CONSTRUCTIVE THEORY
OF FUNCTIONS, Varna '91
Sofia, 1992, pp. 193-202

REAL-NUMBER CODES FOR FAULT-TOLERAT MATRIX INVERSION ON PROCESSOR ARRAYS

E.I.Milovanovic, M.K.Stojcev, I.Z.Milovanovic

Faculty of Electronic Engineering, Beogradska 14, P.O.Box 73,

18000 *Nis, Yugoslavia*

In memory to prof Vasil Popov

ABSTRACT

A method for inversion of triangular matrix based on linear-checksum approach is described in this paper. Iterative Shultz' method adapted for parallel implementation on processor array is used for matrix inversion. The architecture of the system is given also.

1. INTRODUCTION

A number of techniques have been proposed for introducing redundancy in VLSI/WSI arrays. Redundancy is used in VLSI arrays for two purposes, first to increase the yield at manufacture and second, to improve the performance and availability of the array during operation [1]. The production of VLSI/WSI parallel system, usually realized as a processor array (PA), is greatly affected by with the production failures. The application areas of PA demand a large degree of reliability of the computed results. However, the probability of errors in the result increases with the amount of computation. In order to accommodate the contradictory requirements, high complexity and high reliability, the system has to be designed to be fault tolerant (FT). FT strategies can be designed to deal with three distinct types of failures which can occur during: Fabrication time, compile time and run time. In the focus of our interest are the run-time errors. Conventional FT techniques for run-time errors such as triple modular redundancy (TMR), triple time redundancy (TTR), recomputing with shifted operands (RESO), and totally self-checking circuits (TSC), suffer from either a high hardware overhead (cost), or time overhead (degraded performance) [2]. The technique called algorithm-based fault-tolerant (ABFT) gives a compromise.

In this paper a method for inversion of triangular matrix based on linear-checksum approach is proposed. To compute an inverse matrix we use iterative Shultz' method, which in the case of triangular matrix gives the solution in a finite number of iteration steps. The architecture of the system for linear-checksum matrix inversion is proposed also. Detection and correction of a single error in PA is considered.

2. MATRIX INVERSION

The fundamental direct method for determination an inverse matrix of regular square matrix A_{nxn}, is based on Gaussian method of elimination [3]. Namely, the Gaussian method of elimination is applied on the extended matrix $(A|I)$ which is, after finite number of steps, transformed into matrix $(I|A^{-1})$. In [4] this method was proposed for weighted-matrix inversion.

From the iterative methods for matrix inversion we will point out to the Sultz' one, because it will be used in this paper. It is defined with the following theorem:

Theorem 1. Let for given matrix X_0 the following inequality

$$\| I - A \cdot X_0 \| \leq q < 1$$

is valid. Then a sequence $(X_k)_{k \in N_0}$, defined by

$$X_{k+1} = (2I - X_k A)X_k, \qquad k = 0,1,\ldots$$

converges towards A^{-1}.

More details related to Shultz' and other similar methods can be found in [3,5]. Let us point out that Shultz' method is especially suitable when matrix A is a triangular one. In that case, for suitable defined initial matrix X_0, an inverse matrix A^{-1}, is obtained in finite number of iteration steps [5].

2.1. *Linear-checksum approach*

In [4,6,7] real number codes such as checksum and weighted-checksum codes have been proposed for fault-tolerant matrix operations such as matrix transposition, addition, multiplication, LU decomposition and matrix inversion. Weighted checksums have been shown to have a problem with floating point and fixed-point arithmetic in that to do checking, a tolerance level must be chosen to take account of the rounding errors which arise during computation [8]. The overflow problem due to exremly large weights can be overcame using linear codes. In [2] the selection of a particular code from

the general set of real number codes in order to reduce numerical errors during encoding was proposed.

2.2. *The proposed method*

The concentration of this paper is on developing algorithm for linear-checksum inversion of triangular matrices L and U. The inversion is based on Shultz' approach.

A procedure for linear-checksum matrix inversion of regular square matrix A_{nxn} can be decomposed into three steps:

1. Linear-checksum LU decomposition of matrix A,
2. Linear-checksum inversion of triangular matrices L and U,
3. Linear-checksum multiplication of triangular matrices.

PHASE 1.

Fault tolerant arrays for matrix product and LU factorisation using the triangular arrays were considered in [9]. A methodology which maps a matrix multiplication algorithm to a fault tolerant array processor with different topologies and dimensions was considered in [10].

PHASE 2.

In this phase a linear-checksum matrix inversion of matrices L and U are performed. The idea is to compress the information contained in the row/column elements of a matrix into a single element which is called a check element. Information is compressed in such a way that it is preserved during the computations for which the code is used. The method for linear-checksum triangular matrix inversion, that we propose in this paper, uses the following well-known apparatus [2].

If $\vec{a}^T=[a_1,\ldots,a_n]$ is an arbitrary vector and $\vec{e}^T=[c_1,\ldots,c_n]$ is linear checksum vector, than coded vector $\vec{D}$ has the following form

$$\vec{D}=[a_1,\ldots,a_n,CS]$$

where

$$CS=\vec{a}^T\vec{e}^T=[a_1,\ldots,a_n][c_1,\ldots,c_n]=a_1c_1+\ldots+a_nc_n.$$

Similarly, the row, column and full checksum matrices of an nxn matrix $A=(a_{ij})$ are defined as follows:

$$A_c=\begin{bmatrix} A \\ \vec{e}^T A \end{bmatrix}$$

$$A_r=[\ A\ \ A\vec{e}\ ]$$

$$A_f=\begin{bmatrix} A & A\vec{e} \\ \vec{e}^T A & \vec{e}^T A\vec{e} \end{bmatrix}.$$

Now, we will describe the procedures for calculating checksum inverse matrix of matrices L and U. We assume that matrix U is with unit diagonal.

a) Computing U_f^{-1}

a.1 $X^{(0)}= diag(1,1,\ldots,1)_{nxn}$ $\quad (X^{(0)}=I)$

a.2 $R_c^{(0)}=I_c-X_c^{(0)}U$

a.3 $P_c^{(1)}=I_c+R_c^{(1)}$

a.4 $X_f^{(i+1)}=P_c^{(i)}X_r^{(i)}$ $\quad i=1,2,\ldots,k$

a.5 $R_f^{(i+1)}=R_c^{(i)}R_r^{(i)}$

After k iteration steps we obtain $X_f^{(k+1)}=U_f^{-1}$.

b) Computing L_f^{-1}

b.1 $Y^{(0)}=diag(\frac{1}{l_{11}},\ldots,\frac{1}{l_{nn}})_{nxn}$

b.2 $R_c^{(0)}=I_c-Y_c^{(0)}L$

b.3 $P_c^{(1)}=I_c+R_c^{(1)}$

b.4 $Y_f^{(i+1)}=P_c^{(i)}Y_r^{(i)}$ $\quad i=1,2,\ldots,k$

b.5 $R_f^{(i+1)}=R_c^{(i)}R_r^{(i)}$

After k iteration steps we obtain $Y_f^{(k+1)}=L_f^{-1}$.

In both cases, a) and b), the matrix $R^{(0)}$ is strictly triangular matrix of order nxn, for which the following is valid:

i) $(R^{(0)})^n = 0$, and

ii) $R^{(k)}= (R^{(k-1)})^2 = (R^{(0)})^2$.

So, in both cases, a) and b), the number of iteration steps k is the smallest integer that satisfies the inequality

(1) $$2^k \geq n.$$

When inequality (1) is satisfied then $R^{(k)}= 0$, and $R_c^{(k)}=0$. According to that it follows that $X^{(k+1)}= X^{(k)}$, i.e. $X_r^{(k+1)}=X_r^{(k)}$ Since $P_c^{(k)}=P_c^{(k+1)}=I_c$, it follows that

$$X_f^{(k+2)}=P_c^{(k+1)}X_r^{(k+1)}=I_c X_r^{(k)}=X_f^{(k+1)} .$$

This means that iterative process a.4 reaches fixed point for final number of iteration steps, k. The same conclusion is valid for iterative process b.4.

In the third phase linear-checksum inverse matrix A_f^{-1} of matrix A is obtained according to the product $U_c^{-1}\cdot L_r^{-1}$.

Let us note that with linear checksums only a single error in whole processor array can be detected and corrected, while with weighted checksums a single error in each row (column) of processor array can be detected and corrected [2,11]. Exploitation experience with PA implies that single error are dominant over multiple errors, i.e. more than 90% of all errors are single.

3. HARDWARE REALIZATION

The processor array for linear checksum matrix inversion for matrix of order nxn, consists of n(n+1)/2+2n+1 PEs. The structure of PA,for the case n=4, is represented in Fig.1. PA consists of PEs of two types. PEs represented by squares are intended for matrix body calculations, while PEs denoted as double squares are used for calculating checksum elements.

According to the described mathematical model single error detection and correction is performed after every iteration step a.4 and a.5 (i.e. b.4 and b.5). The structure of PA and input data formats for calculating U_f^{-1} are shown in Fig.2.a and 2.b. Fig.2.a (2.b) corresponds to input data formats for calculating matrix $X_f^{(l+1)}$ ($R_f^{(l+1)}$) according to the step a.4 (a.5).

The host computer performs following activities:

a) Calculates steps defined by a.1, a.2 and a.3.

b) Carries out I/O.

c) Detects and corrects a single error in PA.

Detection and correction of a single error is performed in the following way:

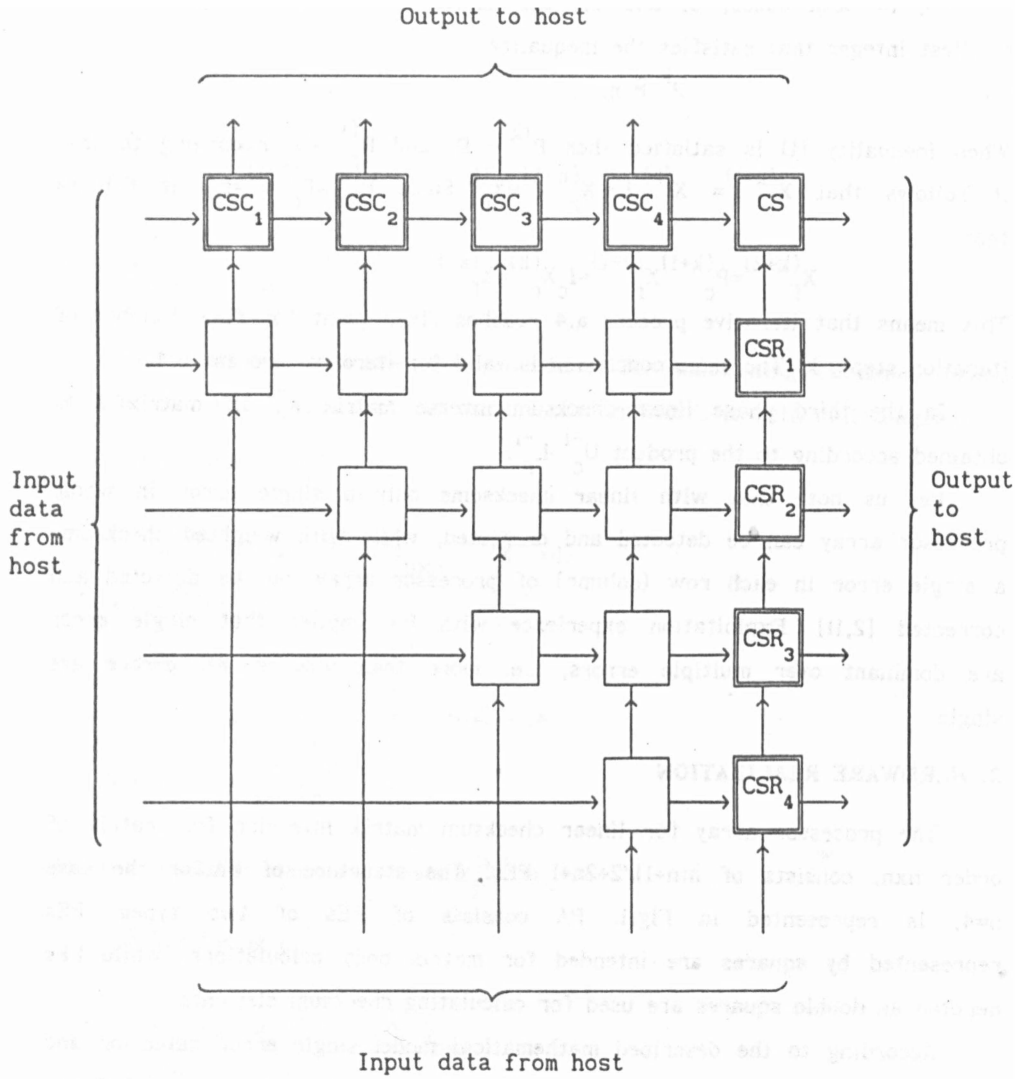
Output to host
CSC1
CSC2
CSC3
CSC4
CS
CSR1
CSR2
CSR3
CSR4
Input data from host
Output to host
Input data from host

Fig. 1.

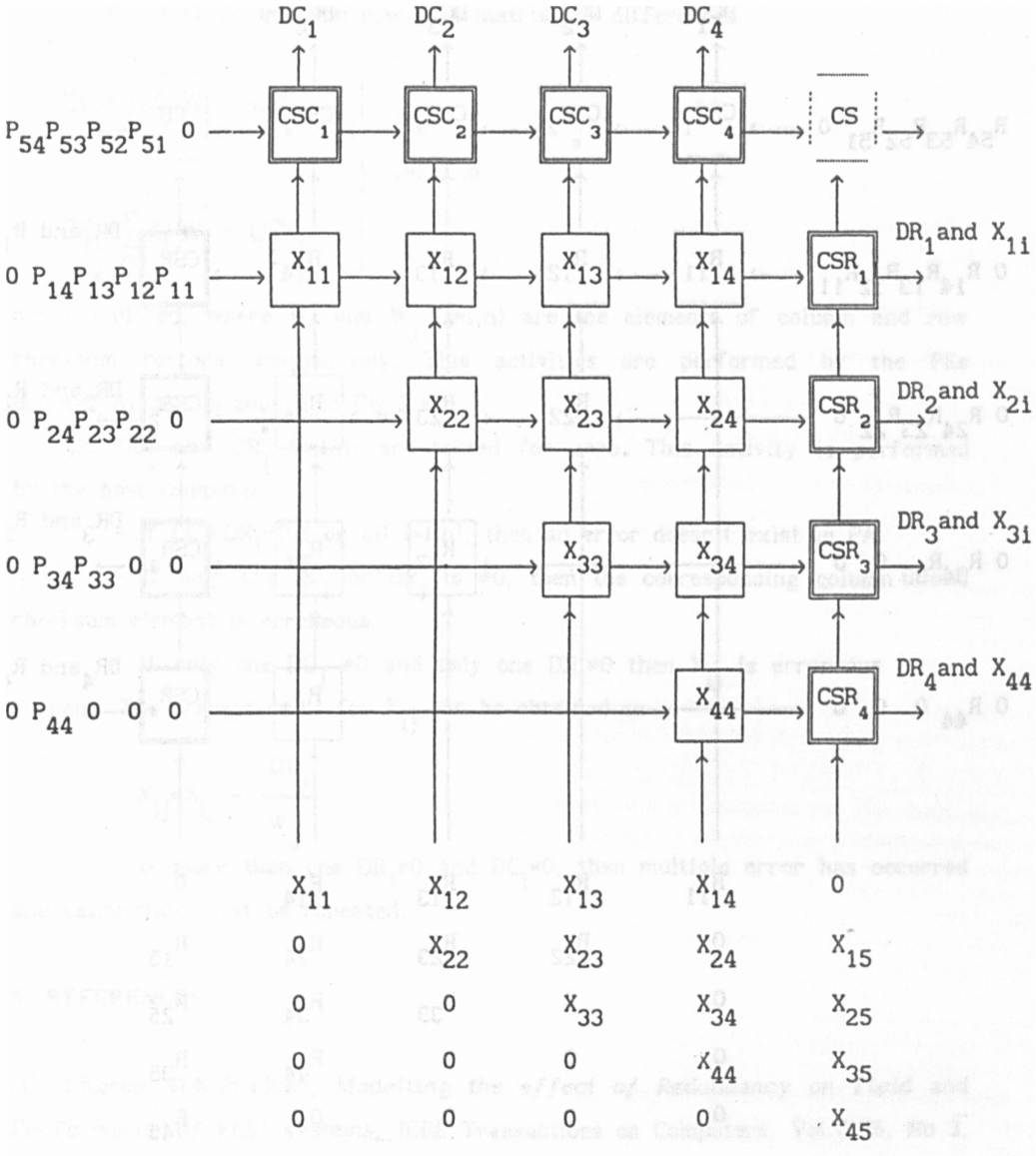
DC1
DC2
DC3
DC4
P54P53P52P51 0
CSC1
CSC2
CSC3
CSC4
CS
0 P14P13P12P11
X11
X12
X13
X14
CSR1
DR1and X11
0 P24P23P22 0
X22
X23
X24
CSR2
DR2and X21
0 P34P33 0 0
X33
X34
CSR3
DR3and X31
0 P44 0 0 0
X44
CSR4
DR4and X44
X11 X12 X13 X14 0
0 X22 X23 X24 X15
0 0 X33 X34 X25
0 0 0 X44 X35
0 0 0 0 X45

Fig. 2. a.

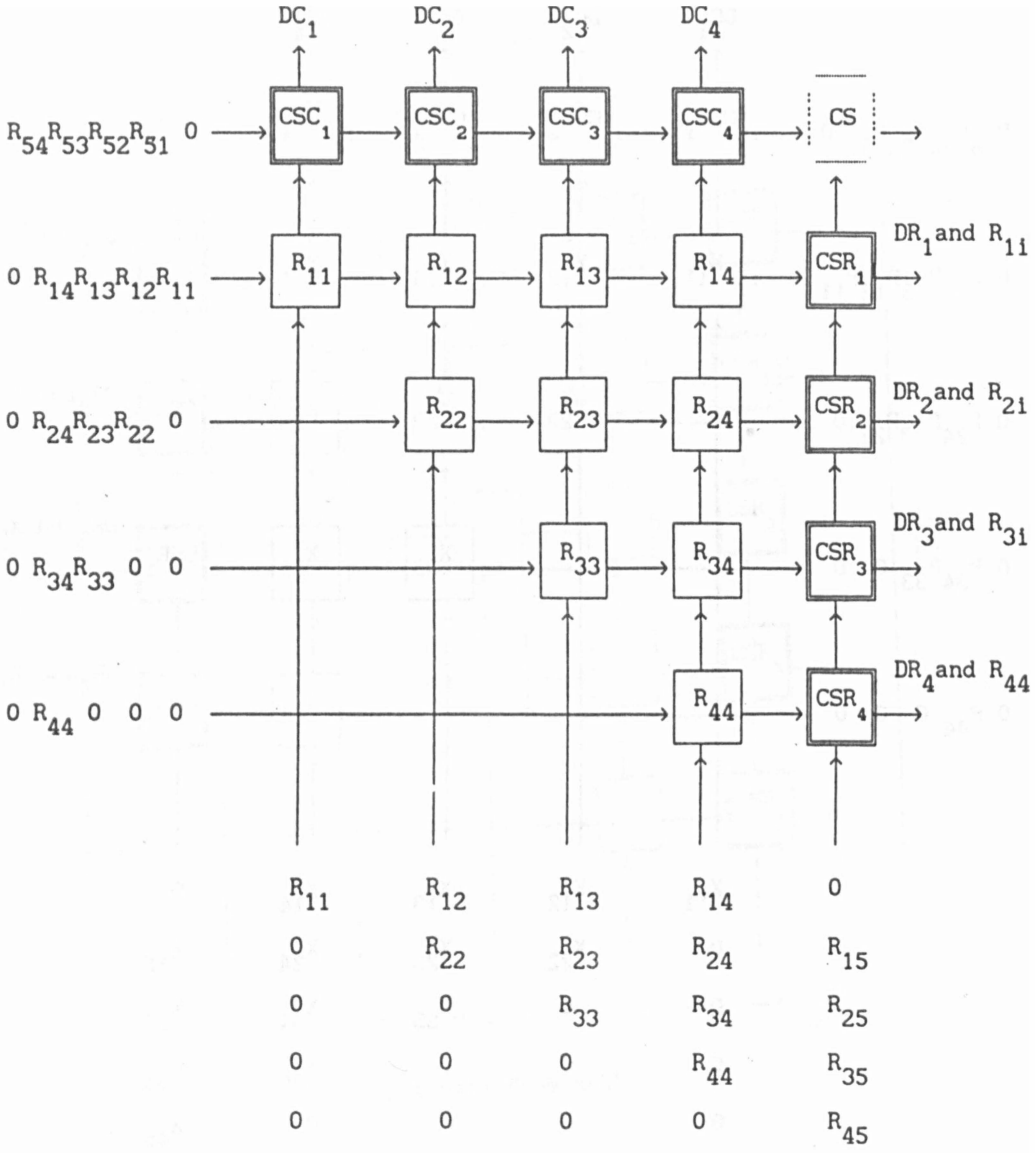
DC1
DC2
DC3
DC4
R54R53R52R51 0
CSC1
CSC2
CSC3
CSC4
CS
0 R14R13R12R11
R11
R12
R13
R14
CSR1
DR1 and R11
0 R24R23R22 0
R22
R23
R24
CSR2
DR2 and R21
0 R34R33 0 0
R33
R34
CSR3
DR3 and R31
0 R44 0 0 0
R44
CSR4
DR4 and R44
R11 R12 R13 R14 0
0 R22 R23 R24 R15
0 0 R33 R34 R25
0 0 0 R44 R35
0 0 0 0 R45

Fig. 2. b.

1. For each column and row of a matrix the differences

$$\left.\begin{aligned} DC_i &= \sum_{j=1}^{n} X_{ji} W_j - CSC_i \\ DR_i &= \sum_{j=1}^{n} X_{ij} W'_j - CSR_i \end{aligned}\right\} \quad i=1,\ldots,n$$

are calculated, where W_j and W'_j (j=1,n) are the elements of column and row checksum vectors, respectively. This activities are performed by the PEs denoted by double square in Fig.2.a.

2. DC_i and DR_i (i=1,n) are tested for zero. This activity is performed by the host computer.

- If $DC_i=DR_i=0$ (for all i=1,n), then an error doesn't exist in PA.
- If only one DC_i or DR_i is $\neq 0$, then the corresponding column (row) checksum element is erroneous.
- If only one $DC_j \neq 0$ and only one $DR_i \neq 0$ then X_{ij} is erroneous element. The correct value for X_{ij} can be obtained as

$$X_{ij}=X_{ij} - \frac{DR_i}{W'_j}$$

- If more than one $DR_i \neq 0$ and $DC_i \neq 0$, than multiple error has occurred and calculation must be repeated.

5. REFERENCES

[1]. I.Koren, D.K.Pradhan, *Modelling the effect of Redundancy on Yield and Performance of VLSI systems*, IEEE Transactions on Computers, Vol.C.36, No 3, (1987), 344-355.

[2]. V.S.S.Nair, J.A.Abraham, *Real-Number Codes for Fault-Tolerant Matrix Operations on Processor Arrays*, IEEE Transaction on Computers, Vol.39, No 4, (1990), 426-435.

[3]. G.V.Milovanovic, *Numerical Analysis*, Naucna Knjiga, Beograd, 1988. (In Serbian).

[4]. J.-Y.Jou, J.A.Abraham, *Fault-tolerant matrix arithmetic and signal processing on higly concurrent computing structures*, Proc. IEEE Vol.74, (1986), 732-741.

[5]. R.Schreiber, *Block Algorithms for Parallel Machines*, Technical Report, RPI Troy, New York 12180-3590, No 5 (1987), 1-16.

[6]. J.A.Abraham, P.Banerjee, C.-Y.Chen, W.K.Fuchs, S.-Y.Kuo, A.L.N.Reddy, *Fault tolerance for systolic arrays*, IEEE Comput., Vol.20, No 7 (1987),65-74.

[7]. K.H.Huang, J.A.Abraham, *Algorithm-based fault tolerance for matrix operations*, IEEE Trans.Comput., Vol. C-33 (1987), 518-528.

[8]. C.J.Anfinson, F.T.Luk, *A linear Algebraic Model of Algorithm-Based Fault Tolerance*, IEEE Transaction on Computers, Vol.37, No 12 (1988), 1599-1604.

[9]. G.M.Megson, D.J.Evans, *Algorithmic fault tolerance for matrix operationson triangular arrays*, Parallel Comput. 10 (1989), 207-219.

[10]. C.-M.Liu, C.-W.Jen, *Design of algorithm based fault tolerant VLSI array processor*, IEE Proc., Vol 136, Pt.E. No 6 (1989), 539-542.

[11]. M.K.Stojcev, E.I.Milovanovic, I.Z.Milovanovic, *Algorithm Based Fault-Tolerant Technique for Matrix Inversion*, International Conference PaCT-91, Novosibirsk 1991.