Processor Efficient Parallel Solution of Linear Systems over an Abstract Field*

*Erich Kaltofen***

Department of Computer Science, University of Toronto
Toronto, Canada M5S 1A4; Inter-Net: kaltofen@cs.toronto.edu

Victor Pan

Department of Mathematics and Computer Science
Lehman College, City University of New York
Bronx, New York 10468; Inter-Net: vpan@lcvax.bitnet

Parallel randomized algorithms are presented that solve n-dimensional systems of linear equations and compute inverses of $n \times n$ non-singular matrices over a field in $O((\log n)^2)$ time, where each time unit represents an arithmetic operation in the field generated by the matrix entries. The algorithms utilize within a $O(\log n)$ factor as many processors as are needed to multiply two $n \times n$ matrices. The algorithms avoid zero divisions with controllably high probability provided the $O(n)$ random elements used are selected uniformly from a sufficiently large set. For fields of small positive characteristic, the processor count measures of our solutions are somewhat higher.

1. Introduction

A processor efficient parallel algorithm is a parallel algorithm that has a running time that is poly-logarithmic in the input size and that utilizes asymptotically as many processors as the best known sequential step count for solving the problem; a poly-logarithmic factor in the asymptotic processor count is allowed (Karp and Ramachandran 1990). This paper considers the problem of solving a linear system of n equations with n unknowns over an abstract field, as well as the closely related problems of computing the inverse, determinant, and rank of an $n \times n$ matrix. An individual step in our algorithms is an addition, subtraction, multiplication, division, or zero-test of elements in the field that the entries of the linear system generate. Gaussian elimination is a sequential method for all these computational problems over abstract fields, whose running time can be asymptotically related to the sequential complexity of $n \times n$ matrix multiplication (Bunch and Hopcroft 1974). We present processor efficient randomized parallel algorithms for solving non-singular systems and for inverting non-singular matrices.

Csanky (1976) used Leverrier's approach to devise a parallel linear system solver, but the best processor count known for this approach exceeds by a factor of almost $\sqrt{n}$ the complexity of matrix multiplication (Preparata and Sarwate 1978), (Galil and Pan 1989). Leverrier's algorithm does not work for fields whose characteristic is positive and less than n, in which case the best known parallel algorithms needed by a factor of n more processors (Berkowitz 1984), (Chistov 1985). All previous parallel solutions compute the characteristic polynomial of the coefficient matrix without divisions. For this restricted algebraic model these algorithms are processor optimal, i.e., it appears not to be known how

*This material is based on work supported in part by the National Science Foundation under Grant No. CCR-90-06077 and under Grant No. CDA-88-05910 (first author), and under Grant No. CCR-88-05782 and Grant No. CCR-90-20690 and by the PSC CUNY Awards #661340 and #669290 (second author).

**Permanent address: Department of Computer Science, Rensselaer Polytechnic Institute, Troy, New York 12180-3590; Inter-Net: kaltofen@cs.rpi.edu.

In Proc. 3rd Ann. ACM Symp. Parallel Algor. Architectures, ACM Press (July 1991), pp. 180–191.

to compute the determinant of a matrix faster sequentially avoiding divisions.

Our processor efficient solution is not division free, but our algorithms realize shallow algebraic circuits and thus have no zero-tests. In order to avoid a division by zero, we instead randomly perturb the coefficient matrix. Our solution requires $O(n)$ random field elements, and we prove that if these elements are taken uniformly from a set containing s field elements, the probability of a zero-division on a non-singular input is no more than $3n^2/s$. In a substep our algorithm uses Leverrier's method. It thus has the same restriction on the characteristic of the field as does Csanky's solution, namely it divides by $2, 3, \ldots, n$. It should be noted that our solution uses matrix multiplication as a black-box. Therefore, the processor count and especially the constant in the big-O estimate is directly related to the particular matrix multiplication algorithm used, and for the classical method may yield a practical algorithm.

Our algorithms combine several advances in algebraic computational complexity theory. The first is the field independent randomized method for solving sparse linear systems by Wiedemann (1986). That approach provides a randomized parallel processor efficient reduction of linear system solving to the Berlekamp/Massey problem for finding a linear generator of a linear recurrence. Further reduction is possible to solving a non-singular Toeplitz system. The second advance is solving such Toeplitz systems in parallel processor-efficiently. Putting these two approaches together yields processor efficient randomized parallel algorithms for computing solutions to non-singular systems and for computing the determinant of a matrix. In order to obtain the inverse from the determinant of a non-singular matrix we employ the simple but ingenious reduction by Baur and Strassen (1983). Our contribution is to prove that this reduction can be realized without increasing the parallel time by more than a constant factor. We finally present one more application of that reduction, which relates the complexity of solving non-singular systems to the complexity of solving the transposed systems.

The methods presented here also yield randomized parallel processor efficient algorithms for the problems of computing the rank of a matrix, a single solution of a singular linear system, the basis for the nullspace of a matrix, and a least-squares solution of a linear system. We discuss these extensions briefly in the last section.

Notation: By $\omega > 2$ we denote the exponent of the matrix multiplication algorithm used, which must for dimension n yield a circuit that is simultaneously of depth $O(\log n)$ and size $O(n^\omega)$; currently the best known exponent is $\omega < 2.3755$ (Coppersmith and Winograd 1990). By $S^{\mathbb{N}}$ we denote the set of all infinite sequences $\{a_i\}_{i=0}^{\infty}$, $a_i \in S$. Finally, the function $\log x$, $x \geq 2$, denotes $\log_2(x)$, and the function $\operatorname{loglog} x$, $x \geq 4$, denotes $\log_2(\log_2(x))$.

2. Wiedemann's Method

Wiedemann (1986) presents a randomized Las Vegas algorithm for computing the determinant of a sparse matrix over a finite field. As it turns out, this method is a field independent algorithm that reduces the problem of computing the determinant to the problem of solving a non-singular Toeplitz system. The reduction is processor efficient and of poly-logarithmic parallel time complexity. In the following we present Wiedemann's argument with a slight change in the probabilistic analysis, which is warranted because we work over an abstract field.

Let V be a vector space over the field K, and let $\{a_i\}_{i=0}^{\infty}$ be an infinite sequence with elements $a_i \in V$. The sequence $\{a_i\}_{i=0}^{\infty}$ is *linearly generated* over K if there exist $c_0, c_1, \ldots, c_n \in \mathsf{K}$, $n \geq 0$, $c_k \neq 0$ for some k with $0 \leq k \leq n$, such that

$$\forall j \geq 0 \colon c_0 a_j + \cdots + c_n a_{j+n} = 0.$$

The polynomial $c_0 + c_1\lambda + \cdots + c_n\lambda^n$ is called a *generating polynomial* for $\{a_i\}_{i=0}^{\infty}$. The set of all generating polynomials for $\{a_i\}_{i=0}^{\infty}$ together with the zero polynomial forms an ideal in $\mathsf{K}[\lambda]$. The unique polynomial generating that ideal, normalized to have leading coefficient 1, is called the *minimum polynomial* of a linearly generated sequence $\{a_i\}_{i=0}^{\infty}$. Every generating polynomial is a multiple of the minimum polynomial.

Let $A \in \mathsf{K}^{n\times n}$ be a square matrix over a field. The sequence $\{A^i\}_{i=0}^{\infty} \in (\mathsf{K}^{n\times n})^{\mathbb{N}}$ is linearly generated, and its minimum polynomial is the minimum polynomial of A, which will be denoted by f^A. For any column vector $b \in \mathsf{K}^n$, the sequence $\{A^i b\}_{i=0}^{\infty} \in (\mathsf{K}^n)^{\mathbb{N}}$ is also linearly generated by f^A. However, its minimum polynomial, denoted by $f^{A,b}$, can be a proper divisor of f^A. For any row vector $u \in \mathsf{K}^{1\times n}$, the sequence $\{uA^i b\}_{i=0}^{\infty} \in \mathsf{K}^{\mathbb{N}}$ is linearly generated as well, and its minimum polynomial, denoted by $f_u^{A,b}$, is again a divisor of $f^{A,b}$. Wiedemann proves the following fact (loc. cit., §VI).

Theorem 1. *Let $m = \deg(f^{A,b})$, and let W be the linear space of polynomials of degree less than m in $\mathsf{K}[\lambda]$. There exists a surjective linear map $\ell \colon \mathsf{K}^{1\times n} \longrightarrow W$ such that*

$$\forall u \in \mathsf{K}^{1\times n} \colon f_u^{A,b} = f^{A,b} \iff \operatorname{GCD}(f^{A,b}, \ell(u)) = 1.$$

Clearly, the sequence $\{uA^i\}_{i=0}^{\infty} \in (\mathsf{K}^{1\times n})^{\mathbb{N}}$ is the symmetric counterpart of $\{A^i b\}_{i=0}^{\infty}$. We write $f^{u,A}$ for the minimum polynomial of the former. By considering the rational canonical form of A, one establishes the existence of a row vector $u_0 \in \mathsf{K}^{1\times n}$ with $f^{u_0,A} = f^A$. Let

$m' = \deg(f^A)$ and let W' be the linear subspace of $\mathsf{K}[\lambda]$ spanned by $\{1, \lambda, \ldots, \lambda^{m'-1}\}$ over K. By Theorem 1 it follows that there exist surjective linear maps

$$\ell'_{u_0} : \mathsf{K}^n \longrightarrow W^{m'}$$

such that

$$\forall b \in \mathsf{K}^n : f_{u_0}^{A,b} = f^{u_0,A} = f^A \iff \mathrm{GCD}(f^A, \ell'_{u_0}(b)) = 1.$$

Thus, the probability that $f_u^{A,b} = f^A$ for randomly selected vectors u and b is essentially the probability of randomly selecting two polynomials of degree less than m' that are both relatively prime to f^A. For a finite field K with q elements, Wiedemann (loc. cit., Proposition 3) proves that the probability is no less than

$$\frac{1}{12 \max\{\lceil \log_q(\deg\ f^A) \rceil, 1\}}.$$

We shall present a different estimate. For this, we state the following Lemma, which is a key element in our results.

Lemma 1. *Let $\{a_i\}_{i=0}^{\infty} \in \mathsf{K}^{\mathbb{N}}$ be linearly generated, and let m be the degree of its minimum polynomial. For $\mu \geq 0$, consider the Toeplitz matrices*

$$T_\mu := \begin{pmatrix} a_{\mu-1} & a_{\mu-2} & \ldots & a_1 & a_0 \\ a_\mu & a_{\mu-1} & \ldots & a_2 & a_1 \\ \vdots & a_\mu & \ddots & \vdots & a_2 \\ & \vdots & & & \vdots \\ a_{2\mu-3} & & & a_{\mu-1} & \\ a_{2\mu-2} & a_{2\mu-3} & \ldots & a_\mu & a_{\mu-1} \end{pmatrix} \in \mathsf{K}^{\mu \times \mu}.$$

Then $\mathrm{Det}(T_m) \neq 0$ and for all $M > m$, $\mathrm{Det}(T_M) = 0$.

Proof. If the polynomial $g(\lambda) = \lambda^M + c_{M-1}\lambda^{M-1} + \cdots + c_0$ generates $\{a_i\}_{i=0}^{\infty}$, then

$$T_M \begin{pmatrix} c_{M-1} \\ \vdots \\ c_0 \end{pmatrix} = - \begin{pmatrix} a_M \\ \vdots \\ a_{2M-1} \end{pmatrix}.$$

Clearly, for each $M > m$, the above linear system has several solutions corresponding to all the polynomials $g(\lambda)$ that are multiples of the minimum polynomial, hence $\mathrm{Det}(T_M) = 0$. For $M = m$, the only solution to the system is formed by the low order coefficients of the minimum polynomial. This is because a polynomial of degree m that linearly generates the initial segment $\{a_0, \ldots, a_{2m-1}\}$ must already generate the entire sequence, hence its monic associate must coincide with the minimum polynomial. ⊠

From this lemma we can derive, using the approach of Schwartz (1980) (see also Zippel (1979)), the following probability estimate.

Lemma 2. *Let $A \in \mathsf{K}^{n \times n}$, and let $S \subset \mathsf{K}$. Randomly and uniformly select a row vector $u \in S^{1 \times n}$ and a column vector $b \in S^n$. Then the probability*

$$\mathrm{Prob}(f_u^{A,b} = f^A) \geq 1 - \frac{2\deg(f^A)}{\mathrm{card}(S)}.$$

Proof. Let $\vec{v}$ be an n-dimensional row vector with entries $v_1, \ldots, v_n$ being indeterminates, and let $\vec{\beta}$ be an n-dimensional column vector with entries being fresh indeterminates $\beta_1, \ldots, \beta_n$. Then

$$\{\alpha_i\}_{i=0}^{\infty} \in \mathsf{L}^{\mathbb{N}} \text{ with } \alpha_i := \vec{v} A^i \vec{\beta},$$

$\mathsf{L} := \mathsf{K}(v_1, \ldots, v_n, \beta_1, \ldots, \beta_n)$, is linearly generated by f^A. For $m = \deg(f^A)$, consider the Toeplitz matrix $\mathcal{T}_m$ define in Lemma 1 with respect to $\mu = m$ and the sequence $\{\alpha_i\}_{i=0}^{\infty}$. Since there exist vectors $u \in \mathsf{K}^{1 \times n}$ and $b \in \mathsf{K}^n$ with $f_u^{A,b} = f^A$, by Lemma 1 there exist values in K for the indeterminates $v_1, \ldots, \beta_n$ such that the evaluation of $\mathcal{T}_m$ at those values yields a non-singular matrix. Hence, $\mathcal{T}_m$ is non-singular as a matrix over L, and

$$0 \neq \tau := \mathrm{Det}(\mathcal{T}_m) \in \mathsf{K}[v_1, \ldots, v_n, \beta_1, \ldots, \beta_n].$$

If evaluating τ at the coordinates of $u \in S^{1 \times n}$ and $b \in S^n$ results in a non-zero value, the corresponding Toeplitz matrix for $\{uA^ib\}_{i=0}^{\infty}$ remains non-singular, which again by Lemma 1 implies that $\deg(f_u^{A,b}) \geq m$. By the Schwartz/Zippel lemma the probability that τ does not vanish is bounded from below by $1 - \deg(\tau)/\mathrm{card}(S)$. ⊠

For a matrix $A \in \mathsf{K}^{n \times n}$, one can now in a Las Vegas randomized fashion verify that $\mathrm{Det}(A) = 0$. First, one randomly selects $u \in S^{1 \times n}$ and $b \in S^n$, with $S \subset \mathsf{K}$ and $\mathrm{card}(S) \geq 2n/\epsilon$, where $0 < \epsilon \ll 1$. Second, one computes

$$\{a_0, a_1, \ldots, a_{2n-1}\}, \quad a_i := uA^ib.$$

Third, one finds the minimum degree linear generating polynomial for the above sequence of $2n$ elements. By the theory of linearly generated sequences, this polynomial is equal to $f_u^{A,b}$. Finally, if $\lambda \mid f_u^{A,b}(\lambda)$, then $\mathrm{Det}(A) = 0$. For a singular matrix, this condition will occur with probability no less than $1 - \epsilon$.

For a non-singular matrix $A \in \mathsf{K}^{n \times n}$, Wiedemann presents a Las Vegas randomized algorithm for computing $\mathrm{Det}(A)$. Our parallel solution will utilize his method, but the randomization can be simplified. We owe the next theorem to B. David Saunders.

Theorem 2. *Let $A \in \mathsf{K}^{n\times n}$ be non-singular, and let $S \subset \mathsf{K}$. Consider the matrix*

$$\widehat{A} := AH, \quad H := \begin{pmatrix} h_0 & h_1 & \dots & h_{n-2} & h_{n-1} \\ h_1 & \dots & \iddots & h_{n-1} & h_n \\ \vdots & & & & \vdots \\ h_{n-1} & h_n & \dots & & h_{2n-2} \end{pmatrix}$$

where the elements of the Hankel matrix H are randomly and uniformly selected from the set S. With $\widehat{A}_i$ denoting the leading principal $i \times i$ submatrix of $\widehat{A}$, the probability

$$\mathrm{Prob}(\mathrm{Det}(\widehat{A}_i) \neq 0 \text{ for all } 1 \leq i \leq n) \geq 1 - \frac{n(n+1)}{2\,\mathrm{card}(S)}.$$

Proof. For an $n \times n$ matrix B, denote by $B_{I,J}$ the determinant of the submatrix of B that is formed by removing from B all rows not contained in the set I and all columns not contained in the set J. First, assume that $\mathcal{H}$ is a generic Hankel matrix, whose entries are new variables $\eta_0, \dots, \eta_{2n-2}$ replacing $h_0, \dots, h_{2n-2}$, and let $\widehat{\mathcal{A}} = A\mathcal{H} \in \mathsf{L}^{n\times n}$, where $\mathsf{L} := \mathsf{K}(\eta_0, \dots, \eta_{2n-2})$. For $I = \{1, \dots, i\}$ the Cauchy-Binet formula yields

$$\widehat{\mathcal{A}}_{I,I} = \sum_{\substack{J=\{j_1,\dots,j_i\} \\ 1\leq j_1<\dots<j_i\leq n}} A_{I,J}\mathcal{H}_{J,I}.$$

We claim that $\widehat{\mathcal{A}}_{I,I}$, which is the ith leading principal minor of $\widehat{\mathcal{A}}$, is non-zero in L. The argument observes that, for $J = \{j_1, \dots, j_i\}$, the diagonal term $\eta_{j_1-1}\eta_{j_2}\eta_{j_3+1}\cdots\eta_{j_i+i-1}$ is the lexicographically lowest order term in the minor expansion for $\mathcal{H}_{J,I}$. Therefore, all $\mathcal{H}_{J,I}$ have distinct lowest terms, hence are linearly independent over K, and thus $\widehat{\mathcal{A}}_{I,I} \neq 0$, provided there exists a J_0 with $A_{I,J_0} \neq 0$. This is true since the first i rows of A are linearly independent. If we set

$$0 \neq \sigma := \prod_{i=1}^{n} \mathcal{A}_{I,I} \in \mathsf{K}[\eta_0, \dots, \eta_{2n-2}],$$

then it is clear that all those H whose entries are not zeros of the polynomial σ will satisfy the lemma. Again by the Schwartz/Zippel lemma, the probability that σ does not vanish on random values from S for the η's is no less than $1 - \deg(\sigma)/\mathrm{card}(S)$. ⊠

If all principal submatrices of $\widehat{A}$ are non-singular, Wiedemann shows that for the matrix

$$\widetilde{A} := \widehat{A}D, \quad D := \mathrm{Diag}(d_1, \dots, d_n),$$

where the d_i are uniformly randomly selected from S, the probability

$$\mathrm{Prob}(f^{\widetilde{A}}(\lambda) = \mathrm{Det}(\lambda I - \widetilde{A})) \geq 1 - \frac{n(2n-2)}{\mathrm{card}(S)}. \quad (1)$$

The algorithm picks a random Hankel matrix H, a random diagonal matrix D, a random row vector u, and a random column vector b, all with entries in S. First, it computes the sequence

$$\{\widetilde{a}_0, \dots, \widetilde{a}_{2m-1}\}, \quad \widetilde{a}_i := u\widetilde{A}^i b, \quad \widetilde{A} := AHD.$$

Second, it determines the minimum polynomial $f_u^{\widetilde{A},b}$ of that sequence, i.e., it finds a polynomial of minimum degree that linearly generates $\{\widetilde{a}_0, \dots, \widetilde{a}_{2m-1}\}$. If $\deg(f_u^{\widetilde{A},b}) < n$ or $f_u^{\widetilde{A},b}(0) = 0$, the algorithm reports failure. Otherwise, $\mathrm{Det}(\lambda I - \widetilde{A}) = f_u^{\widetilde{A},b}(\lambda)$, so it can return

$$\frac{f_u^{\widetilde{A},b}(0)}{\mathrm{Det}(H)\mathrm{Det}(D)}$$

as the value of the determinant of A. Note that since $f_u^{\widetilde{A},b}(0) \neq 0$, $\widetilde{A}$ is non-singular, hence both H and D must also be non-singular, and the division is possible. Putting Theorem 2, Lemma 2, and (1) together we obtain for any non-singular matrix $A \in \mathsf{K}^{n\times n}$ the probability estimate

$$\mathrm{Prob}(\deg(f_u^{\widetilde{A},b}) = n \text{ and } f_u^{\widetilde{A},b}(0) \neq 0) \geq 1 - 3n^2/\mathrm{card}(S). \quad (2)$$

For Galois fields K with $\mathrm{card}(\mathsf{K}) < 3n^2$, the algorithm is performed in an algebraic extension L over K, so that the failure probability can be bounded away from 0.

We have not specified yet how the generating polynomials for the linearly generating sequences are found. Sequentially, the best method is the Berlekamp-Massey algorithm, which can find $f_u^{A,b}$ from $\{ub, uAb, \dots, u \times A^{2n-1}b\}$ in $O(n \deg(f_u^{A,b}))$ field operations. Our parallel solution is based on solving the Toeplitz system described in the proof of Lemma 1, and is discussed in the following section. That approach as a by-product will also produce a method for finding $\mathrm{Det}(H)$.

3. Solution of Toeplitz and General Systems

In §2 we have shown that the problem of solving general systems of linear equations can be reduced to the problem of solving the Toeplitz systems arising from the linearly generated sequences discussed. We now present an algorithm for finding the characteristic polynomial of a Toeplitz matrix. First, we observe that for any matrix $A \in \mathsf{K}^{n\times n}$ one may compute the power series expansion

$$(I - \lambda A)^{-1} = I + \lambda A + \lambda^2 A^2 + \cdots \in \mathsf{K}^{n\times n}[[\lambda]]$$

$$T^{-1} =: \begin{pmatrix} u_1 & \dots & v_n \\ u_2 & \dots & v_{n-1} \\ \vdots & & \vdots \\ u_{n-1} & \dots & v_2 \\ u_n & \dots & v_1 \end{pmatrix} = \frac{1}{u_1}\Bigg(\begin{pmatrix} u_1 & & & & \\ u_2 & u_1 & & & \\ u_3 & u_2 & u_1 & & \\ \vdots & & \ddots & \ddots & \\ u_n & u_{n-1} & \dots & u_2 & u_1 \end{pmatrix} \begin{pmatrix} v_1 & v_2 & v_3 & \dots & v_n \\ & v_1 & v_2 & & v_{n-1} \\ & & \ddots & \ddots & \vdots \\ & & & v_1 & v_2 \\ & & & & v_1 \end{pmatrix} - \begin{pmatrix} 0 & & & & \\ v_n & 0 & & & \\ v_{n-1} & v_n & 0 & & \\ \vdots & & \ddots & \ddots & \\ v_2 & v_3 & \dots & v_n & 0 \end{pmatrix} \begin{pmatrix} 0 & u_n & u_{n-1} & \dots & u_2 \\ & 0 & u_n & & u_3 \\ & & \ddots & \ddots & \vdots \\ & & & 0 & u_n \\ & & & & 0 \end{pmatrix} \Bigg). \tag{5}$$

Figure 1: The Gohberg/Semencul formula.

by Newton iteration:

$$\begin{aligned} &X_0 \leftarrow I; B \leftarrow I - \lambda A; \\ &X_i \leftarrow X_{i-1}(2I - BX_{i-1}) \text{ for } i \leftarrow 1, 2, \dots \end{aligned} \tag{3}$$

Note that X_i is a matrix polynomial in λ of degree no more than $2^i - 1$. Since

$$\begin{aligned} I - BX_i &= I - BX_{i-1}(2I - BX_{i-1}) \\ &= (I - BX_{i-1})(I - BX_{i-1}), \end{aligned}$$

it follows by induction on i that $I - BX_i \equiv 0 \pmod{\lambda^{2^i}}$, i.e.,

$$X_i = I + \lambda A + \cdots + \lambda^{2^i-1} A^{2^i-1}.$$

We now apply (3) to Toeplitz matrices. For a nonsingular $n \times n$ Toeplitz matrix over K,

$$T := \begin{pmatrix} a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \\ a_n & a_{n-1} & \dots & a_2 & a_1 \\ \vdots & a_n & \ddots & \vdots & a_2 \\ & \vdots & & & \vdots \\ a_{2n-3} & & & a_{n-1} & \\ a_{2n-2} & a_{2n-3} & \dots & a_n & a_{n-1} \end{pmatrix}, \tag{4}$$

the inverse can be represented implicitly by the Gohberg/Semencul formulas (see, e.g., Brent et al. (1980)), one of which applies in the case where

$$u_1 := (T^{-1})_{1,1} = (T^{-1})_{n,n} =: v_1 \neq 0,$$

and is stated in Figure 1. Note that T^{-1} is thus fully determined by the entries of its first and last rows.

Note that

$$\begin{aligned} \text{Trace}(T^{-1}) = \frac{1}{u_1}(nu_1v_1 + (n-2)u_2v_2 + \\ \cdots + (-n+2)u_nv_n). \end{aligned}$$

The algorithm (3) is now applied to the Toeplitz matrix $B = T(\lambda) := I - \lambda T$. Then

$$X_i \equiv T(\lambda)^{-1} \pmod{\lambda^{2^i}}$$

where $T(\lambda)$ can be viewed as a Toeplitz matrix with entries in the field of extended power series $\mathsf{K}((\lambda)) = \bigcup_{k \geq 0} \lambda^{-k}\mathsf{K}[[\lambda]]$. Clearly, $T(\lambda)^{-1} \in \mathsf{K}[[\lambda]]^{n \times n}$ and $T(0)^{-1} = I$, so $(T(\lambda)^{-1})_{1,1} \bmod \lambda^i \neq 0$ for any $i \geq 1$. We compute the first and last columns of X_i from the first and last columns of X_{i-1} by formula (5). The first column, for instance, is computed as

$$X_{i-1}(2I - T(\lambda)X_{i-1}) \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{6}$$

Multiplying X_{i-1} from the right by a vector reduces, by (5), to multiplying triangular Toeplitz matrices by vectors, vector subtractions, and dividing a vector by a scalar, i.e., a polynomial in λ. The Toeplitz matrix times vector products can be accomplished by polynomial multiplication. For example, for

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix} := \begin{pmatrix} u_1 & & & & \\ u_2 & u_1 & & & \\ u_3 & u_2 & u_1 & & \\ \vdots & & \ddots & \ddots & \\ u_n & u_{n-1} & \dots & u_2 & u_1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_n \end{pmatrix}$$

we have

$$\begin{aligned} &(u_1 + \cdots + u_n z^{n-1})(w_1 + \cdots + w_n z^{n-1}) \\ &\qquad \equiv y_1 + y_2 z + \cdots + y_n z^{n-1} \pmod{z^n}. \end{aligned}$$

Note that the entries in the first and last columns in X_{i-1}, denoted by $u_j^{(i-1)}$ and $v_j^{(i-1)}$, and the entries in the arising vectors y_i, are themselves polynomials in $\mathsf{K}[\lambda]$ of degree no more than $2^i - 1$. According to (5), the resulting polynomial vector of degree $2^i + 2^{i-1} - 2$

has to be divided by $u_1^{(i-1)}$, the first entry in the first column of X_{i-1}. This is accomplished by multiplying each entry with the power series inverse of $u_1^{(i-1)}$,

$$\frac{1}{u_1^{(i-1)}(\lambda)} \equiv 1 + w_1^{(i-1)}\lambda + \cdots + w_{2^i-1}^{(i-1)}\lambda^{2^i-1} \pmod{\lambda^{2^i}},$$

$w_j^{(i-1)} \in \mathsf{K}$. Note that expansion for the inverse of $u_1^{(i)}$ to order $\lambda^{2^{i+1}-1}$ can be obtained from the first 2^{i-1} terms of this expansion and from $u_1^{(i)}$ with 2 Newton iteration steps (Lipson 1981, §3.3). That expansion will be needed for the determination of the rows and columns of X_{i+1}.

The overall complexity of each iteration in (3) is bounded from above by the complexity of bivariate polynomial multiplication with input degrees bounded by $2n$ and 2^{i+1} in the individual variables, respectively, and can be performed on an algebraic circuit over K in

$$O(2^i n \log n \ \mathrm{loglog}\, n) \text{ size and } O(\log n) \text{ depth}$$

(Cantor and Kaltofen 1987). The overall algebraic circuit complexity for finding the first and last columns of $X_{\lceil \log_2 n \rceil}$ is thus

$$O(n^2 \log n \ \mathrm{loglog}\, n) \text{ size and } O((\log n)^2) \text{ depth.} \quad (7)$$

Again by (5), we can compute

$$\mathrm{Trace}(X_{\lceil \log_2 n \rceil}) \bmod \lambda^n = n + \mathrm{Trace}(T)\lambda + \cdots + \mathrm{Trace}(T^{n-1})\lambda^{n-1}$$

from the first and last columns of $X_{\lceil \log_2 n \rceil}$, again within the complexity (7).

Once we have for all $0 \le i \le n-1$,

$$s_i := \mathrm{Trace}(T^i) = \lambda_1^i + \lambda_2^i + \cdots + \lambda_n^i,$$

where $\lambda_1, \ldots, \lambda_n$ are all eigenvalues of T, it is possible by the Leverrier/Csanky (1976) method to obtain the coefficients of the characteristic polynomial of T. In particular, for

$$\mathrm{Det}(\lambda I - T) =: \lambda^n - c_1\lambda^{n-1} - c_2\lambda^{n-2} - \cdots - c_{n-1}\lambda - c_n,$$

we have

$$\begin{pmatrix} 1 & & & & \\ s_1 & 2 & & & \\ s_2 & s_1 & \ddots & & \\ \vdots & & & & \\ s_{n-2} & \ldots & & n-1 & \\ s_{n-1} & s_{n-2} & \ldots & s_1 & n \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ \vdots \\ s_{n-1} \\ s_n \end{pmatrix}.$$

Clearly, in order for the system to determine all c_i uniquely, one has to divide by $n!$. However, under that condition one can solve such systems in

$$O(n \log n \ \mathrm{loglog}\, n) \text{ size and } O((\log n)^2) \text{ depth}$$

(Schönhage 1982); see also (Pan 1990a, Appendix A). We have thus proven the following theorem.

Theorem 3 (Pan 1990b). *The characteristic polynomial of an $n \times n$ Toeplitz matrix over a field of characteristic zero or greater than n can be computed on an algebraic circuit of $O(n^2 \log n \ \mathrm{loglog}\, n)$ size and $O((\log n)^2)$ depth.*

From Theorem 3 we can obtain by the approach of §3 size-efficient randomized circuits for solving general non-singular systems. Assume now that $A \in \mathsf{K}^{n\times n}$ is a general non-singular matrix. The algorithm picks a random Hankel matrix H, a random diagonal matrix D, a random row vector u, and a random column vector v, and computes

$$\{a_i\}_{i=0}^{2n-1}, \quad a_i := u\widetilde{A}^i v \in \mathsf{K}, \ \widetilde{A} = AHD. \quad (8)$$

The algebraic circuit finds first $\widetilde{A}$, then computes all $\widetilde{A}^i v$, and finally performs the inner products with u. From the doubling argument

$$A^{2^i}(v \mid Av \mid \ldots \mid A^{2^i-1}v) = (A^{2^i}v \mid A^{2^i+1}v \mid \ldots \mid A^{2^{i+1}-1}v) \quad (9)$$

(cf. (Borodin and Munro 1975, p. 128); Keller-Gehrig (1985)) it is easily seen that the algebraic circuit for this step has complexity

$$O(n^\omega \log n) \text{ size and } O((\log n)^2) \text{ depth.} \quad (10)$$

We remark that on an algebraic PRAM, here and in the following estimates a factor $\log n$ can be saved in the asymptotic processor count.

Next, we compute the characteristic polynomial for the Toeplitz matrix T in (4), whose entries are the elements of the sequence (8). By Theorem 3, (10) dominates the complexity of this part of the construction. By (2), the generating polynomial of (8) is with high probability the characteristic polynomial $\lambda^n + \widetilde{c}_1\lambda^{n-1} + \cdots + \widetilde{c}_n$ of $\widetilde{A}$, which by Lemma 1 implies that the Toeplitz matrix T is non-singular. We deduce the solution of

$$T \begin{pmatrix} \widetilde{c}_1 \\ \vdots \\ \widetilde{c}_n \end{pmatrix} = - \begin{pmatrix} a_n \\ \vdots \\ a_{2n-1} \end{pmatrix},$$

from the characteristic polynomial of T, denoted by $\lambda^n + c_1\lambda^{n-1} + \cdots + c_n$, as

$$\begin{pmatrix} \widetilde{c}_1 \\ \vdots \\ \widetilde{c}_n \end{pmatrix} \leftarrow \frac{1}{c_n}\left(T^{n-1} \begin{pmatrix} a_n \\ \vdots \\ a_{2n-1} \end{pmatrix} + \cdots + c_{n-1} \begin{pmatrix} a_n \\ \vdots \\ a_{2n-1} \end{pmatrix}\right).$$

This is because by the Cayley/Hamilton theorem,

$$T^n + c_1 T^{n-1} + \cdots + c_n I = 0.$$

Note that $\widetilde{c}_n$ is equal to $(-1)^n \mathrm{Det}(\widetilde{A})$, which by (2) is likely to be not equal to zero. Again from (9) we deduce that the circuit complexity of this step is (10). Next, we find in the same manner and at the same complexity

$$\widetilde{x} \leftarrow -\frac{1}{\widetilde{c}_n}(\widetilde{A}^{n-1}b + \widetilde{c}_1\widetilde{A}^{n-2}b + \cdots + \widetilde{c}_{n-1}b\,).$$

Now we must have $\widetilde{A}\,\widetilde{x} = b$. We finally compute $A^{-1}b =: x \leftarrow H D\,\widetilde{x}$. We therefore have the following theorem.

Theorem 4. *For $n \geq 1$ there exists a randomized algebraic circuit with $n^2 + n$ inputs, n outputs, $O(n)$ nodes that denote random (input) elements, and of*

$$O(n^\omega \log n) \text{ size and } O((\log n)^2) \text{ depth,}$$

with the following property. If the inputs are the entries of a non-singular matrix $A \in \mathsf{K}^{n\times n}$ and of a vector $b \in \mathsf{K}^n$, where K is a field of characteristic zero or greater than n, and if the random nodes uniformly select field elements in $S \subset \mathsf{K}$, then with probability no less than $1 - 3n^2/\mathrm{card}(S)$ the circuit outputs the entries of $A^{-1}b$. On the other hand, if the random choices are unlucky or if the input matrix is singular, the circuit divides by zero. On non-singular inputs zero-divisions occur with probability no more than $3n^2/\mathrm{card}(S)$.

4. Computing the Inverse Matrix

We now show how a circuit for the determinant of a non-singular matrix can be transformed to a circuit for the inverse of that matrix. Our solution follows the approach by Baur and Strassen (1983). Suppose that a rational function $f \in \mathsf{K}(x_1, \ldots, x_k)$ is computed from the input values $x_1, \ldots, x_k$ by a straight-line program of length l, i.e., an algebraic circuit of size l; for the following arguments it is more convenient to enumerate the nodes, hence the straight-line program model. The program can divide, but it is assumed that the division is by a rational function that is not identical to 0, i.e., there will always exist input values in the algebraic closure of K for which the program avoids a zero-division. Baur and Strassen show that then there exists a straight-line program of length no more than $5l$ that computes all first order partial derivatives

$$\partial_{x_1}(f), \partial_{x_2}(f), \ldots, \partial_{x_k}(f).$$

Furthermore, the new program will divide by exactly the same rational functions as the old, hence no new zero-division will be introduced. Their motivating example was the same as ours. Let $k = n^2$ and

$$f(x_{1,1}, \ldots, x_{n,n}) = \mathrm{Det}(A), \quad A = (x_{i,j})_{1\leq i,j\leq n}.$$

If f is computed by a straight-line program of length $l(n)$, then the inverse of A can also be computed by a program of asymptotic length $O(l(n))$, namely as

$$A^{-1} = \Big((-1)^{i+j}\frac{\partial_{x_{j,i}}(f)}{f}\Big)_{1\leq i,j\leq n}.$$

However, the original construction of Baur and Strassen does not preserve the depth of the program within a constant factor. In order to achieve this, we have to analyze their method more closely and employ implicitly a theorem by Hoover et al. (1984).

Theorem 5 (Kaltofen and Singer 1990). *Let $f \in \mathsf{K}(x_1, \ldots, x_k)$ be computed by a straight-line program P of length l and depth d. Then f and all derivatives $\partial_{x_1}(f), \partial_{x_2}(f), \ldots, \partial_{x_k}(f)$ can be computed by a straight-line program Q of length no more than $4l$ and depth $O(d)$.*

Proof. Let $v_i \leftarrow v_{I_1(i)} \circ_i v_{I_2(i)}$, $k+1 \leq i \leq k+l$, be the $(i-k)$-th instruction in the program P. Here the function I_1 retrieves the index of the left operand of right-hand side expression, and the function I_2 the right operand. We set $v_i := x_i$ for $1 \leq i \leq k$, hence we have a range for the operand indices of $1 \leq I_1(i), I_2(i) < i$. If the left or right operands are scalars, no such indexing will be needed. For $i > k$ the symbol v_i stands, strictly speaking, for a program variable in P, or a node in the computation DAG for f. However, we also use it to identify with it the rational function in $\mathsf{K}(x_1, \ldots, x_k)$ that is computed in this variable. Baur and Strassen's construction proceeds by viewing f as a sequence of functions

$$g_i(y_1, \ldots, y_i) \in \mathsf{K}(y_1, \ldots, y_i), \quad k + l \geq i \geq k.$$

We will have

$$g_i(v_1, \ldots, v_i) = f(x_1, \ldots, x_k) \text{ for all } k \leq i \leq k + l.$$

The interpretation of g_i is the function that gets computed in the program variable v_l if one omits the instructions for $v_{k+1}, \ldots, v_i$ in the program P and replaces $v_1 = x_1, \ldots, v_k = x_k, v_{k+1}, \ldots, v_i$ by the new variables $y_1, \ldots, y_i$ whenever they are used in the truncated program of length $l - i + k$. In particular, we will have

$$g_k(x_1, \ldots, x_k) = f(x_1, \ldots, x_k).$$

Now let

$$h_i(y_{I_1(i)}, y_{I_2(i)}) = y_{I_1(i)} \circ_i y_{I_2(i)} \in \mathsf{K}(y_1, \ldots, y_{i-1}),$$

$k + l \geq i \geq k + 1$, denote the rational function that gets formally computed by the $(i-k)$th instruction in P. The functions g_i, $i = k+l, k+l-1, \ldots, k$, are therefore

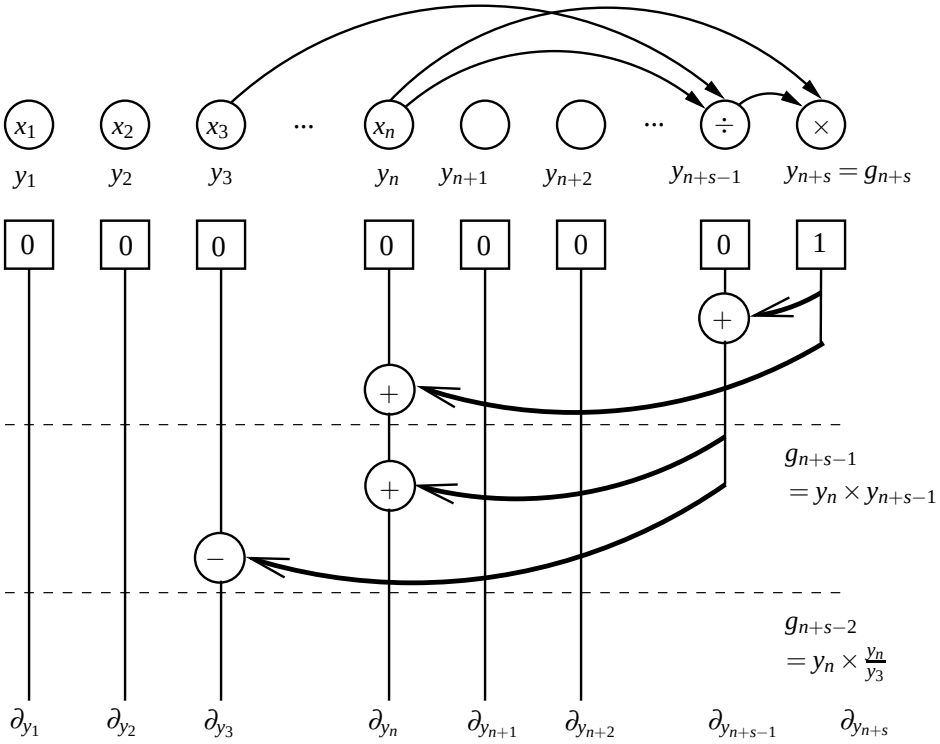

Figure 2: Coarse view of the Baur and Strassen construction.

inductively defined to be related by

$$g_{i-1}(y_1, \ldots, y_{i-1}) := g_i(y_1, \ldots, y_{i-1}, h_i(y_{I_1(i)}, y_{I_2(i)})), \tag{11}$$

where initially $g_{k+l}(y_1, \ldots, y_{k+l}) := y_{k+l}$. The goal is to compute

$$\partial_{y_1}(g_k), \partial_{y_2}(g_k), \ldots, \partial_{y_k}(g_k).$$

This is done by using the inductive definition of g_i and the chain rule for partial derivatives. This rule states for $g \in \mathsf{K}(y_1, \ldots, y_m)$ and $\hbar_1, \ldots, \hbar_m \in \mathsf{K}(x_1, \ldots, x_k)$ that

$$\partial_{x_i}(g(\hbar_1, \ldots, \hbar_m)) = \sum_{j=1}^{m} (\partial_{y_j} g)(\hbar_1, \ldots, \hbar_m)\ \partial_{x_i}(\hbar_j),$$

provided the denominator of g does not become zero in $\mathsf{K}(x_1, \ldots, x_k)$ by setting y_j to $\hbar_j(x_1, \ldots, x_k)$ for all $1 \leq j \leq m$, in which case the same is true for $\partial_{y_j} g$. Note that this rule can be proven for any field by entirely algebraic means (see (Kaltofen and Singer 1990)). In our case, only the last function $\hbar_m$ will not be the identity.

We first have

$$\partial_{y_j}(g_{k+l}) = 0 \text{ for all } 1 \leq j \leq k+l-1,\ \partial_{y_{k+l}}(g_{k+l}) = 1.$$

Now let us assume that at level $k+l-i$ we have already computed the derivatives

$$\partial_{y_1}(g_i), \partial_{y_2}(g_i), \ldots, \partial_{y_i}(g_i).$$

From (11) we get by the chain rule for $j_1 := I_1(i)$ and $j_2 := I_2(i)$ that

$$(\partial_{y_j} g_{i-1})(y_1, \ldots, y_{i-1}) = (\partial_{y_j} g_i)(y_1, \ldots, y_{i-1}, h_i(y_{j_1}, y_{j_2}))$$

for $1 \leq j \leq i-1$, $j \neq j_1$, $j \neq j_2$, and

$$(\partial_{y_j} g_{i-1})(y_1, \ldots, y_{i-1}) = (\partial_{y_j} g_i)(y_1, \ldots, y_{i-1}, h_i(y_{j_1}, y_{j_2})) + (\partial_{y_i} g_i)(y_1, \ldots, y_{i-1}, h_i(y_{j_1}, y_{j_2}))(\partial_{y_j} h_i)(y_{j_1}, y_{j_2})$$

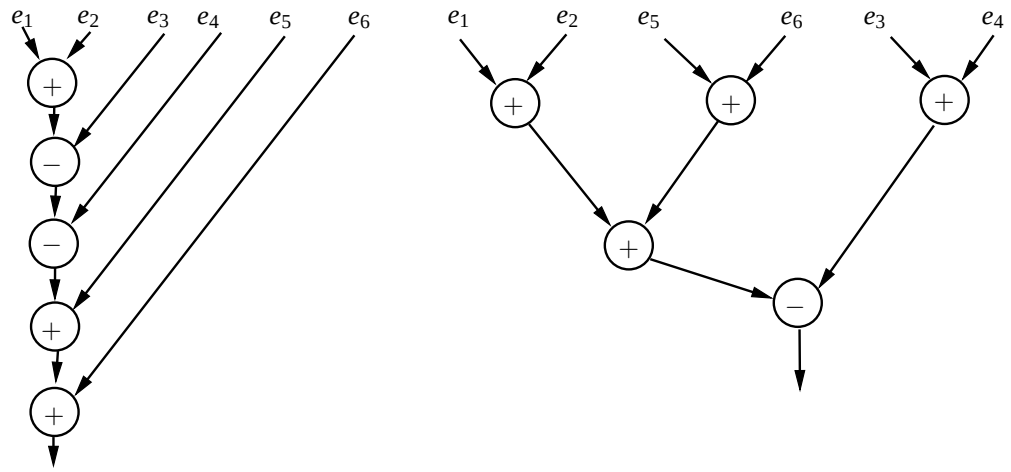

Figure 3: Balancing an accumulation tree.

for $j = j_1$ or $j = j_2$.

The dynamics of these rules are displayed in Figure 2. The substitution $h_i(y_{j_1}, y_{j_2})$ for y_i is accomplished by connecting the corresponding node to the nodes for y_{j_1} and y_{j_2} and performing the operation $\circ_i$ in the node. Then the derivatives $\partial_{y_j}(g_{i-1})$ are computed from those of $\partial_{y_j}(g_i)$ plus a value derived from

$$\overline{\partial_{y_i}(g_i)} = (\partial_{y_i} g_i)(y_1, \ldots, y_{i-1}, h_i(y_{j_1}, y_{j_2}))$$

and the derivatives of $h_i(y_{j_1}, y_{j_2})$. The latter are solely dependent on the nodes corresponding to the variables y_{j_1} and y_{j_2} and require constant work. In Figure 2, this is indicated by a thick connection from the line for ∂_{y_i} to ∂_{y_j}. Let us for a moment consider the operation $\circ_i$ with the most costly work, namely division. For $h_i(y_{j_1}, y_{j_2}) = y_{j_1}/y_{j_2}$ we have

$$\partial_{y_{j_1}}(h_i) = \frac{1}{y_{j_2}} \text{ and } \partial_{y_{j_2}}(h_i) = -\frac{y_{j_1}}{y_{j_2}^2}.$$

The strategy is to divide $\overline{\partial_{y_i}(g_i)}$ by y_{j_2}, add that into the $\partial_{y_{j_1}}$ line, or multiply it with y_{j_1}/y_{j_2}, the value computed in the node for y_i, and then subtract that from the ∂_{y_2} line. In other words, if $\circ_i = \div$ one needs 4 additional operations to go to the next level. There is one more issue that needs to be settled in the division case. Later substitutions for y_{j_2} are not allowed to cause a division by a function that is identical zero. This never occurs because the circuit computing the derivatives will only divide by quantities that the original program P divides by.

We now discuss how to accomplish the given length and depth measures. For each v_i in P, $k+1 \le i \le k+l$, we will introduce at most 5 instructions in our new program Q, one from the original program and at most 4 more to eliminate y_i. This leads to an upper bound of $5l$ for the length of Q, but l of these instructions either add the initial $\partial_{y_j}(g_{k+l}) = 0$ to $\overline{\partial_{y_i}(g_i)}\,(\partial_{y_j} h_i)(y_{j_1}, y_{j_2})$ or multiply $\partial_{y_{k+l}}(g_{k+l}) = 1$ by $\partial_{y_j}(h_{k+l})(y_{j_1}, y_{j_2})$. Since we have each instruction v_i participate in the computation of v_{k+l}, there are at least l trivial instructions that can be eliminated from such a Q. Note that if we only have subtractions on a line for ∂_{y_j} we pass the minus sign along to the level for the derivatives of g_{j-1}. On each line for ∂_{y_j}, $1 \le j \le k$, we might then have to negate the final result, potentially costing us an additional k instructions. However, we also save that many instructions at the starting level of those lines, and therefore we do not need more than $4l$ instructions overall.

Lastly, we discuss how to accomplish the stated depth. First we observe that if we were to treat each line in Figure 1 on which we accumulate the ∂_{y_j} as a single node, and if we were to treat each connection from ∂_{y_i} to $\partial_{y_{j_1}}$ and to $\partial_{y_{j_2}}$ as a single edge, then the circuit that computes the derivatives would be a mirror image of the original circuit for f. Therefore, the depth of this abstraction of the part of Q that implements the chain rules and which has "superedges" and "supernodes" is d. Furthermore, on each "superedge" we only have a constant delay. Let t_j be the fan-out for v_j in P, i.e., the number of times the variable v_j is used in later instructions. Then in each supernode corresponding to the line for ∂_{y_j} we have exactly $t_j - 1$ addition and subtraction nodes. Separating the lines that get added from the ones that get subtracted, we can build with $t_j - 1$ nodes a tree that performs the same computation but which has $O(\log t)$ depth (see Figure 3). Hence the entire depth of Q can be made at least $O(d \log t)$, where $t = \max\{t_j\}$.

We finally reduce the depth of Q further to $O(d)$ without increasing the length. To accomplish this, we apply

the transformation of Hoover et al. (loc. cit.) in an implicit way to the circuit constructed above. Consider the lower part of Q that is a mirror image of P (see Figure 1). Furthermore, assume that the subtrees in Q which perform additions on the ∂_{y_j} lines are again contracted to "supernodes". We suppose that subtractions are already separated out, and subtraction nodes on those lines remain untouched. Thus, the depth of this abstraction of the circuit is still $O(d)$, the extra factor of $\log t$ coming from the delay in the supernodes. Now we apply the construction by Hoover et al. to this high level description of the lower part of Q, reversing the flow of information. That construction will insert behind each node of high fan-out a binary tree of duplication nodes whose root is that node and whose leaves are the targets of the arcs leaving that node. Hoover et al. then show that if one optimizes the structure of that tree with respect to the distance of the target nodes to the output node in such a way that target nodes from which there are long paths to output nodes are close to the root, one can overall retain depth $O(d)$. Once such duplication trees are in place behind the supernodes, all we have to do is to reverse the flow of information and perform additions in both supernodes and duplication nodes. ⊠

We now apply this Theorem to the circuit constructed in Theorem 4, which as an auxiliary value computes

$$\widetilde{c}_n = (-1)^n \mathrm{Det}(AHD).$$

The random matrix H is of Hankel form, whose mirror image across a horizontal line that evenly splits the rows becomes a Toeplitz matrix. By Theorem 3, we can thus determine $\mathrm{Det}(H)$ efficiently in parallel. Therefore we have found a circuit that efficiently computes in parallel

$$\mathrm{Det}(A) = (-1)^n \widetilde{c}_n / (\mathrm{Det}(H)\mathrm{Det}(D)),$$

and by applying Theorem 5 to that circuit, we obtain the following result.

Theorem 6. *For $n \geq 1$ there exists a randomized algebraic circuit with n^2 inputs, n^2 outputs, $O(n)$ nodes that denote random (input) elements, and of*

$$O(n^\omega \log n) \text{ size and } O((\log n)^2) \text{ depth,}$$

with the following property. If the inputs are the entries of a non-singular matrix $A \in \mathsf{K}^{n \times n}$, where K is a field of characteristic zero or greater than n, and if the random nodes uniformly select field elements in $S \subset \mathsf{K}$, then with probability no less than $1 - 3n^2/\mathrm{card}(S)$ the circuit outputs the entries of A^{-1}. On the other hand, if the random choices are unlucky or if the input matrix is singular, the circuit divides by zero. On non-singular inputs zero-divisions occur with probability no more than $3n^2/\mathrm{card}(S)$.

There is a second interesting application of Theorem 5 to linear system solving. Assume that one is given a circuit with $n^2 + n$ inputs and n outputs that computes $A^{-1}b$ in size $l(n)$ and depth $d(n)$. Then there exists a circuit of size $4l(n)$ and depth $O(d(n))$ that computes $(A^{\mathrm{tr}})^{-1}b$, where A^{tr} denotes the transposed matrix of A. The proof is by considering

$$f(x_1, \ldots, x_n) := (x_1 \ \ldots \ x_n)(A^{\mathrm{tr}})^{-1}b.$$

The function f can be quickly computed by performing the multiplications from the left to the right, using the given circuit for finding

$$(x_1 \ \ldots \ x_n)(A^{\mathrm{tr}})^{-1} = (A^{-1} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix})^{\mathrm{tr}}.$$

Observing that

$$(\partial_{x_1} f \ \ldots \ \partial_{x_n} f)^{\mathrm{tr}} = (A^{\mathrm{tr}})^{-1}b,$$

we deduce the stated claim from Theorem 5. In a special case this construction gives us a fast transposed Vandermonde system solver based on fast polynomial interpolation. Note that for a fixed matrix A^{-1}, i.e., the case of multiplying a matrix by b vs. multiplying the transpose of that matrix by b (the non-singularity assumption can then be dropped), this fact was proven by Kaminski et al. (1988) without making use of the Baur and Strassen result.

5. Extensions

The results for computing the characteristic polynomial of a Toeplitz matrix in Theorem 3 can be extended to the case where the field of entries has small positive characteristic. The approach is to appeal to Chistov's (1985) method for finding the characteristic polynomial of an arbitrary matrix in conjunction with computing for all $i \leq n$ by the algorithm of §3 the entry $((I_i - \lambda T_i)^{-1})_{i,i} \bmod \lambda^{n+1}$ for the $i \times i$ identity matrix I_i and the $i \times i$ leading principal submatrix T_i of an $n \times n$ Toeplitz matrix $T = T_n$. The resulting circuit then can compute the characteristic polynomials of all T_i over a field of any characteristic in

$$O(n^3 \log n \ \mathrm{loglog}\, n) \text{ size and } O((\log n)^2) \text{ depth.} \quad (12)$$

The efficient parallel algorithms for computing the characteristic polynomial of a Toeplitz matrix are extendible to structured Toeplitz-like matrices such as Sylvester matrices. In particular, it is then possible to compute the greatest common divisor of two polynomials of degree n over a field of characteristic zero or greater n, and also the coefficients of the polynomials

in the Euclidean scheme, on circuits of

$$O(n^2 \log n \ \mathrm{loglog}\, n) \text{ size and } O((\log n)^3) \text{ depth.}$$

Again using a factor of n more processors, the algorithms extend to fields of any characteristic. We refer to (Pan 1990b) and (Bini and Pan 1991) for the details of these results.

The complexity measures (12) in the case of small positive characteristic also apply to the problem of solving general linear systems of equations. For example, there exist randomized circuits of complexity (12) that compute the inverse of an $n \times n$ non-singular matrix over any field. Furthermore, the methods presented here can be also used to compute the rank r of a matrix and to solve a singular system. The former can be accomplished, for instance, by a randomization such that precisely the first r principal minors in the randomized matrix are not zero, and then by performing a binary search for the largest non-singular principal submatrix (cf. (Borodin et al. 1982)). Similarly, one can compute a vector in the solution manifold of a singular linear system, and a basis for the null space of a matrix. For the latter claim, one needs Theorem 6 as follows: consider $A \in \mathsf{K}^{n\times n}$, and assume that for random non-singular matrices $U, V \in \mathsf{K}^{n\times n}$, the product matrix $\widehat{A} := UAV$ has the property that the $r \times r$ leading principal submatrix $\widehat{A}_r$ of $\widehat{A}$ is non-singular, where r is the rank of A. Then

$$\widehat{A}\,E = \begin{pmatrix} \widehat{A}_r & 0^{r\times(n-r)} \\ C & 0^{(n-r)\times(n-r)} \end{pmatrix}$$

for

$$\widehat{A} =: \begin{pmatrix} \widehat{A}_r & B \\ C & D \end{pmatrix} \text{ and } E := \begin{pmatrix} I_r & -\widehat{A}_r^{-1}B \\ 0^{(n-r)\times r} & I_{n-r} \end{pmatrix},$$

hence the right null space of A is spanned by the columns of

$$VE \begin{pmatrix} 0^{r\times(n-r)} \\ I_{n-r} \end{pmatrix}.$$

Finally, the techniques of Pan (1990a) combined with the processor efficient algorithms for linear system solving presented here immediately yield processor efficient least-squares solutions to general linear systems over any field of characteristic zero.

Literature Cited

Baur, W. and Strassen, V., "The complexity of partial derivatives," *Theoretical Comp. Sci.* **22**, pp. 317–330 (1983).

Berkowitz, S. J., "On computing the determinant in small parallel time using a small number of processors," *Inform. Process. Letters* **18**, pp. 147–150 (1984).

Bini, D. and Pan, V., *Numerical and Algebraic Computations with Matrices and Polynomials*; Lecture Notes in Theor. Comput. Sci., edited by R. V. Book; Birkhäuser Boston, Inc., 1991. To appear.

Borodin, A., von zur Gathen, J., and Hopcroft, J. E., "Fast parallel matrix and GCD computations," *Inf. Control* **52**, pp. 241–256 (1982).

Borodin, A. and Munro, I., *Computational Complexity of Algebraic and Numeric Problems*; American Elsevier, New York, N.Y., 1975.

Brent, R. P., Gustavson, F. G., and Yun, D. Y. Y., "Fast solution of Toeplitz systems of equations and computation of Padé approximants," *J. Algorithms* **1**, pp. 259–295 (1980).

Bunch, J. R. and Hopcroft, J. E., "Triangular factorization and inversion by fast matrix multiplication," *Math. Comp.* **28**, pp. 231–236 (1974).

Cantor, D. G. and Kaltofen, E., "Fast multiplication of polynomials over arbitrary rings," *Tech. Report* **87-35**, Dept. Comput. Sci., Rensselaer Polytechnic Institute, December 1987. Revised version to appear in *Acta Informatica*.

Chistov, A. L., "Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic," *Proc. FCT '85, Springer Lec. Notes Comp. Sci.* **199**, pp. 63–69 (1985).

Coppersmith, D. and Winograd, S., "Matrix multiplication via arithmetic progressions," *J. Symbolic Comput.* **9**/3, pp. 251–280 (1990).

Csanky, L., "Fast parallel matrix inversion algorithms," *SIAM J. Comput.* **5**/4, pp. 618–623 (1976).

Galil, Z. and Pan, V., "Parallel evaluation of the determinant and of the inverse of a matrix," *Inform. Process. Letters* **30**, pp. 41–45 (1989).

Hoover, H. J., Klawe, M. M., and Pippenger, N. J., "Bounding fan-out in logical networks," *J. ACM* **31**/1, pp. 13–18 (1984).

Kaltofen, E. and Singer, M. F., "Size efficient parallel algebraic circuits for partial derivatives," *Tech. Report* **90-32**, Dept. Comput. Sci., Rensselaer Polytechnic Inst., Troy, N.Y., October 1990.

Kaminski, M., Kirkpatrick, D. G., and Bshouty, N. H., "Addition requirements for matrix and transposed matrix products," *J. Algorithms* **9**, pp. 354–364 (1988).

Karp, R. M. and Ramachandran, V., "Parallel algorithms for shared-memory machines," in *Handbook for Theoretical Computer Science*; North-Holland, pp. 869–941, 1990.

Keller-Gehrig, W., "Fast algorithms for the characteristic polynomial," *Theor. Comput. Sci.* **36**, pp. 309–317 (1985).

Lipson, J., *Elements of Algebra and Algebraic Computing*; Addison-Wesley Publ., Reading, Mass., 1981.

Pan, V., "Parallel least-square solution of general and Toeplitz-like linear systems," *Proc. 2nd Ann. Symp. Parallel Algorithms Architecture*, pp. 244-253 (1990a).

Pan, V., "Parameterization of Newton's iteration for computations with structured matrices and applications," *Tech. Report* **CUCS-032-90**, Comput. Sci. Dept., Columbia University, New York, N. Y., 1990b.

Preparata, F. P. and Sarwate, D. V., "An improved parallel processor bound in fast matrix inversion," *Inform. Process. Letters* **7**/3, pp. 148–150 (1978).

Schwartz, J. T., "Fast probabilistic algorithms for verification of polynomial identities," *J. ACM* **27**, pp. 701–717 (1980).

Schönhage, A., "The fundamental theorem of algebra in terms of computational complexity," *Tech. Report*, Univ. Tübingen, 1982.

Wiedemann, D., "Solving sparse linear equations over finite fields," *IEEE Trans. Inf. Theory* **IT-32**, pp. 54–62 (1986).

Zippel, R. E., "Probabilistic algorithms for sparse polynomials," *Proc. EUROSAM '79, Springer Lec. Notes Comp. Sci.* **72**, pp. 216–226 (1979).