DECENTRALIZED PARALLEL ALGORITHMS FOR MATRIX COMPUTATION

Rajani M. Kant and Takayuki Kimura
Department of Statistics and Computer Science
University of Delaware
Newark, Delaware 19711

Abstract

It is shown that a mesh-connected n×n multiprocessor system can compute the inverse of a n×n matrix in linear time to n. The algorithm is based on a theorem known to Sylvester in 1851. It computes the cofactor matrix in n steps, each of which involves 4 unit distance message routing and 4 arithmetic operations for every processor. A coding and memory requirement for each processor is the same and is independent of n.

It is also shown that the same algorithm solves systems of n linear equations in linear time of n with n × (n+1) processors.

1. Introduction

Below we will give a matrix inversion algorithm (※K-algorithm) suitable for execution by a mesh-connected multiprocessor system. The computation time is linear to the size of the matrix. This is an improvement over the result of VanScoy[5] whose computation time is $O(n^2)$, where n is the size of the matrix. For other parallel algorithms of matrix computation, see Pease[6] and Csanky[7].

Our algorithm is an answer to the following distributive computation problem:

Given n×n processors mesh-connected into a toroidal shape, initially each processor carrying one number to form a n×n matrix, the problem is to find an algorithm for each processor such that the inverse of the matrix can be computed and stored by the toroid under the following constraints:

(1) The interprocessor communication is asynchronous and is independent of n. No central communication facility, ex. clock signal or any broadcasting, is available.
(2) The algorithm is to be same for all the processors with the possible exception of a fixed number, and is independent of n.
(3) The local memory space required in each processor is fixed and independent of n.
(4) The number of types of control messages used is fixed and independent of n. The number of messages used can be a function of n.
(5) The computation time is to be of the order of n.

The reason for the constraint (5) is that with the result of Strassen[3], a single processor system can compute the inverse matrix by $O(n^{2.81})$ steps. With n^2 processors why can not we do it by $O(n^{0.81})$ steps?

The key question here is the problem of communication complexity. If there is a very complex communication device by which every processor can communicate with any other processor or with the shared memory at any time, then to write a program for n^2 processors to achieve $O(n^{0.81})$ computation might not be a difficult problem after Strassen. In this case, however, the complexity of such a device is proportional to n^4. On the other hand, the complexity of communication devices in a mesh-connected multiprocessor system is a constant and is independent of n. Decomposition of such a complex communication requirement as the one in a matrix inversion algorithm into a simple one such as the four neighborhood communication facility is not trivial.

No known parallel algorithm satisfies all of the above constraints. For example, consider the following Gauss-Jordan elimination algorithm executed in parallel[1](for brevity, the algorithm is modified to compute the determinant rather than the inverse of a given matrix):

```
{ FOR k = n STEP -1 UNTIL 2 DO
{ a_ij := a_ij × a_kk - a_kj × a_ik {1≤(i,j)≤k in
          parallel}
{ det := a_11
```

With $2n^2$ processors sharing a random access memory of an arbitrary size, this algorithm requires 2n-1 steps.[1] However, this is a misleading number if one considers the complexity of communication (or message switching) between the processors and the shared memory. To illustrate the point, consider the datum a_{kk} in the k-th step. It will be demanded by $(k-1)^2$ processors causing a memory contention problem. If the memory response time is proportional to the degree of contention, then the actual computation time will be of the order of $\sum_{k=2}^{n} 2(k-1)$, i.e. $O(n^2)$. Furthermore, the algorithm has a problem in its terminating condition. Without a centralized communication facility or without the knowledge of n, each processor cannot tell when to stop. The detection of n and its distribution to all processors will require O(3n) extra steps. For a recent analysis of data movement, see Gentleman[8].

Our algorithm requires n+1 steps. By the first n steps the cofactor matrix and the determinant will be computed simultaneously. By the last step the transposition and division operations will be done to get the inverse. During the first n steps, every processor in the toroid demands one datum each from its south, east and south-east (diagonal) neighbors. The communication time will be 4 times the unit of message transmission time between two immediate neighbors. Four arithmetic operations will be performed by every processor in each step. Therefore the computation time of each step is independent of n.

The last step consists of three phases: (1) to identify the diagonal processors with 2n delay, (2) to spread the ready signal from the diagonal processors to the processors in the same column with n delay, and (3) to migrate a datum southbound first to the diagonal processor then west-bound by the same distance requiring 2n delay at worst. The total computation time, therefore, will be proportional to (4+5+a)n, where a is a constant for the arithmetic operation delay time.

In order to apply the algorithm, the initial matrix is required to satisfy a stronger condition than that of non-singularity. We call such a matrix strongly non-singular. The notion will be defined in the next section. The foundation of our algorithm was known to Sylvester in 1851[4]. In the section 3 we will present it as the key lemma, and the algorithm itself

as the main theorem.

In the section 4, we will modify the algorithm for solving a system of linear equations in linear time.

2. Notations and Definitions

Let $A = (a_{ij})$, $1 \le (i,j) \le n$ be a $n \times n$ matrix on real numbers. We will use the following standard notations[2].

$A\begin{pmatrix} p_1,p_2,\ldots,p_r \\ q_1,q_2,\ldots,q_r \end{pmatrix}$, $1 \le (r,p_1,\ldots,p_r,q_1,\ldots,q_r) \le n$

: the $r \times r$ submatrix of A, consisting of $p_1,p_2,\ldots,p_r$-th rows and $q_1,q_2,\ldots,q_r$-th columns of A.

$\det(A)$: the determinant of A.

$\mathrm{adj}(A)$: the adjoint matrix of A.

A_{ij} : the cofactor of a_{ij}, $1 \le (i,j) \le n$, i.e. $\mathrm{adj}(A) = (A_{ji})$.

$A^{-1} \overline{\Delta} (\tilde{a}_{ij})$: the inverse matrix of A.

In the remainder of this paper, all matrix indices are to be interpreted as modulo n unless so stated otherwise.

Definitions:

(1) $B^k_{ij} \overline{\Delta} A\begin{pmatrix} i+1,i+2,\ldots,i+k \\ j+1,j+2,\ldots,j+k \end{pmatrix} 1 \le (i,j,k) \le n$

: the ij-th block of order k.

(2) A is strongly non-singular if every block of A is non-singular, i.e. for all $1 \le (i,j,k) \le n$, $\det(B^k_{ij}) \neq 0$.

Example: $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{bmatrix}$ is strongly non-singular.

Note that any non-singular Vandermonde matrix is strongly non-singular.

3. The Algorithm

In this section we will give the algorithm first in the form of a theorem, then illustrate the algorithm by a simple computation example. Then we will prove the theorem after presenting the key lemma with a proof.

Main Theorem:

Let A be a $n \times n$ strongly non-singular matrix. Define x^k_{ij} for $0 \le k \le n+1$, $1 \le (i,j) \le n$, by:

$$\begin{cases} x^0_{ij} \overline{\Delta} 1, \\ x^1_{ij} \overline{\Delta} a_{ij}, \\ x^{k+1}_{ij} \overline{\Delta} (x^k_{ij} \cdot x^k_{i+1,j+1} - x^k_{i+1,j} \cdot x^k_{i,j+1}) / x^{k-1}_{i+1,j+1}. \end{cases}$$

Then, for any $1 \le (i,j) \le n$,

(1) $x^{n+1}_{ij} = 0$.

(2) $x^n_{ij} = (-1)^{(n-1)(i+j)} \det(A)$.

(3) $x^{n-1}_{ij} = (-1)^n \cdot A_{i-1,j-1}$,
i.e. $\tilde{a}_{ij} = (-1)^{n(i+j)} x^{n-1}_{j+1,i+1} / x^n_{ij}$.

Example: Let $X^k \overline{\Delta} (x^k_{ij})$.

Let us apply the algorithm to the Vandermonde matrix given in the section 2.

$$X^0 = (x^0_{ij}) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$X^1 = (x^1_{ij}) = \begin{array}{cccc|c} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 & 1 \\ 1 & 3 & 9 & 27 & 1 \\ 1 & 4 & 16 & 64 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \end{array}$$

$$X^2 = (x^2_{ij}) = \begin{array}{cccc|c} 1 & 2 & 4 & -7 & 1 \\ 1 & 6 & 36 & -19 & 1 \\ 1 & 12 & 144 & -37 & 1 \\ -3 & -12 & -48 & 63 & -3 \\ \hline 1 & 2 & 4 & -7 & 1 \end{array}$$

$$X^3 = (x^3_{ij}) = \begin{array}{cccc|c} 2 & 12 & 22 & 12 & 2 \\ 2 & 48 & 52 & 18 & 2 \\ 6 & 72 & 114 & 48 & 6 \\ 6 & 48 & 84 & 42 & 6 \\ \hline 2 & 12 & 22 & 12 & 2 \end{array}$$

$$X^4 = (x^4_{ij}) = \begin{bmatrix} 12 & -12 & 12 & -12 \\ -12 & 12 & -12 & 12 \\ 12 & -12 & 12 & -12 \\ -12 & 12 & -12 & 12 \end{bmatrix}$$

$$(X^1)^{-1} = \begin{bmatrix} 48/12 & 72/-12 & 48/12 & 12/-12 \\ 52/-12 & 114/12 & 84/-12 & 22/12 \\ 18/12 & 48/-12 & 42/12 & 12/-12 \\ 2/-12 & 6/12 & 6/-12 & 2/12 \end{bmatrix}$$

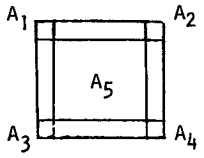
Key Lemma: (Sylvester)

Let A be an arbitrary $n \times n$ matrix.

Let $A_1 \overline{\Delta} A\begin{pmatrix} 1,2,\ldots,n-1 \\ 1,2,\ldots,n-1 \end{pmatrix}$ $A_2 \overline{\Delta} A\begin{pmatrix} 1,2,\ldots,n-1 \\ 2,3,\ldots,n \end{pmatrix}$

$A_3 \overline{\Delta} A\begin{pmatrix} 2,3,\ldots,n \\ 1,2,\ldots,n-1 \end{pmatrix}$ $A_4 \overline{\Delta} A\begin{pmatrix} 2,3,\ldots,n \\ 2,3,\ldots,n \end{pmatrix}$

$A_5 \overline{\Delta} A\begin{pmatrix} 2,3,\ldots,n-1 \\ 2,3,\ldots,n-1 \end{pmatrix}$



Then,

$$\det(A_5) \times \det(A) = \det(A_1) \times \det(A_4) - \det(A_3) \times \det(A_2).$$

Proof:

We will prove the lemma in two steps by showing:

(1) The lemma holds if A is almost diagonal (defined below).

(2) Any matrix A can be transformed into an almost diagonal one A' such that $\det(A) = \det(A')$, and $\det(A_r) = \det(A'_r)$ for $1 \le r \le 5$.

Call a matrix (a_{ij}) almost diagonal if $a_{ij} = 0$ for $1 \le (i,j) \le n$, except when $i=j$ or $i=1, j=n$ or $i=n, j=1$.

Step 1. Assume that A is an almost diagonal matrix.

$$A = \begin{array}{|c|c|c|} \hline a_{11} & 0 & a_{1n} \\ \hline & a_{22} \;\ddots & \\ 0 & \ddots\; 0 & 0 \\ & 0 \;\ddots\; a_{n-1,n-1} & \\ \hline a_{n1} & 0 & a_{nn} \\ \hline \end{array}$$

Then by row and column expansion, we get:

$$\det(A_1) = a_{11} \cdot \det(A_5),$$
$$\det(A_2) = (-1)^{n-2} \cdot a_{1n} \cdot \det(A_5),$$
$$\det(A) = a_{11} \cdot \det(A_4) + (-1)^{n-1} \cdot a_{1n} \cdot \det(A_3)$$
$$= a_{11} \cdot \det(A_4) - (-1)^{n-2} \cdot a_{1n} \cdot \det(A_3).$$

If $\det(A_5) = 0$, then $\det(A)$ and $\det(A_r)$, $1 \le r \le 5$, are zero, and the lemma holds. If $\det(A_5) \neq 0$, then by multiplying $\det(A_5)$ on the both sides of the last equation above, we obtain the lemma.

Step 2. By not using the 1-st and n-th rows for any row operation and the 1-st and n-th columns for any column operation, we can diagonalize A_5 without altering the values of $\det(A_r)$ for $1 \le r \le 5$, into the form:

$$\begin{array}{|c|c|c|} \hline t_{11} & t_{12} \cdots & t_{1n} \\ \hline t_{21} & t_{22} \;\ddots\; 0 & t_{2n} \\ \vdots & \ddots & \vdots \\ & 0 \;\ddots & \\ \hline t_{n1} & t_{n2} \cdots & t_{nn} \\ \hline \end{array}$$

By (n-2) row manipulations, we can eliminate $t_{12}, t_{13}, \ldots, t_{1,n-1}$. Similarly, we can eliminate $t_{n2}, t_{n3}, \ldots, t_{n,n-1}$. Then by (n-2) column operations, we can eliminate $t_{21}, t_{31}, \ldots, t_{n-1,1}$, and similarly, $t_{2n}, t_{3n}, \ldots, t_{n-1,1}$.

All row and column operations involved in the step 2 are from the inside of A_5 to the outside of A_5. Therefore they preserve the values of $\det(A_r)$ for $1 \le r \le 5$. Q.E.D.

Proof of Theorem:

Assume that A is strongly non-singular, i.e. for any $1 \le (i,j,k) \le n$, $\det(B^k_{ij}) \neq 0$, where B^k_{ij} is the ij-th block of order k of A.

First, by induction we will show that for any $1 \le k \le n$,

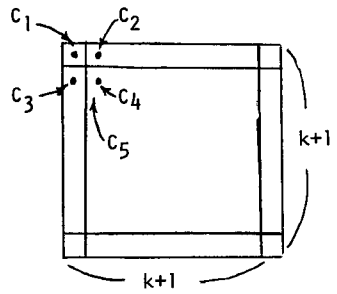
$$x^k_{ij} = \det(B^k_{i-1,j-1}), \text{ where } 1 \le (i,j) \le n.$$

Basis: Since $B^1_{i-1,j-1} = A\begin{pmatrix} i-1+1 \\ j-1+1 \end{pmatrix} = A\begin{pmatrix} i \\ j \end{pmatrix} = a_{ij}$,

$$x^1_{ij} = a_{ij} = \det(B^1_{i-1,j-1}).$$

Induction: Assume that for any $m \le k$, $x^m_{ij} = \det(B^m_{i-1,j-1})$. Let $C \triangleq B^{k+1}_{i-1,j-1}$, then using the notation introduced in the lemma,

$$C_1 \triangleq B^k_{i-1,j-1}, \quad C_2 \triangleq B^k_{i-1,j}, \quad C_3 \triangleq B^k_{i,j-1},$$
$$C_4 \triangleq B^k_{ij}, \quad C_5 \triangleq B^{k-1}_{ij}.$$



By the key lemma, $\det(B^{k-1}_{ij}) \cdot \det(B^{k+1}_{i-1,j-1})$

$$= \det(B^k_{i-1,j-1}) \cdot \det(B^k_{ij}) - \det(B^k_{i,j-1}) \cdot \det(B^k_{i-1,j}).$$

Since $\det(B^{k-1}_{ij}) \neq 0$, by the inductive hypothesis,

$$\det(B^{k+1}_{i-1,j-1}) = (x^k_{ij} \cdot x^k_{i+1,j+1} - x^k_{i+1,j} \cdot x^k_{i,j+1}) / x^{k-1}_{i+1,j+1}$$
$$= x^{k+1}_{ij}.$$

Proof of (2): Since $B^n_{i-1,j-1}$ is obtainable from $A(=B^n_{nn})$ by swapping rows $(n-1)(i-1)$ times, (from $(1,2,\ldots,n)$ to $(i,i+1,\ldots,n,1,\ldots,i-1)$), and swapping columns $(n-1)(j-1)$ times,

$$\det(B^n_{i-1,j-1}) = (-1)^{(n-1)(i-1)+(n-1)(j-1)} \det(B^n_{nn}).$$

Therefore, $x^n_{ij} = \det(B^n_{i-1,j-1})$

$$= (-1)^{(n-1)(i+j)} \cdot \det(A).$$

Proof of (1): Using the result of (2) above,

$$x^n_{ij} = (-1)^{(n-1)(i+j)} \cdot \det(A),$$
$$x^n_{i+1,j+1} = (-1)^{(n-1)(i+j+2)} \cdot \det(A),$$
$$x^n_{i+1,j} = (-1)^{(n-1)(i+j+1)} \cdot \det(A),$$
$$x^n_{i,j+1} = (-1)^{(n-1)(i+j+1)} \cdot \det(A).$$

By definition,

$$x^{n+1}_{ij} \cdot x^{n-1}_{i+1,j+1} = x^n_{ij} \cdot x^n_{i+1,j+1} - x^n_{i+1,j} \cdot x^n_{i,j+1}$$
$$= \det(A) \cdot \det(A) \cdot ((-1)^{2(n-1)(i+j)} - (-1)^{2(n-1)(i+j+1)})$$
$$= 0$$

Since $x^{n-1}_{i+1,j+1} = \det(B^n_{ij}) \neq 0$, $x^{n+1}_{ij} = 0$.

Proof of (3): Let c_{ij} be the ij-th minor of A, i.e.

$$c_{ij} \triangleq A\begin{pmatrix} 1,2,\ldots,i-1,i+1,\ldots,n \\ 1,2,\ldots,j-1,j+1,\ldots,n \end{pmatrix}$$

then $A_{ij} = (-1)^{(i+j)} \cdot \det(c_{ij})$, where $(A_{ji}) = \text{adj}(A)$.

On the other hand, by the definition of B^k_{ij},

$$B^{n-1}_{ij} = A\begin{pmatrix} i+1,i+2,\ldots,n,1,2,\ldots,i-1 \\ j+1,j+2,\ldots,n,1,2,\ldots,j-1 \end{pmatrix}.$$

Since B^{n-1}_{ij} is a row and column permutation of c_{ij},

$$\det(B^{n-1}_{ij}) = (-1)^{(n-2)(i-1)} \cdot (-1)^{(n-2)(j-1)} \cdot \det(c_{ij})$$

$$= (-1)^n A_{ij}.$$

Therefore, $x^{n-1}_{ij} = \det(B^{n-1}_{i-1,j-1}) = (-1)^n \cdot A_{i-1,j-1}$. Q.E.D.

4. Solution of Systems of Linear Equations

The results of the previous section can be immediately extended for a non-square matrix $A=(a_{ij})$, $1\le i\le m$, $1\le j\le n$, by interpreting the first index i to be modulo m, and the second index j to be modulo n. With the same definition of the ij-th block of order k as the one for square matrices, A is called strongly non-singular if $\det(B^k_{ij})\neq 0$ for any $1\le i\le m$, $1\le j\le n$, and $1\le k\le \min(m,n)$.

In this section, we will show that when the algorithm of the main theorem is applied to a strongly non-singular $n\times(n+1)$ matrix, representing a system of n linear rquations, it computes simultaneously all of the n+1 determinants required by the Cramer's method. The computation time is of the order of n.

Theorem:

Let A be a strongly non-singular $n\times(n+1)$ matrix,

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & a_{1,n+1} \\ a_{21} & a_{22} & \cdots & a_{2n} & a_{2,n+1} \\ \vdots & \vdots & & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & a_{n,n+1} \end{pmatrix}$$

and let $X^k \triangleq (x^k_{ij})$, $0\le k\le n$, be the results of iterations of the algorithm given in the main theorem applied to A.

Define $y_r \triangleq (-1)^{nr+1} x^n_{1,r+1}/x^n_{11}$, $1\le r\le n$.

Then,

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} a_{1,n+1} \\ a_{2,n+1} \\ \vdots \\ a_{n,n+1} \end{pmatrix}.$$

Example: $A = X^1 =$

2	-3	2	1	1	2
1	4	-2	7	-4	1
-3	1	1	-3	5	-3
5	-3	-1	2	-6	5
2	-3	2	1	1	2

$X^2 =$

11	-2	16	-11	9
13	6	-1	23	7
4	2	-1	8	7
-9	-9	-5	8	-17

$X^3 =$

23	47	51	71	40
2	-4	-5	21	21
6	19	16	32	1
-39	-77	-73	-115	-53

$X^4 =$

(-31)	(31)	(62)	(93)	(-31)
31	-31	-62	-93	31
-31	31	62	93	-31
31	-31	-62	-93	31

$y_1=(-1)^{4\times1+1}\cdot 31/(-31)=1$

$y_2=(-1)^{4\times2+1}\cdot 62/(-31)=2$

$y_3=(-1)^{4\times3+1}\cdot 93/(-31)=3$

$y_4=(-1)^{4\times4+1}\cdot(-31)/(-31)=-1$

$$\begin{bmatrix} 2 & -3 & 2 & 1 \\ 1 & 4 & -2 & 7 \\ -3 & 1 & 1 & -3 \\ 5 & -3 & -1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ -4 \\ 5 \\ -6 \end{bmatrix}$$

Proof: By the Cramer's rule, it is sufficient to show that

$y_r = \det(d_r)/\det(d_0)$, $1\le r\le n$, where

$$d_0 \triangleq A\begin{pmatrix} 1,2,\ldots,n \\ 1,2,\ldots,n \end{pmatrix},$$

$$d_r \triangleq A\begin{pmatrix} 1,2,\ldots\ldots\ldots\ldots\ldots,n \\ 1,2,\ldots,r-1,n+1,r+1,\ldots,n \end{pmatrix}.$$

By the definition of B^k_{ij},

$$B^n_{n,n+1} = A\begin{pmatrix} n+1,n+2,\ldots,n+n \\ n+2,n+3,\ldots,n+n+1 \end{pmatrix} = A\begin{pmatrix} 1,2,\ldots,n \\ 1,2,\ldots,n \end{pmatrix} = d_0,$$

$$B^n_{nr} = A\begin{pmatrix} n+1,n+2,\ldots,n+n \\ r+1,r+2,\ldots,r+n \end{pmatrix}$$

$$= A\begin{pmatrix} 1,2,\ldots\ldots\ldots\ldots,n \\ r+1,\ldots,n,n+1,1,2,\ldots,r-1 \end{pmatrix}.$$

(Note that the second index j is to be modulo n+1.)

Since d_r is a column permutation of B^n_{nr},

$$\det(d_0) = \det(B^n_{n,n+1}),$$

$$\det(d_r) = \det(B^n_{nr}) \times (-1)^{(n-r+1)(r-1)+(n-r)}$$

$$= \det(B^n_{nr}) \times (-1)^{nr+1}.$$

By the similar argument to the one given in the section 3,

$$x^k_{ij} = \det(B^k_{i-1,j-1}),$$

therefore,

$$y_r = (-1)^{nr+1}\cdot x^n_{1,r+1}/x^n_{11}$$

$$= (-1)^{nr+1}\cdot\det(B^n_{nr})/\det(B^n_{nn})$$

$$= \det(d_r)/\det(d_0).$$ Q.E.D.

References

(1) Borodin, A. & Munro, I.
The Computational Complexity of Algebraic and Numeric Problems.
American Elsevier Publishing Company, Inc. 1975.

(2) Faddeev, D.K. & Faddeeva, V.N.
Computational Methods of Linear Algebra.
W.H. Freeman and Company, 1963.

(3) Strassen, V.
Gaussian elimination is not optimal.
Numerische Mathematik, 13, 354-356(1969).

(4) Sylvester, J.J.
Philosophical Magazine 4(1851). 295-305.

(5) Van Scoy, F.L.
Some parallel cellular matrix algorithms.
1977 ACM Computer Science Conference, Atlanta Georgia.

(6) Pease, M.C.
Matrix inversion using parallel processing.
JACM 14,4; 757-764(1967).

(7) Csanky, L.
Fast parallel matrix inversion algorithms.
SIAM J. Computing, 5,4; 618-623(1976).

(8) Gentleman, W.M.
Some complexity results for matrix computations on parallel processors.
JACM 25,1; 112-115(1978).

Acknowledgment

The authors express their appreciation to Marcel Neuts and Hatem Khalil of the University of Delaware for their comments, suggestions and criticisms which aided to guide this research in the right direction. They are also grateful to Ramaswami Vaidyanathan of the University of Delaware in his contribution to the proof of the key lemma. The authors appreciate referees' comments. They are grateful to Ms. Beverly Crowl for manuscript typing.