Journal of Computational Mathematics, Vol.9, No.1, 1991, 33–40.

A PARALLEL ALGORITHM FOR TOEPLITZ TRIANGULAR MATRICES*1)

Chen Ming-kui Lu Hao
(*Department of Mathematics, Xi'an Jiaotong University, Xi'an, China*)

Abstract

A new parallel algorithm for inverting Toeplitz triangular matrices as well as solving Toeplitz triangular linear systems is presented in this paper. The algorithm possesses very good parallelism, which can easily be adjusted to match the natural hardware parallelism of the computer systems, that was assumed to be much smaller than the order n of the matrices to be considered since this is the usual case in practical applications. The parallel time complexity of the algorithm is $O([n/p]\log n + \log^2 p)$, where p is the hardware parallelism.

§1. Introduction

Parallelly inverting triangular matrices and solving triangular linear systems is an interesting problem both in theory and practice. The best parallel algorithm known so far requires $O(\log^2 n)$ time steps and $O(n^3)$ processors, where n is the order of the matrices[5,6]. The order of the time complexity can not be reduced further even for more strongly structured triangular matrices, but the number of processors needed can be reduced. Although the approximate algorithm for parallelly solving Toeplitz triangular systems presented in [1] reduces the time complexity, it requires precomputations, and does not seem practical since some restriction must be imposed on the parameter ε to ensure the nonsingularity of matrix A_ε[1]. Chen and Lu[2] constructed an algorithm for inverting Toeplitz triangular matrices and solving Toeplitz triangular linear systems, by which the number of processors needed to perform the algorithm can be reduced to n.

In practical applications, the number of processors of a computer system, denoted by p, is limited and frequently much smaller than n, the order of the matrices. We will consider the problem on parallelly inverting Toeplitz triangular matrices as well as solving the associated linear systems in this case. The parallel time complexity of the algorithm presented here is $O([n/p]\log n + \log^2 p)$, where $\log n$ means $\log_2 n$ and $[x]$ is the integer ceiling function of x.

We will first give a method to carry out multiplication of a vector by a circulant or a block circulant matrix in §2, and then develop an algorithm for computing the product of the Toeplitz or the block Toeplitz matrix and vector in §3. In §4, the method for inverting Toeplitz triangular matrices as well as solving the associated Toeplitz systems will be constructed.

* Received April 2, 1987.
1) This work was supported by the National Natural Science Foundation of China.

§2. Circulant Matrices and Block Circulant Matrices

Consider the following special class of Toeplitz matrices

$$C = \begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{q-1} \\ c_{q-1} & c_0 & c_1 & \cdots & c_{q-2} \\ c_{q-2} & c_{q-1} & c_0 & \cdots & c_{q-3} \\ \vdots & \vdots & \vdots & & \vdots \\ c_1 & c_2 & c_3 & \cdots & c_0 \end{bmatrix}, \qquad (2.1)$$

which are called circulant matrices. This kind of matrices are completely defined by their first row, and thus frequently denoted by

$$C = \text{ circ } (c_0, c_1, c_2, \cdots, c_{q-1}).$$

Circulant matrices can be diagonalized by the Fourier matrix $F = (f_{ij})_{q\times q}$ with elements $f_{ij} = q^{-1/2}\omega^{-(i-1)(j-1)}$ $(i, j = 1, 2, \cdots, q)$, where ω is the primitive nth root of unity[4], i.e., it holds for any circulant matrices that

$$C = F^H DF, \qquad (2.2)$$

where

$$D = \text{ diag } (\lambda_0, \lambda_1, \cdots \lambda_{q-1}), \qquad (2.3)$$

and the eigenvalues λ_k's of C are defined by

$$\lambda_k = \sum_{l=0}^{q-1} c_l \omega^{kl}, \quad k = 0, 1, \cdots, q-1. \qquad (2.4)$$

It is easy to see that premultiplying a vector by matrices F and F^H may be accomplished by Fast Fourier Transform (FFT) and its inverse, respectively, and the eigenvalues λ_k's can be computed via FFT[4]. Thus, multiplying any q-vector by a circulant can be accomplished in $(3\log q + 1)$ time steps with q processors[2].

A block matrix of the form

$$C_b = \text{ circ } (T_0, T_1, T_2, \cdots, T_{q-1}),$$

where each of the blocks T_j is a pth order matrix, is called a block circulant matrix. It is easily verified that

$$C_b = \sum_{k=0}^{q-1} P^k \otimes T_k, \qquad (2.5)$$

where the notation $\otimes$ denotes the Kronecker product of matrices, and

$$P = \text{ circ } (0, 1, 0, \cdots, 0)$$

is a circulant of order q, and that (see [4])

$$P = F^H \tilde{D} F, \qquad (2.6)$$