COMPUTING SIZE-INDEPENDENT MATRIX PROBLEMS ON SYSTOLIC ARRAY PROCESSORS

Juan J. NAVARRO
José M. LLABERIA
Mateo VALERO

RR85/17

COMPUTING SIZE-INDEPENDENT MATRIX PROBLEMS

ON SYSTOLIC ARRAY PROCESSORS

J.J. Navarro
J.M. LLaberia
M.Valero

Computer Architecture Department
Facultad de Infomática
Pau Gargallo 5.
08028 BARCELONA (SPAIN)

Phone: 34-3-3338416
Telex: 52821 upc e

This work was supported by the Ministery of Education of Spain (CAYCIT) under Grant Number 2906-83 C03-03.

COMPUTING SIZE-INDEPENDENT MATRIX PROBLEMS ON SYSTOLIC ARRAY PROCESSORS

ABSTRACT

A methodology to transform dense to banded matrices is presented in this paper. This transformation, is accomplished by triangular blocks partitioning, and allows the implementation of solutions to problems with any given size, by means of contraflow systolic arrays, originally proposed by H.T. Kung. Matrix-vector and matrix-matrix multiplications are the operations considered here.

The proposed transformations allow the optimal utilization of processing elements (PE's) of the systolic array when dense matrix are operated. Every computation is made inside the array by using adequate feedback. The feedback delay time depends only on the systolic array size.

1.- INTRODUCTION

The systolic array processors allow to obtain very high throughput from the high degree of parallelism and pipelining they can support. Most part of the systolic systems so far developed, are tailored to some applications. A particular design is made to meet one (or several related) algorithm(s) and to suit the size of a given data structure size.

The situation in many practical cases is that the number of PE's and the interconnection topology are fixed, but several similar problems with dimensional variations are to be solved in the systolic array processor. In such cases, some transformations of the original data structures are needed.

In order to minimize the global computational time it is of great importance a) to have data transformations with low generation difficulties, b) to reach a maximum operations rate in the array, and c) to get a simple attainment of final results from the partial values computed inside the array system. Clearly matrix calculations belong to an application field that requires this kind of transformations (1).

Several authors have focussed attention on this problem. K. Hwang and Y.H. Cheng have worked on this direction (2), proposing matrix partitioning for VLSI arithmetic systems. H.Y. Chuang and G. He presented in (3) a design methodology of problem size independent systolic array systems, taking into account algorithms without data contraflow. For banded matrix operations, good efficiency is achieved with the contraflow systolic arrays proposed by H.T. Kung (4,5), but these systems suffer a throughput decrease when dense matrices are operated. Based upon Kung's design, R.W. Priester and others (6) present a matrix transformation yielding to a 50% size reduction of the systolic array, with no overhead in the algorithm time under certain conditions.

To evaluate the system efficiency, the utilization factor of each PE in the array, can be used. This measure, η, is expressed as N/AT, where N is the number of operations required by the algorithm, A is the number of PE's in the array, and T is the number of steps needed to execute the algorithm in the system.

In the present work, we propose a method to transform dense matrices of any dimension into banded matrices. The transformed matrix may have variable bandwidth, so that an easy and adequate matching to the dimensions of Kung's systolic arrays is achieved. Maximum efficiency is obtained because every array operation cycle is useful, due to the fact that the transformed matrix band is filled (no empty position) with elements from the original matrix. The data transformation is simple enough, and no computation outside the array is needed. Feedback of partial results obtained inside the system, is provided. This fact yields to minimize the algorithm's execution time.

In section 2 two types of data transformation are proposed and developped. We shall name these transformations as DBT (**D**ense to **B**anded matrix transformation by **T**riangular blocks partitioning). To solve the problem of Matrix-Vector multiplication, an algorithm DBT by-rows is proposed in this section. Another algorithm, DBT-transposed-by-rows, is used in section 3, to solve the problem of Matrix-Matrix multiplication

2.- MATRIX-VECTOR MULTIPLICATION.

In what follows, we suppose A to be the original n*m matrix and **A** is the transformed banded matrix, with bandwith w equal to the array size.

The general method that we are proposing, to map A into **A**, is based upon the three following points (see fig.1):

a) to split the original matrix, A(n,m), into $k_n k_m$ submatrices $A_{ij}(w,w)$, where $k_n = \lceil \frac{n}{w} \rceil$ and $K_m = \lceil \frac{m}{w} \rceil$. When n and/or m are not integer multiples of w, A is extended with zero-valued elements in rows and/or columns.

b) every submatrix $A_{ij}(w,w)$ is, in turn, split into triangular submatrices. Let us call them U_{ij} (upper) and L_{ij} (lower). The main diagonal of A_{ij} may belong to any of them. Let us suppose, without lack of generality, that it belongs to U_{ij}.

c) the resulting banded matrix, **A**, is formed by submatrices U_k and L_k. We obtain this matrix if submatrices U_{ij} and L_{rs} are appended together inside the band.

Several ways to obtain **A** may be devised. Of greater interest will be those leading to raise the efficiency in the global proceeding of transformation, operation, and attainment of resulting values.

Let us assume that we need to solve the operation y=Ax+b, where A is an n*m matrix, and m is the dimension of vectors y and x. The original computation must be accomplished by means of the transformed operation **y=Ax+b**.

In the fig. 1.a we show a diagram of this operation to be performed, together with the A matrix decompositions into triangular submatrices U_{ij} and L_{ij}, and of the x,b and y vectors. The x, b and y vectors are split into k_m, k_n and k_n sub-vectors, respectively, all with w elements.

The transformation of this problem to one with banded matrices can be seen in fig. 1.b. We have assumed, without loss of generality, that the transformed matrix is of the upper-band type (A_{ij} = 0 for i>j). A lower band transformed matrix could be considered in a similar way (A_{ij} = 0 for i<j).

The following conditions are to be satisfied, in order

to obtain the required matrix transformation:

1.- For $0 \leqslant k \leqslant k_n k_m$, if $\mathbf{U}_k$ is equal to U_{ij}, $\mathbf{L}_k$ must then be equal to L_{ip} for every p such that $0 \leqslant p < k_m$.

2.- For $0 \leqslant k < k_n k_m - 1$, if $\mathbf{L}_k$ is equal to U_{ij}, $\mathbf{U}_{k+1}$ must then be equal to U_{pj} for every p such that $0 \leqslant p < k_n$.

3.- Only one single copy of the original submatrices U_{ij} and L_{ij} must exist in the **A** matrix.

The transformed vector, **x**, must be composed by $k_n k_m + 1$ sub-vectors ($\mathbf{x}_0$, $\mathbf{x}_1, \ldots, \mathbf{x}_{k_n k_m}$) such that the $k_n k_m$ first of them have w elements and the last one, $\mathbf{x}_{k_n k_m}$, has w-1 elements. The total number of elements in **x** is $k_n k_m w + w - 1$.

The **b** and **y** vectors are formed by $k_n k_m$ sub-vectors, each with w elements. The transformed vectors, **x**, **b** and **y**, have its structure dependent on the chosen transformation algorithm.

The rules defining the correspondence between the sub-vectors of **x**, **b** and **y** and the x_i, b_i, y_i sub-vectors of x, b and y, are as follows:

1) For $0 \leqslant k < k_n k_m$, if $\mathbf{U}_k = U_{ij}$ then

$$\mathbf{x}_k = x_j$$

$$\mathbf{b}_k = \begin{cases} b_i & \text{if } k = \min(I_i) \\ y_i^R & \text{otherwise, where } R = \max(J_i^k) \end{cases}$$

$$y_k = \begin{cases} y_i & \text{if } k=\max(I_i) \\ y_i^R & \text{otherwise} \end{cases}$$

I_i and J_i^k are the set of indices

$$I_i = \{ q \in \mathbb{N} \quad U_q = U_{ip}, \text{ with } 0 \leqslant p < k_m \}$$

$$J_i^k = \{ q \in \mathbb{N} \quad y_p = y_i^q, \text{ with } 0 \leqslant p < k \}$$

2) if $L_{k_n k_m - 1} = L_{ij}$ then $x_{k_n k_m} = x'_j$
where x'_j is the sub-vector formed by the (w-1) first elements of the x_j block.

Note that data from the original b_i vector, as well as previously computed partial results, y^R_i, are inputs to the array system. By using this type of feedback, final results are obtained without need of any calculation external to the array processor.

We can see, from the expressions above, that several optimal DBT transformations can be devised. We refer to the optimality with regard to the required computational time. From those transformations, in this section we choose and present now one that allows simple and regular transformed structures, and with a constant time value of the required feedback. A **DBT transformation by rows**, accomplishing the above stated requirements, will be presented in the following paragraphs.

The rules to define such a transformation are as follows:

a) Attainment of **A** from A.

For $0 \leqslant k < k_n k_m$

$$U_k = U_{r,s} \quad \text{with } r = \left\lfloor \frac{k}{k_m} \right\rfloor \text{ and } s = k \bmod k_m$$

$$L_k = L_{r,s} \quad \text{with } r = \left\lfloor \frac{k}{k_m} \right\rfloor \text{ and } s = (k \bmod k_m + 1) \bmod k_m$$

b) Attainment of **x**, **b**, and **y** from x, b, and y.

For $0 \leqslant k < k_n k_m - 1$

$$\mathbf{x}_k = x_{k \bmod k_m} \qquad \text{and} \qquad \mathbf{x}_{k_m k_n} = x'_0$$

$$\mathbf{b}_k = \begin{cases} b_{k/k_m} & \text{if } k \bmod k_m = 0 \\ y^{k-1}_{k/k_m} & \text{otherwise} \end{cases}$$

$$\mathbf{y}_k = \begin{cases} y_{k/k_m} & \text{if } (k+1) \bmod k_m = 0 \\ y^{k}_{k/k_m} & \text{otherwise.} \end{cases}$$

The PRT transformation proposed by R.W. Priester et al. (6) is a particular case of the DBT transformation by rows when $k_n = k_m = 1$.

In this transformation, the number of steps to have the required feedback equals the array size, w, and can be implemented with w registers. This implementation is very

modular and easily expandible.

Let us think now of a particular and practical case, with n=6, m=9 and w=3. The block level original data structures can be seen in fig. 2.a; the corresponding transformed data structures are shown in fig. 2.b. In fig. 3 the data coming in and out of the systolic array in every one of the 39 required computational cycles, are shown.

The value of the PE's utilization can be raised 100% by grouping every 2 PE's in 1, or overlapping the execution of several problems, or partitioning the transformed problem into two disjoint sub-problems. The dotted line in fig. 2.b shows the optimal partitioning for the concrete case we are considering.

In a general case, the number of steps required to solve the problem, with no overlapping is: $T = 2w + 2k_n k_m w - 3$

If overlapping is used, the number of steps is
$T = 2w+2k_n k_m w - \alpha$.

$$\text{with } \alpha \begin{cases} = 2 & \text{if } k_n k_m \mod 2 = 0 \\ = 3 & \text{otherwise} \end{cases}$$

The processor utilization, , is given by

$$\frac{n.m}{AT} = \frac{1}{2+\frac{2}{k_n k_m} - \frac{3}{k_n k_m w}} \quad \text{with no overlapping}$$

$$\text{and} \quad \frac{1}{1+\frac{2}{k_n k_m} - \frac{\alpha}{k_n k_m w}} \quad \text{with overlapping}$$

When the value of the product $k_n k_m$ is large, the PE's utilization aproaches to 1/2 and 1, respectively.

In the next section beside the above presented transformation, another one is to be used. This new transformation, named **DBT transposed by rows**, yields to attainment of a lower-band transformed matrix. The method consists in transposing the matrix resulting from the application of a DBT by rows transformation to the transposition of the original matrix; that is:

$$DBT_{\text{transposed-by-rows}}(A) = DBT_{\text{by-rows}}(A^T)^T$$

3.- MATRIX-MATRIX MULTIPLICATION.

In this section we are directed to solve the problem C=A*B; A, B and C are matrices of (n,p), (p,m) and (n,m) dimensions respectively. k_n, k_p and k_m are the coefficients that relate the problem and array dimensions, as follows:

$$k_n = \left\lceil \frac{n}{w} \right\rceil \quad , \quad k_p = \left\lceil \frac{p}{w} \right\rceil \quad \text{and} \quad k_m = \left\lceil \frac{m}{w} \right\rceil$$

To achieve this problem's solutions, the B matrix is divided in column submatrices of width w. The C matrix is attained by succesively multiplying the A matrix by each column submatrix of B matrix. By using this algorithm, the transformed problem is now **C=A*B** (see fig. 4.a), and the matrices A and B are calculated as follows:

1. Algorithm to obtain the transformed matrix **A** can be expressed as follows:

a) To apply the transformation **DBT by-rows** to matrix a. this step yields to matrix a^b.

b) To juxtapose k_m blocks A^b and one triangular block U'. This step yields to matrix **A**. The block U' is composed by the first (w-1) rows and (w-1) columns of A^b.

Consequently, **A** is a square matrix of dimension $k_p k_n k_m + w - 1$

2. The matrix **B** has the same dimension as **A**. An algorithm to obtain the transformed matrix **B** can be expressed as follows:

a) The B matrix is divided into k_m column sub-matrices, with p*w elements each. Let us name these sub-matrices by $B_0, B_1, \ldots, B_{k_m-1}$.

b) A **DBT transposed-by-rows** is applied to each B_i. This step yields to a banded matrix, B_i^b.

c) By juxtaposition of k_n blocks B_i^b, the banded matrix B_i^d is attained.

d) By juxtaposition of k_m matrices $B^d_0, B^d_1, \ldots, B^d_{k_n-1}$

and appending at the end the main sub-matrix L', the **B** matrix is attained. L' is formed by the (w-1) rows and (w-1) columns of B^b_0.

Now, we will describe the process of attainment of matrix C, starting from the partial results produced by the systolic array system, when it operates to make the multiplication **C=A*B**.

The partial results matrix, C, is shown in fig 4.b, where the C matrix can be also seen. Both matices are decomposed as square w*w matrices. Each square matrix is split into three blocks, named U, L and D. The subscript indices notation in C reflects its correspondence with matrix C, and the superscript indices express the computational sequence of partial results attainment. To obtain the blocks for matrix C, we must compute:

$$D_{ij} = \sum_{t=0}^{k_p-1} D_{ij}^{t} \qquad U_{ij} = \sum_{t=0}^{2k_p-1} U_{ij}^{t} \qquad L_{rs} = \sum_{t=0}^{2k_p-1} U_{rs}^{t}$$

Fig. 5 shows a feedback topology suited to complete the above computations. In (7) the systolic arrays with this kind of feedback scheme are named "spiral systolic arrays". The main diagonal is "auto-feedbacked", and the sub-diagonals are feedbacked in pairs, in such a way that the number of processing elements in the loop equals w.

In the computation of U_{ij} and L_{rs}, the feedback delay time equals w, except in some special cases where this delay time is greater than w. These irregularities in the feedback delay time arise because we aim to minimize the computational time required. Nevertheless, a regular delay time can be reached, and therefore a simplification of the control section attained, at the expense of increasing the global computational time. To achieve this goal of regularity, the original problem should be partitioned into k_m subproblems, each subproblem consisting of multiplication of matrix A by every column submatrix of B. The transformed problems, from these subproblems, cannot be linked together without a loss of efficiency; to maintain the calculations, separation of subproblems with zero value blocks is needed.

For the case we are considering, three types of irregularities in the feedback delay time can be recognized. One of these types arise when the blocks U_{0j} are feedbacked. The other two types arise when feedback of blocks $L_{k_n-1,j}$ is used.

To calculate blocks U_{0j}, a partial result is first needed, namely

$$SU_{0j} = \sum_{t=0}^{2k_p-2} U_{0j}^t$$

and the feedback delay time of partial results equals w. The feedback delay of the last partial result is $6(w-1)(k_n-1)k_p+w$.

The calculation proces for $L_{k_p-1,0}$ is the same as for U_{0j}, but now the required feedback delay of the last partial result is $6(k_nk_p)(k_m-1)(w-1)+w$.

When calculating $L_{k_n-1,j}$, with $j<>0$, the feedback delay of the first partial result is $6(w-1)(k_n-1)k_p+w$. Once the second partial result has been obtained, the following ones are obtained every w cycles.

To use feedback with constant delay time makes necessary a number of 2w and w memory elements, for the main diagonal and sub-diagonals, respectively. With regard to the irregular feedbacks, w(w-1)3/2 memory elements are needed.

Taking into account these features, as well as the feedback topology, a modular and easily expandible design is possible for the systolic array system.

The number of steps required to solve the problem is: $T = 2wk_pk_nk_m + 3w - 5$. The processor utilization is given by

$$\frac{n.m.p}{AT} = \frac{1}{3 + \frac{3}{k_nk_mk_p} - \frac{5}{k_nk_mk_pw}}$$

A formal description of the data flow that must feed the array, can be found in the Appendix of this paper. The input of data must be according to the diagonals direction, to attain the computation of C=A*B+E without need of any operation outside the array. The description of the data flow is made and presented at a D, L and U blocks level.

4.- CONCLUSIONS

A transformation method for arbitrarily sized matrices has been presented in this paper. The aim is to use, with a maximum efficiency, the systolic arrays proposed by H.T. Kung. The method consists in the transformation of a dense matrix, A, into a banded matrix, **A**; this banded matrix has a bandwidth equal to the array size. To achieve this goal, A is split into triangular sub-matrices. Later, a general juxtaposition algorithm (DBT) for triangular sub-matrices, is used, and the transformed matrix is obtained.

Matrix transformation algorithms that have been presented in this paper are of two kinds: **by rows and transpose-by-rows**. These algorithms allow utilization of systolic arrays without any efficency loss when solving problems of the classes matrix-vector and matrix-matrix multiplication.

The number of memory elements required depends only on the array size. The topology of feedback flows is regular and fixed for a given system. Moreover, all the computations are

made inside the systolic system, and modular, expandible structures are adequate to this purpose.

From the proposed transformations, some other related types of transformations are easily deduced, and simplification of the feedback scheme are attained, at the cost of a lower optimization of the PE's utilization

In the case of computing with matrices of a known degree of sparsity, transformation algorithms can be devised and developed, to exclude the need of zero-valued elements sub-matrices. A reduction of computational time would be the consequence of using such algorithms.

The methodology that has been presented in this paper has been applied to solve the problems of linear equations triangular systems, Gauss-Seidel iterative method and L-U decomposition. Other types of matrix algorithms are presently in study, and the hardware design of a prototype system is under execution.

APPENDIX

The flow of data that must feed the array system to accomplish the computation C=A*B+E, without any external operation, is expressed now.

A banded matrix with a value of width equal to 2w-1, is formed from the data introduced through the array diagonals; let us name this matrix as I (input). These data are partialy processed inside the array system, and another matrix is formed at the array output; let us name this matrix as O (output).

We divide these two matrices, I and O, into square sub-matrices with w*w elements, and each one of these

sub-matrices is again divided into lower triangular blocks (L), diagonal blocks (D), and upper triangular blocks (U).

To denote the sub-matrices in the band of every row of blocks, subscript indices are used as shown in Fig. 6.

Each sub-matrix name receives one superscript index, denoting the matrix to which it belongs.

A input matrix I, is formed from the data matrix E and from the feedbacked array output, 0.

I can be composed as follows:

For $0 \leqslant k \leqslant k_n k_m$

$$U^I_{k,0} = \begin{cases} U^0_{k-k_p(k_n-1)-1,1} & \text{if } k \bmod k_p k_m = 0 \\ U^E_{k/k_p \bmod k_n, \lfloor k/k_p k_n \rfloor} & \text{if } k \bmod k_p = 0 \text{ and } k \bmod k_p k_n = 0 \\ U^0_{k-1-,1} & \text{otherwise} \end{cases}$$

$$L^I_{k,0} = \begin{cases} L^0_{k-k_p(k_n-1)-1,1} & \text{if } (k+k_p) \bmod k_p k_n = 0 \text{ and } k = k_p(k_n-1) \\ L^E_{k/k_p \bmod k_n,\ k/k_p k_n} & \text{if } k \bmod k_p = 0 \text{ and } (k+k_p) \bmod k_p k_n = 0 \\ L^0_{k-1,1} & \text{otherwise} \end{cases}$$

$$D_k^I = \begin{cases} D^E_{k/k_p \bmod k_n,\ \lfloor k/k_p k_n \rfloor} & \text{if } k \bmod k_p = 0 \\ D^0_{k-1} & \text{otherwise} \end{cases}$$

$$U^I_{k,1} = \begin{cases} U^E_{0,\ k/k_p k_n} & \text{if } k \bmod k_p = 0 \\ U^0_{k,0} & \text{otherwise} \end{cases}$$

$$L^I_{k,1} = \begin{cases} L^0_{k_p k_n - 1,\ 0} & \text{if } k = k_p k_n k_m - 1 \\ L^E_{k_n - 1,\ k+1/k_p k_n} & \text{if } (k+1) \bmod k_p k_n = 0 \text{ and } k = k_p k_n k_m \\ L^0_{k,0} & \text{otherwise} \end{cases}$$

The submatrices from the results matrix C are attained, at the array output, as follows:

For $0 \leqslant i, < k_n$ $\qquad$ $0 \leqslant j < k_m$

$$L^C_{ij} = \begin{cases} L^0_{k_n k_p k_m - 1,\ 1} & \text{if } (i,j) = (k_n - 1, 0) \\ L^0_{(j+1)k_p k_n - 1,\ 0} & \text{if } i = k_n - 1,\ j > 0 \\ L^0_{(i+jk_n+1)k_p - 1,\ 1} & \text{otherwise} \end{cases}$$

$$D^{C}_{ij} = D^{0}_{(i+jk_n+1)k_p-1}$$

$$U^{C}_{ij} = \begin{cases} U^{0}_{(j+1)k_pk_n,0} & \text{if } i = 0 \\ U^{0}_{(i+jk_n+1)k_p-1,1} & \text{otherwise} \end{cases}$$

REFERENCES.

(1) C.L. Seitz, "Concurrent VLSI Architectures", IEEE Trans. on Computers, Vol. C-33. N 12, Dec. 1984, pp. 1247-1265.

(2) K. Hwang and Y.H. Cheng, "Partitioned Matrix Algorithms for VLSI Arithmetic Systems,". IEEE Transactions on Computers, Vol. C-31. No. 12, December 1982 p.p. 1215-1224.

(3) H.Y. Chuang and G. He, "A Versatile Systolic Array for Matrix Computations", 12th Anual International Symposium on Computer Architecture 1985. Conference Proceedings, pp. 315, 322.

(4) H.T. Kung and C.E. Leiserson, "Systolic Arrays (for VLSI)", in I.S. Duff and G.W. Stewart (editors), Sparse Matrix Proceedings 1978, Society for Industrial and Applied Mathematics, p.p. 256-282, 1979.

(5) C. Mead and L. Conway, Introduction to VLSI Systems, Addison-Wesley, 1980. Section 3 of Chapter 8.

(6) R.W. Priester, H.J. Whitehouse, K. Bromley, J.B. Clary, "Signal Processing with Systolics Arrays," Proceedings of the 1981 International Conference on Parallel Processing 1981, pp. 207-215.

(7) S.Y. Kung, "VLSI Array Processors," IEEE ASSP Magazine, July 1985, pp. 4-22.

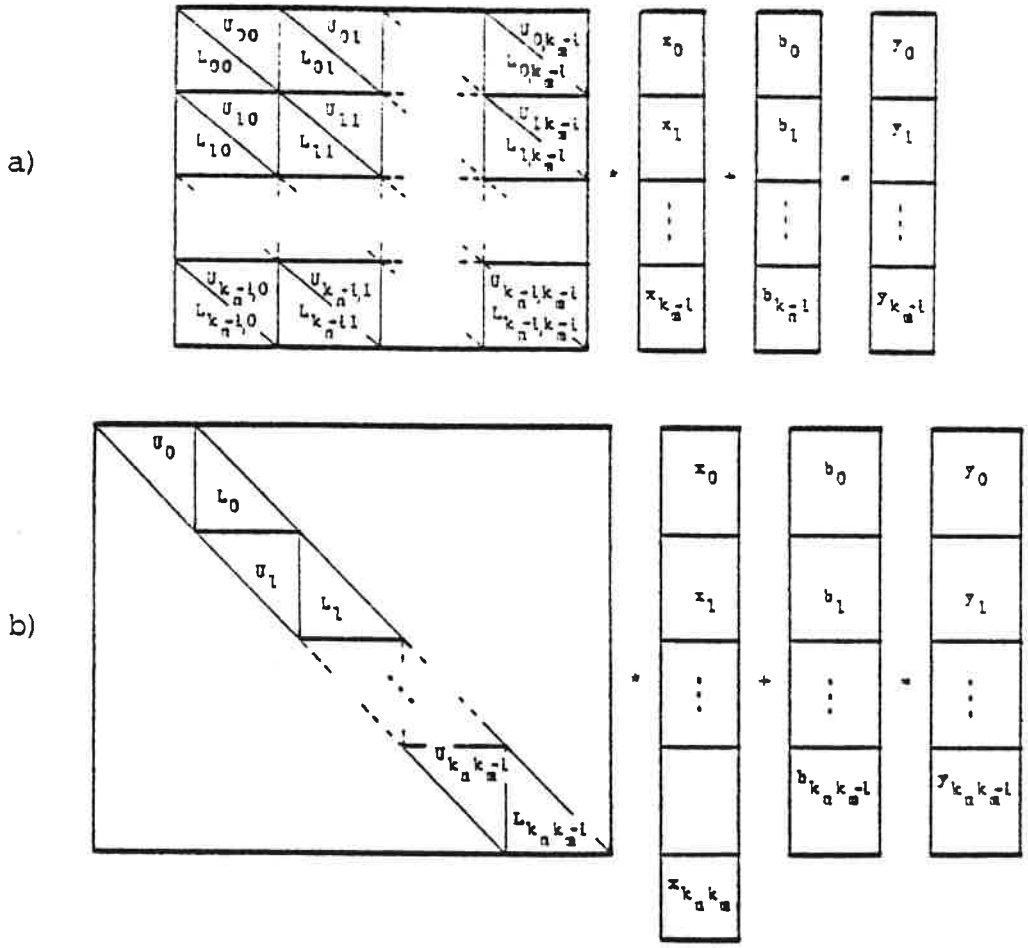

Fig. 1: Blocks structure of the problem Ax+b=y
a)Original problem
b)Transformed problem

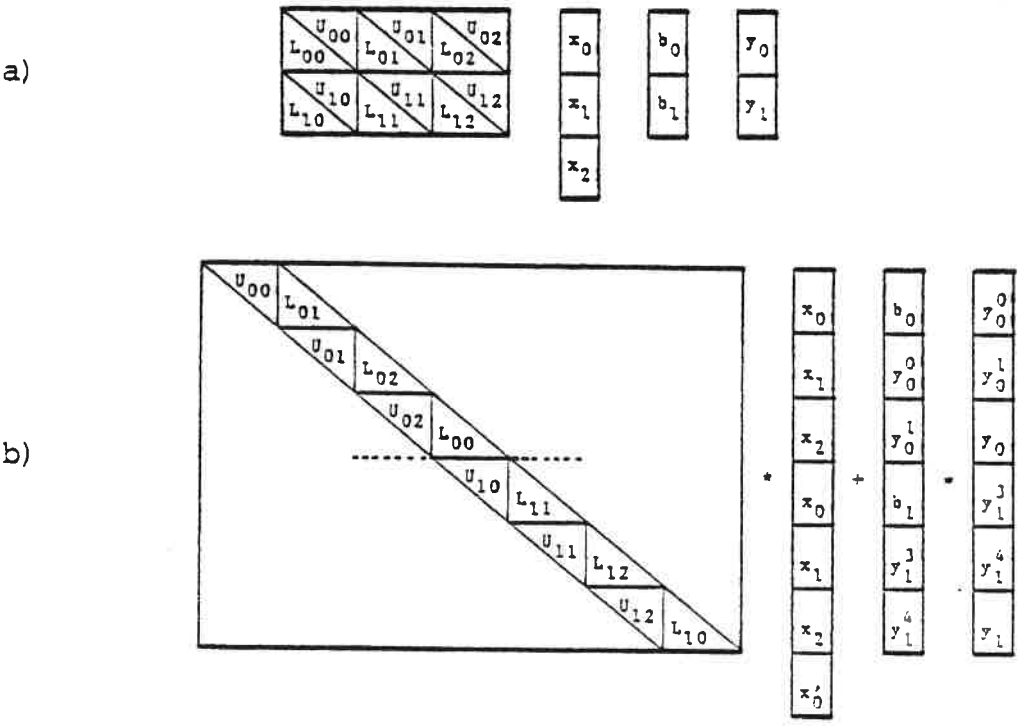

Fig. 2: Blocks structure of the problem Ax+b=y for $k_n=2$, $k_m=3$. a) Original problem. b)Transformed problem through the DBT-by-rows algorithm.

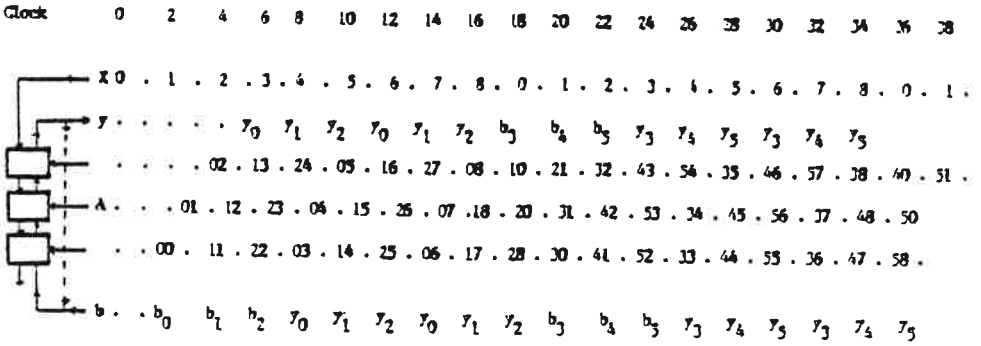

Fig. 3: Input and output data flow in every cycle for the problem Ax+b=y with n=6, m=9, w=3.

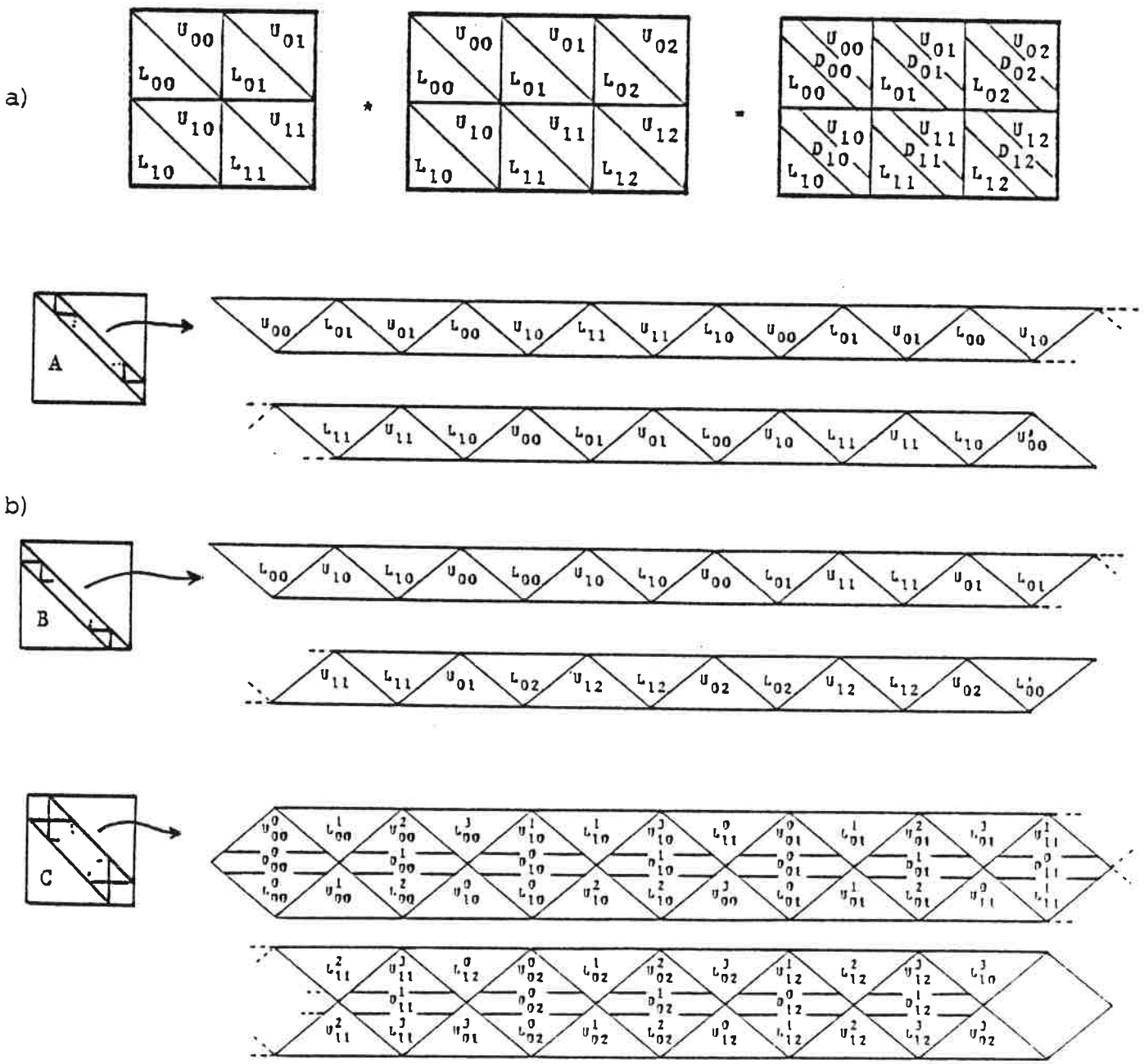

Fig. 4: Blocks structure of the problem AB=C with $k_n=2$, $k_p=2$, $k_m=3$. a) Original problem. b)Transformed problem.

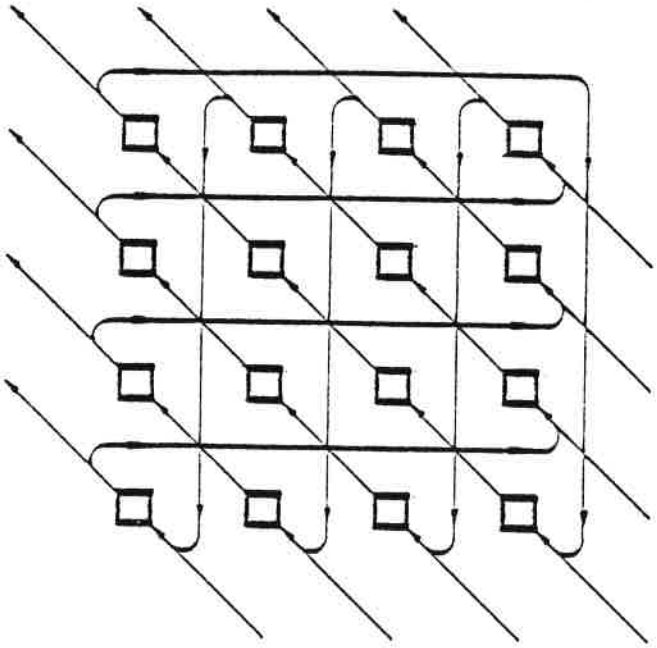

Fig. 5: Interconnection topology of the feedback scheme in the spiral systolic array.

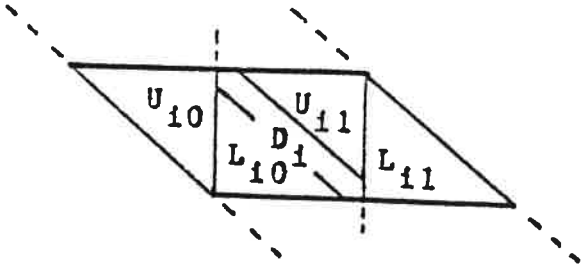

Fig. 6: Row block i of matrices I(input) and O (output).