

Systolic Implementation of Nonlinear Transformation of Two Sequences

E. I. Milovanović, I.Ž. Milovanović, B. M. Randjelović, M. K. Stojčev

Abstract— This paper concerns with linear bidirectional systolic array (BLSA) since it can be applied for solving a broad class of scientific and technical problems, such as matrix-vector multiplications, eigen value computation, solving systems of linear equations iteratively. This makes this array some kind universal. The array with the same topology, but different functionality of PEs, can be applied for solving problems in operational research, graph theory, discrete mathematics, etc. In this paper we will show how a nonlinear transformation of two given sequences (a_1, a_2, \dots, a_n) and (b_1, b_2, \dots, b_n) and matrix $W = (w_{ij})$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, n$, can be computed on BLSA. The transformation can be described as $c_{ij} = w_{ij} \oplus a_i \odot b_j$, for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$. Binary operations \oplus and \odot are closed, associative and commutative. This transformation represents a mathematical model for broad class of problems that are met in operational research, graph theory, discrete mathematics, etc. We illustrate the presentation on the example of finding shortest path in a given graph.

Keywords— Systolic array, Matrix multiplication, Shortest path

I. INTRODUCTION

During the previous decades, enormous technological advances have been achieved in the area of VLSI circuits. Such technological advances gave rise to a totally new way of computing, constituted of highly parallel computing systems for specific applications. An interesting approach is the so-called systolic array computing, proposed originally by Kung [2], at the end of the seventies. A systolic array is parallel computing device for a specific application, that is constituted of large number of processing elements (abbreviated as PEs), interconnected in a regular way, with local communication. In a similar way as blood circulates in the human body, data circulate inside the PEs of a systolic array, and interact with other data. Results computed in PEs are again pumped into other PEs for further computing. The regularity of these arrays leads to inexpensive and dense VLSI implementations, which imply high performance and low cost. Application specific processor arrays fit naturally into the concept of hardware library, where functional units are in relation to the host computer as subroutines from a software library are to production code.

I.Ž. Milovanović, E. I. Milovanović, B. M. Randjelović, M. K. Stojčev are with the Faculty of Electronic Engineering, University of Niš, Beogradska 14, 18000 Niš, Serbia, E-mail: ema@elfak.ni.ac.yu

Systolic arrays have been designed for a wide variety of computationally intensive problems in signal processing, numerical problems, pattern recognition, database and dictionary machines, graph algorithms etc. Systolic arrays implemented in silicon chips are typically laid out in a linear array or bidimensional grid of cells. One dimensional or linear systolic arrays are especially popular because of low number of I/O pins required for the interconnection with the "outside world".

The most interesting linear systolic array is the bidirectional linear systolic array (BLSA). The reason why BLSA has leading role in the class of all 1D linear SA lies in the fact that this SA has many desirable properties, like maximal data-flow ([2], [3]), possibilities for fault-tolerant calculations ([7]) and applicability in many scientific and technical problems, such as matrix-vector multiplication ([2], [3], [9], [10]), correlation and convolution of two sequences ([4], [6]), iterative processes ([5]), linear and dynamic programming ([11]), stochastic processes ([8]), graph theory, etc. Of course, BLSAs for different problems may differ from each other with respect to functional properties of processing elements (PE), but they all have the same topology. To clarify this consider the following general problem. Assume that we have two sequences and a matrix. If we define two binary operations with some predefined properties, we can define transformation, i.e. algorithm, for calculating elements of a new matrix, obtained from the elements of given sequences and matrix. That algorithm would have high degree of universality, because it could be applied in many problems in science and technique, by choosing binary operations and elements of sequences and matrix. In this paper we synthesize BLSA suitable for the implementation of such transformation with minimal (i.e. optimal) number of PEs for a given problem size, and minimal possible execution time for this number of PEs. Then we illustrate obtained results on the example of calculating the shortest path in a given graph.

II. PROBLEM DEFINITION

Let R be the set of real numbers and R^* be its extension, i.e. $R^* = R \cup \{+\infty\}$. Now, introduce two binary operations \oplus and \odot on the set R^* , that are closed, associative and commutative.

Further, assume we have two sequences (a_i) , $i = 1, 2, \dots, n$ and (b_j) , $j = 1, 2, \dots, n$, and matrix $W =$

(w_{ij}) , $i = 1, 2, \dots, n$, $j = 1, 2, \dots, n$, with elements from the set R^* . We want to compute matrix $C = (c_{ij})$, according to the following expression

$$c_{ij} = w_{ij} \oplus a_i \odot b_j, \quad (1)$$

for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$. Our goal is to implement computation (1) on the BLSA with optimal number of PEs with respect to a problem size, i.e. n . The execution time of this BLSA should be as minimal as possible. Figure 1 show the functional properties of the PE in BLSA.

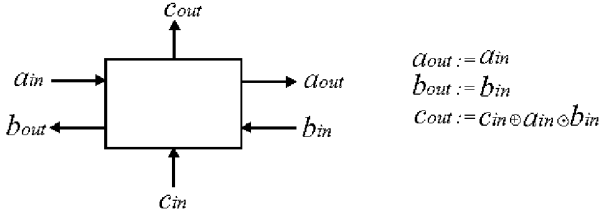


Fig. 1. Functional properties of the PE

III. GENERATING OF SYSTOLIC ALGORITHMS

The basic systolic algorithm which can be used for calculating (1) has the following form

Algorithm_1

```

for  $j := 1$  to  $n$  do
  for  $i := 1$  to  $n$  do
     $a(i, j, 1) := a(i, j - 1, 1)$ ;
     $b(i, j, 1) := b(i - 1, j, 1)$ ;
     $c(i, j, 1) := c(i, j, 0) \oplus a(i, j, 1) \odot b(i, j, 1)$ ;
  endfor  $\{i, j\}$ ;

```

where $c(i, j, 0) \equiv w_{ij}$, $a(i, 0, 1) \equiv a_i$, $b(0, j, 1) \equiv b_j$, $c(i, j, 1) \equiv c_{ij}$ for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$.

The BLSA is usually designed according to the projection direction $\vec{\mu} = [1 \ 0 \ 1]^T$ (see for example [9], [10]). But, in our case, this projection vector does not lead us towards the BLSA with desired properties. Therefore, we introduce a new approach in designing BLSA. Namely, we synthesize BLSA as "degraded" two-dimensional orthogonal SA, obtained according to the projection direction $\vec{\mu} = [1 \ 1 \ 0]^T$. Degradation means that during the synthesis process we obtain 1D orthogonal SA, topologically identical as BLSA, instead of 2D orthogonal SA.

Note that properties of binary operations \oplus and \odot in (1) enables computation of matrix C elements by any order, i.e. by any permutation of index set $\{1, 2, \dots, n\}$, for both index variables i and j . We use this property to minimize the number of PEs in BLSA. Therefore, instead of Algorithm_1, we define an equivalent algorithm which is adjusted to the direction $\vec{\mu} = [1 \ 1 \ 0]^T$ (see for

example [1], [12]). The adjustment can be performed by any index variable. The following algorithm is adjusted to the direction $\vec{\mu} = [1 \ 1 \ 0]^T$, by index variable i .

Algorithm_2

```

for  $j := 1$  to  $n$  do
  for  $i := 1$  to  $n$  do
     $a(i, i + j - 1, 1) := a(i, i + j - 2, 1)$ ;
     $b(i, i + j - 1, 1) := b(i - 1, i + j - 1, 1)$ ;
     $c(i, i + j - 1, 1) := c(i, i + j - 1, 0) \oplus a(i, i + j - 1, 1) \odot b(i, i + j - 1, 1)$ ;
  endfor  $\{i, j\}$ ;

```

where $a(i, j + n, 1) \equiv a(i, j, 1) \equiv a_i$, $b(i, j + n, 1) \equiv b(i, j, 1) \equiv b_j$, for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$.

If the adjustment is performed by index variable j , then the systolic algorithm has the following form

Algorithm_3

```

for  $j := 1$  to  $n$  do
  for  $i := 1$  to  $n$  do
     $a(i + j - 1, j, 1) := a(i + j - 1, j - 1, 1)$ ;
     $b(i + j - 1, j, 1) := b(i + j - 2, j, 1)$ ;
     $c(i + j - 1, j, 1) := c(i + j - 1, j, 0) \oplus a(i + j - 1, j, 1) \odot b(i + j - 1, j, 1)$ ;
  endfor  $\{i, j\}$ ;

```

where $a(i + n, j, 1) \equiv a(i, j, 1) \equiv a_i$, $b(i + n, j, 1) \equiv b(i, j, 1) \equiv b_j$, for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$.

Since we consider square case in (1), i.e. $i = 1, 2, \dots, n$, $j = 1, 2, \dots, n$, the results will be the same for both Algorithm_2 and 3. Without affecting the generality, in further analysis we will use Algorithm_2.

IV. SYNTHESIS OF BLSA

Here we are not going to derive formulas for BLSA synthesis. Instead, we will use those derived in [1], [10], [12].

Denote with $PE \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}$ position of the PE in the synthesized BLSA, and with $a(i, 0, 1) \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}_a$, $b(0, i + j - 1, 1) \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}_b$ and $c(i, i + j - 1, 0) \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}_c$ positions of input data elements at the beginning of the computation.

The BLSA that implements (1) according to Algorithm_2 is synthesized according to the following formulas

$$PE \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} j - 1 \\ 1 \end{bmatrix},$$

$$a(i, 0, 1) \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}_a = \begin{bmatrix} 1 - 2i \\ 1 \end{bmatrix} + (r_1 + r_2)\bar{n} \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

$$b(0, i + j - 1, 1) \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}_b = \begin{bmatrix} 2i + 2j - 3 \\ 1 \end{bmatrix} + (r_1 + r_2)\bar{n} \begin{bmatrix} -1 \\ 0 \end{bmatrix},$$

$$c(i, i + j - 1, 0) \rightarrow \begin{bmatrix} x \\ y \end{bmatrix}_c = \begin{bmatrix} j - 1 \\ 3 - 2i - j \end{bmatrix} + (r_1 + r_2)\bar{n} \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$, where \bar{n} is defined as

$$\bar{n} = \begin{cases} n, & \text{if } n \text{ is odd} \\ n - 1, & \text{if } n \text{ is even.} \end{cases}$$

while r_1 is greater of the integers from the set $\{0, 1\}$ for which the following is valid

$$-2(i - 1) + r_1 \bar{n} < 0, \quad i = 1 \Rightarrow r_1 = 0. \quad (2)$$

Similarly, r_2 is greater of the integers from the set $\{0, 1\}$, for which the following holds

$$-2(i - 1) - (j - 1) + (r_1 + r_2) \bar{n} \leq 0.$$

Note that r_1 should be determined first.

Data flow in the BLSA that computes (1) according to Algorithm_2, for the case $n = 3$ is presented in Figure 2.

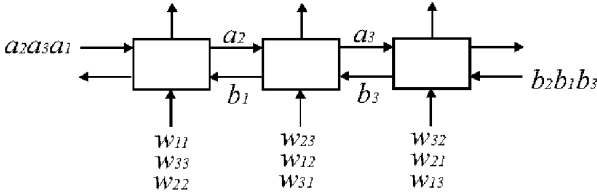


Fig. 2. Data flow in BLSA for the case $n = 3$

V. PERFORMANCES OF THE BLSA

Synthesized BLSA consist of $\Omega = n$ PEs, that is minimal number for the given problem size n . Total execution time of Algorithm_2 implemented on BLSA, is $T_{tot} = 3n - 2$, if n is odd, where $T_{in} = n - 1$, $T_{exe} = n$, $T_{out} = n - 1$, and $T_{tot} = 3n - 1$ (one time unit more) if n is even. The execution time can not be better for $\Omega = n$ PEs.

VI. APPLICATION

As we have mentioned in the Introduction, obtained BLSA can be used for systolic implementation of many problems, by choosing elements of sequences (a_i) and (b_j) and matrix $W = (w_{ij})$. We will illustrate this on the example of computing shortest path in a given graph.

Let $G = (V, E)$ be a given directed or undirected graph, where $V = \{x_1, x_2, \dots, x_n\}$ is the set of nodes, and $E = \{l_1, l_2, \dots, l_m\}$ is a set of arcs. Graph G can be described by a square matrix $D^{(0)} = (d_{ij}^{(0)})$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, n$, where $d_{ij}^{(0)}$ represents the distance between nodes x_i and x_j . We assume that $d_{ii}^{(0)} = 0$. If $d_{ij}^{(0)} = +\infty$, the arc between x_i and x_j does not exist.

Starting with matrix $D^{(0)}$ and recurrence relation

$$d_{ij}^{(k)} = \min \left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \right), \quad 1 \leq k \leq n, \quad (3)$$

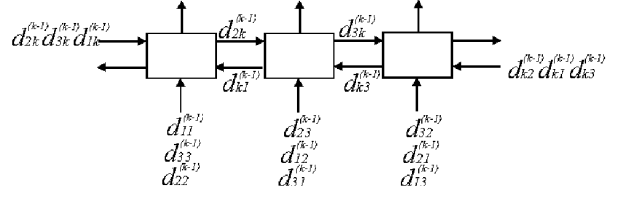


Fig. 3. Data flow in the BLSA that computes shortest path for some fixed k and $n = 3$.

we can obtain matrix $D^{(n)} = (d_{ij}^{(n)})$, that contains the values of the shortest paths between the nodes in a given graph $G = (V, E)$.

If in (1) we substitute a_i with $d_{ik}^{(k-1)}$, b_j with $d_{kj}^{(k-1)}$ and w_{ij} with $d_{ij}^{(k-1)}$ for each $i, j = 1, 2, \dots, n$ and some fixed k , and define binary operations \oplus and \odot as

$$x \oplus y = \min(x, y) \quad \text{and} \quad x \odot y = x + y,$$

for each $x, y \in R^*$, we obtain an expression that computes elements of matrix $D^{(k)}$ according to $D^{(k-1)}$. If we repeat this calculation n times, i.e. for $k = 1, 2, \dots, n$, we will obtain matrix $D^{(n)}$. It is not difficult to see that the total execution time needed to compute $D^{(n)}$ on BLSA is equal to $T_{tot} = n(2n - 1)$, when n is odd, and $T_{tot} = 2n^2$, when n is even. Data-flow in the BLSA that computes $D^{(k)}$ for some fixed k , $1 \leq k \leq 3$ and $n = 3$ is presented on Figure 3.

VII. CONCLUSION

We have defined a nonlinear transformation of two given sequences and a matrix, which represents mathematical model for a great number of problems that are met in scientific and technical applications. Then we have synthesized BLSA as a degraded bidirectional orthogonal SA which can be used to calculate the defined transformation efficiently. Depending of the type of operations involved in the transformation, structure of the processing element in the BLSA may vary, while the interconnection pattern remains the same. The obtained BLSA is optimal with respect to a number of processing elements for a given problem size. Execution time is minimized for a given number of PEs. We have illustrated the presentation on the example of determining the shortest path in a given graph.

REFERENCES

- [1] M.P. Bekakos, E.I. Milovanović, N.M. Stojanović, T.I. Tokić, I.Ž. Milovanović, I.Z. Milentijević, *Transformation Matrices for Systolic Array Synthesis*, J. Electrotehn. Math., **7**(2002), 9-15.
- [2] H.T. Kung, "Why Systolic Architectures?," *Computer* **15** (1982), 37-46.
- [3] S.Y. Kung, *VLSI array Processors*, Prentice Hall, New Jersey, 1988.
- [4] Y.-C. Lin, *Array Size Anomaly of Problem-Size Independent Systolic Arrays for Matrix-Vector Multiplication*, *Parallel Comp.* **17** (1991), 515-522.
- [5] K.G. Margaritis, D.J. Evans, *Folding Techniques for Systolic Iterations*, *Parallel Algor. Appl.*, **7** (1995), 87-105.

- [6] I.Z. Milentijević, I.Ž. Milovanović, E.I. Milovanović, *Solutions for Convolution on Systolic Arrays for Matrix-Vector Multiplication*, 3rd Inter. Confer. on Telecommunication in Modern Satellite, Cable and Broadcasting Services (B. Milovanović, ed.), Niš'97, Vol. 1 (1997), 271-274.
- [7] E.I. Milovanović, I.Z. Milentijević, I.Ž. Milovanović, T.I. Tokić, *ABFT Technique for Matrix Multiplication on Space-Time Optimal Systolic Array*, FILOMAT, 11 (1997), 109-117.
- [8] E.I. Milovanović, R.M. Stanković, M.K. Stojčev, I.Ž. Milovanović, *Calculating State Vector in Stochastic Processes on Linear Systolic Array*, International Scientific Conference, UNITECH'02, Gabrovo (2002), 654-657.
- [9] I.Ž. Milovanović, E.I. Milovanović, I.Z. Milentijević, *New Efficient Systolic Array for Matrix-Vector Multiplication*, 3rd Inter. Confer. on Telecommunication in Modern Satellite, Cable and Broadcasting Services (B. Milovanović, ed.), Niš'97, Vol. 1 (1997), 271-274.
- [10] I.Ž. Milovanović, E.I. Milovanović, I.Z. Milentijević, M.K. Stojčev, *Designing of Processor-Time Optimal Systolic Arrays for Band Matrix-Vector Multiplication*, Comput. Math. Appl. Vol. 32, 2(1996), 21-31.
- [11] I.Ž. Milovanović, R.M. Stanković, E.I. Milovanović, M.K. Stojčev, *Implementation of Optimal Investment Problem on Linear Systolic Array*, International Scientific Conference, UNITECH'02, Gabrovo (2002), 650-653.
- [12] I.Ž. Milovanović, E.I. Milovanović, M.K. Stojčev, T.I. Tokić, N.M. Stojanović, *Determining Space Parameters in Systolic Array Design*, FILOMAT, 15 (2001), 55-60.