht. *Nearly optimal algorithms for canonical matrix forms* . PhD thesis, Department of Computer Science, University of Toronto, 1993.

[11] M. Giesbrecht. Nearly optimal algorithms for canonical matrix forms. *SIAM Journal on Computing*, 1994. To appear.

[12] I. Gohberg, P. Lancaster, and L. Rodman. *Matrix polynomials.* Academic Press, New York, 1982.

[13] T. Gómez-Díaz. *Quelques applications de l'évaluation dynamique* . PhD thesis, Université de Limoges, France, 1994.

[14] E. Kaltofen, M.S. Krishnamoorthy, and B.D. Saunders. Fast parallel computation of Hermite and Smith forms of polynomials matrices. *SIAM J. Alg. Disc. Meth.*, 8 4, pp 683-690, 1987.

[15] E. Kaltofen, M.S. Krishnamoorthy, and B.D. Saunders. Parallel algorithms for matrix normal forms. *Linear Algebra and its Applications*, 136:189–208, 1990.

[16] R. Kannan. Solving systems of linear equations over polynomials. *Theoretical Computer Science*, 39:69–88, 1985.

[17] R. Kannan and A. Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM J. Comput.*, 8 4, pp 499-507, 1979.

[18] W. Keller-Gehrig. Fast algorithms for the characteristic polynomial. *Theoretical Computer Science*, 36:309–317, 1985.

[19] P. Ozello. *Calcul exact des formes de Jordan et de Frobenius d'une matrice.* PhD thesis, Université Scientifique et Médicale de Grenoble, France, 1987.

[20] J.L. Roch and G. Villard. Fast parallel computation of the Jordan normal form of matrices. *Parallel Processing Letters*, 1994. To appear.

[21] J.L. Roch and G. Villard. Parallel computations with algebraic numbers, a case study: Jordan normal form of matrices. In *Parallel Architectures and Languages Europe 94, Athens Greece*, LNCS, July 1994.

[22] A. Schrijver. *Theory of linear and integer programming.* Wiley-Interscience series in Discrete Mathematics, 1986.

[23] G. Villard. Computation of the Smith normal form of polynomial matrices. In *International Symposium on Symbolic and Algebraic Computation, Kiev, Ukraine.* ACM Press, pp 209–217, July 1993.

[24] Y. Zalcstein and M. Garzon. An NC^2 algorithm for testing similarity of matrices. *Information Processing Letters*, 30:253–254, 1989.