A FAST SYSTOLIC ALGORITHM FOR MATRIX MULTIPLICATION

G. ISERN

UNIVERSIDAD CENTRAL DE VENEZUELA
ESCUELA DE COMPUTACION
Apartado 47002, Caracas
Venezuela

ABSTRACT

This paper describes a very fast Systolic Array for Matrix Product. Systolic Sistems usually achieve high performance by allowing computations to be pipelined over a large array of processing elements. To achieve even higher performance the systolic array proposed in this paper utilizes a column broadcasting technique plus a second level of pipelining by allowing the processing elements themselves to be pipelined. This systolic array shows not only high performance, but also optimality in terms of area and time complexity by applying recursively the fitting rows method to reduce the I/O bandwidth and speedup the matrix product algorithm

Keyword: Systolic algorithms , pipeline , matrix multiplication, speedup , VLSI.

1.- INTRODUCTION

The developments in microelectronics have revolutionized computer design. Integrated circuit technology has increased the number and complexity of components that can fit on a chip. As a result, this new technology makes it feasible to build low cost special devices to solve sophisticated problems. These devices are based on multiprocessor structures and in the use of parallel algorithms. For these reasons, our main interest is in designing parallel algorithms with simple and regular data flows and using pipelining as a general method for implementing these algorithms in hardware. The results are Systolic arrays , which is the name given by Kung [1] to a single and regular interconnection of processor elements which cooperate to carry on an especific task (matrix multiplication in this case). Pipelining a set of processor elements plus pipelining within each stage should lead to the best possible performance this is optimize area and time obtaining a great improvment with respect to previous work.

The first contribution to systolic system was made by Kung and Leiserson in 1978 [1], Suros and Montagne [2, 3] optimized the systolic networks proposed by Kung by fitting first diagonals and then fitting rows for reducing I/O bandwidth, Isern [7] and Montagne , Suros and Isern [4] obtained acceleration by the use of a systematic aligning method called row fitting which also reduce the area of the systolic array. Kung also proposed a two level pipelined systolic array [5] for an algorithm of multidimensional convolution.

Section 2 describes one level pipeline matrix-multiplication with regular structure and fitting rows, showing estimates of acceleration and Area. Section 3, describes the new proposed algorithm, the fitting rows model for a two levels piplined systolic array with broadcasting, and finally section 4 shows results and conclusions.

2.- MATRIX PRODUCT ON SYSTOLIC ALGORITHM:

Before describing systolic arrays, some concepts are to be defined:

(1) Elemental processor: a processor that executes

$$c = c + (a*b)$$

(2) $T_s(n)$: Sequencial execution time

(3) $T_p(n)$: Parallel execution time

(4) EPU: Efficiency of processor utilization

$$EPU = T_s(n)/T_p(n)*P$$

where P is the number of processors and 0 EPU 1

(5) Area: Number of processors in a systolic network

(6) IPSP : Inner product step processor.

A well-know systolic array to compute an n * n band matrix-vector multiplication AX = Y was originally reported elsewhere [8]. Figure 1 and 2 depicts this organization for a band matrix and the first seven steps of the algorithm.

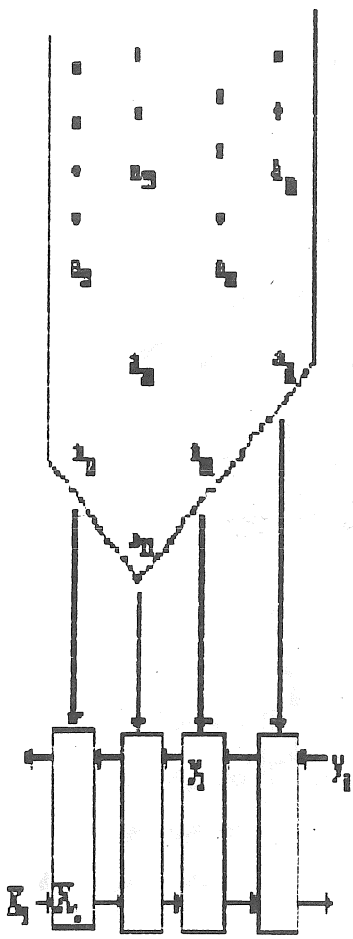

FIGURE 1

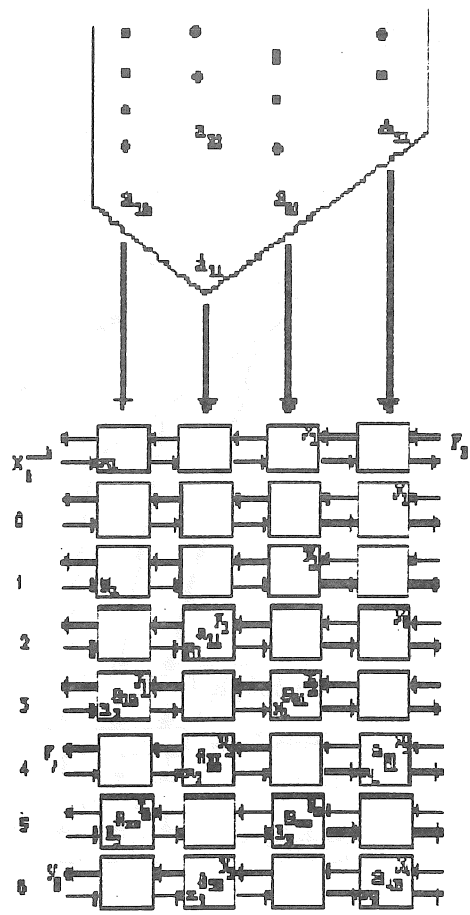

FIGURE 2

For the case of matrix multiplication it is easy to see that the matrix product $C = (c_{ij})$ of $A = (a_{ij})$ and $B = (b_{ij})$ can be computed by the following recurrences.

$$c_{ij}^{(1)} = 0,$$
$$c_{ij}^{(k+1)} = c_{ij}^{(k)} + a_{ik}\, b_{kj},$$
$$c_{ij} = c_{ij}^{(n+1)}$$

This recurrences can be evaluated by pipelining the a_{ij}, b_{ij} and c_{ij} through a systolic array having w_1. w_2 (bandwidth of A and B respectively) Hex-connected IPSP or through a systolic array having W.N (NxN matrix) orthogonally-connected IPSP (see figure 3). The latter will be the one used all along this work, this one is an extension of Leiserson/Kung matrix-vector multiplication algorithm for N vectors.

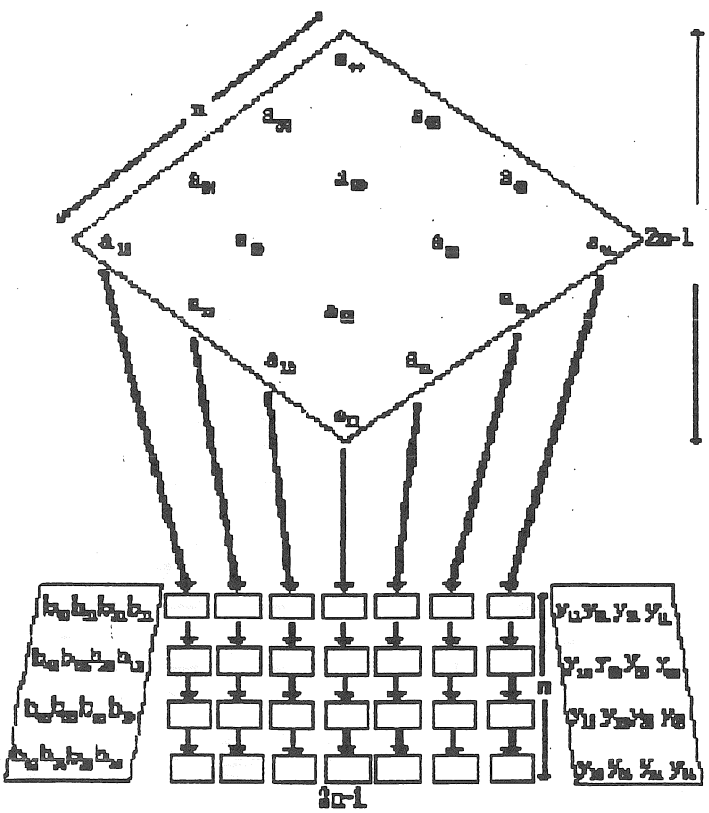

FIGURE 3

Let A = (aij) be a nxn matrix with bandwidth w = 2n-1 and X a n elements vector. It is readily seen that after w units of time (filling time FT) the results of the product Y=Ax start shifting out from the left-end processor at the rate of one output every two time units. Therefore, using our network all the n components of Y can be computed in 2n-1+w time units, which compared with the O(wn) time units needed for the sequential algorithm on a uniprocessor, is a great improvement.

Let see the formula:

a) Filling time FT = w
= 2n-1

b) Every two clock pulses (Figure 5) it is produced one result, therefore 2n cycles are necesary for the total vector
Computing Time CT = 2n - 1

c) Execution Time T(n) = FT + CT
= 4n - 2

Area = w = 2n - 1

Making an extension of the algorithm over a systolic network (adding rows), we obtain a network for multiply a matrix times several vectors, therefore an architecture for matrix product (orthogonally-connected, see figure 3).

The execution time for this model can be described in this way:

a) First Row filling time = 2n - 1

b) First vector computing time: 2n - 1

c) N-1 remaining vectors computing time: n-1

$T_m(n) = 2n - 1 + 2n - 1 + n-1 = 5n-3$

$\text{Area}_m(n) = w \times n$
$= (2n - 1)n$
$= 2n^2 - n$

Fitting Rows Method (FRM)

Proposed by Suros and Montagne (SMM, Suros/Montagne Model) in 1987, describes how modifying the processor internal architecture a_{ij} stays for two IPSP cycles eliminating the every two clocks idle processor cycle. It is required one application of FRM on B matrix to keep dataflow consistency (see Figure 4).

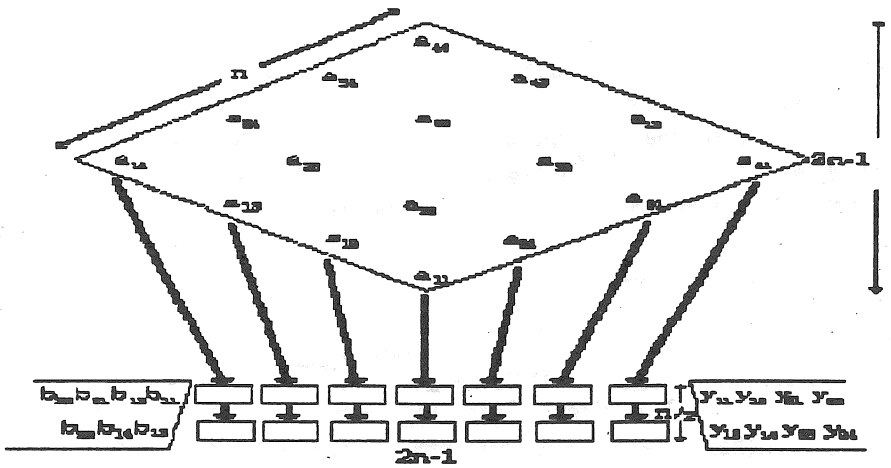

FIGURE 4

Definition

Let B = (b_{ij}) a n x n Matrix, f_1 and f_2 two adjacents n elements B rows.

$f_1 = (b_{i,1} ; b_{i,2} ; \ldots , b_{i,n})$

$f2 = (b_{i+1,1} ; b_{i+1,2} ; \ldots ; b_{i+1,n})$

A row fitting is defined as:

$f_\partial = f_1 \underline{o} f_2$

$f_\partial = (b_{i,1} ; b_{i+1} ; \ldots ; b_{i,n} ; b_{i+1,n})$

where length (f_∂) = Length (f_1) + Length (f_2)

Therefore every n rows B matrix can be transformed to n/2 rows B matrix using FRM and the execution time of the systolic network using FRM is given by:

a) First row filling time: 2n-1 (w)
b) First two vectors computing time: 2n
c) n/2 - 1 remaining vectors computing
Time : n/2 - 1

$T_{m_\partial}(n) = 2n - 1 + 2n + n/2 - 1$

$= 4.5\, n - 2$

$$\begin{aligned} \text{Area}_m^{\partial}(n) &= \text{Area}_m(n)/2 \\ &= (2n^2-n)/2 \\ &= n^2 - n/2 \end{aligned}$$

Applying recursively the FRM we reduce the I/O bandwidth to one, but with less speed up.

3. TWO LEVEL PIPELINING PLUS FITTING ROWS METHOD (TLFRM)

The two techniques to be described here will accelerate the matrix product with the same area of FRM or reducing it by applying recursively n times FRM [4].

The first model, consists in broadcasting the a_{ij} values to all IPSP in the same column of the systolic network at the same time, reducing the time for matrix multiplication to the time of matrix-vector multiplication [6] (see Figure 5)

$T_{mb}(n) = 4n-2$ (without FRM)

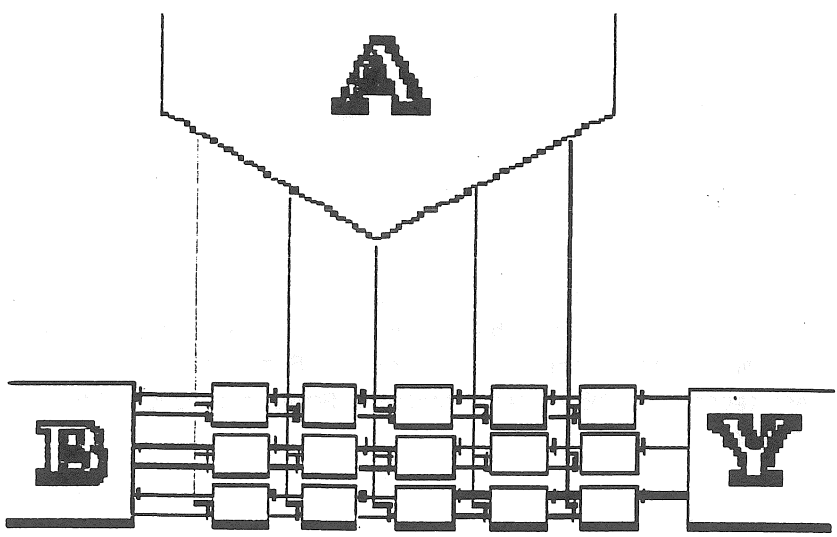

FIGURE 5

Combining the latter technique with FRM we obtain, a lot faster matrix product than with LKM (Leiserson/Kung Model) or FRM, using a smaller area, therefore better EPU is achieved.

The second method uses a second level of pipelining inside each IPSP, consisting of a two four-stages floating point operators, one for floating point addition and the other one for floating point multiplication, therefore the real pipe length is:

$$\begin{aligned} LENGTH_{2pipe}(n) &= w.8 \\ &= (2n-1).8 \\ &= 16n - 8 \quad \text{Elements} \end{aligned}$$

This change generates some minors modifications to the IPSP owing to the operands synchronization of the add operator with the multiplication result (see Figure 6). In order to synchronize it is necessary to attach a four stages shift register. The systolic network general operation will be developed using FRM.

This model executing time is:

a) Filling Time: $FT = w$
$= 2n-1$

b) First two vectors computing time is equal to the corresponding time in the FRM divided by the length of the inner pipeline (eight in this study). This is:

$$CT = 2n/8$$

c) The n/2 remaining rows computing time has two values depending on which is the chosen method, in the straight one (IMM, Isern/Montagne Model), the time is (n/2-1)/8 but combining with the broadcast technique (IMBM, Isern/Montagne Broadcast Model) this time is equal to zero.

$$\begin{aligned} T_{m\ \partial\ 2^{(n)}} &= 2n-1 + 2n/8 + n/16 - 1/8 \\ &= 2.31\,n - 1.12 \\ T_{m\ \partial\ 2b^{(n)}} &= 2n-1 + 2n/8 \\ &= 2.25n - 1 \end{aligned}$$

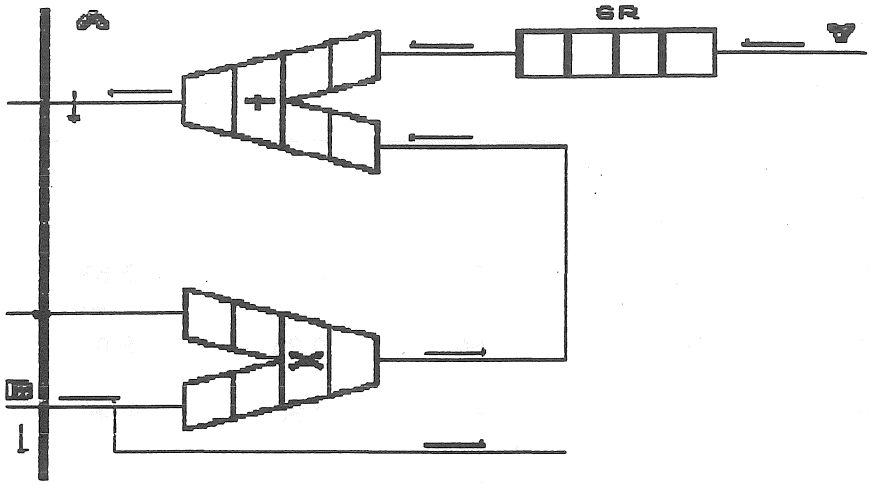

FIGURE 6

The systolic network area is equal to the one obtained by the FRM. It is possible to apply recursively the FRM reducing 50% of the precedent matrix on each application (IM2M, IM3M, Isern/Montagne Model with 2 and 3 fitting rows respectively). Combining FRM with two level pipelining method we obtain a very fast algorithm with the minimum processor area.

$$T_{m_{\partial}\ 2^{2(n)}} = 2n - 1 + 4n/8 + n/4.8 - 1/8$$
$$= 2.53\,n - 1.12$$
$$Area_{m_{\partial}\ 2^{2(n)}} = n^2/2 - n/4$$

TABLE 1 and 2 sumarizes executing times for the studied methods, in constrast with von Neumann Model, VNM.

MODEL	TIME	AREA
VNM	n^3	-
LKM	5n - 3	$2n^2 - n$
SMM	4,5n - 2	$n^2 - n/2$
IMM	2,31n - 1,12	$n^2 - n/2$
IMBM	2,25n - 1	$n^2 - n/4$
IM2M	2,53 - 1,12	$n^2/2 - n/4$

TABLE 1

MODEL	TIME	AREA	EPU	SPU
VNM	512	-	-	-
LKM	37	120	0,11	13,83
SMM	34	60	0,25	15,05
IMM	17,36	60	0,50	29,49
IMBM	17	60	0,51	30,11
IM2M	19,12	30	0,89	26,77

TABLE 2 (n=8)

Figure 7 show the executing time and figure 8 show the area.

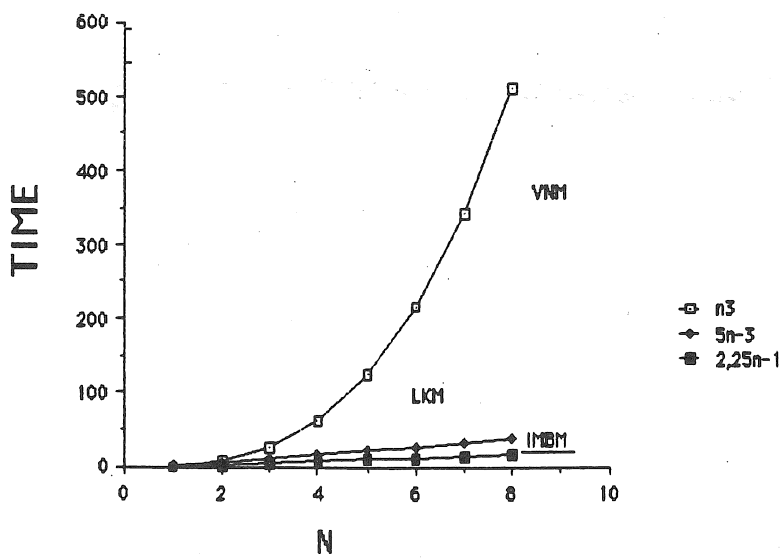

FIGURE 7

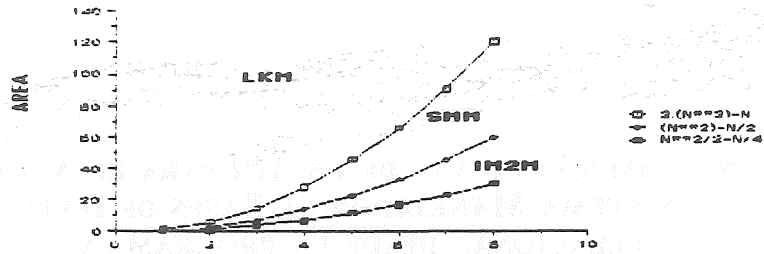

FIGURE 8

4. CONCLUSION:

In conclusion, by the application of this method to any matrix we can halve the computation time in comparison with SMM, keeping the same area, furthermore it is possible to have the minimum area with the recursively FRM application [4] and still having a better time than the SMM.

REFERENCES

[1] H.T. Kung and C.E. Leiserson, Algorithms for VLSI Processor Arrays, in: C. Mead and L. Conway, Eds., Introduction to VLSI Systems (Addison-Wesley, Reading, MA, 1980) Section 8.3 pp. 271-292.

[2] Surós, R. and Montagne, E. "Optimizing Systolic Networks by Fitting Diagonals"., Parallel Computing, North-Holland 4 (1987) 167-174.

[3] Surós, R. and Montagne, E. "Fitted Diagonals for Reducing I/O Bandwidth in Systolic Systems", Information Processing Letters, North-Holland (1987).

[4] Montagne E., Suros R., Isern G., "A deterministic model for spatial data scheduling on systolic arrays" Advances in parallel computing, volume 2 pages 291-304, 1992 JAI Press Ltd.

[5] H.T. Kung, "A two-level pipelined systolic array for multi-dimensional convolution" Department of Computer Science, Carnegie-Mellon University, USA, Nov. 1982.

[6] K.H. Cheng and S. Sahni, "VLSI Systems for band matrix multiplication", Parallel Computing, Vol. 4, Number3, June 1987 North-Holland, 239-258.

[7] Isern G. "Optimizacion del tiempo de ejecucion y del numero de procesadores en un algoritmo sistolico de multiplicacion de matrices". Tesis de Mestria, Facultad de Ciencias , UCV, 1990.

[8] Leiserson, C. 1983, Area-Efficiency VLSI Computation, ACM Doctoral dissertation award, MIT Press, Mass.